

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ
ОБМІНУ ПОВІДОМЛЕННЯМИ
SOFTWARE FOR INSTANT MESSAGING

Виконав(ла): студент(ка) 4 курсу, групи КНТ-130
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ОКОПНИЙ В.Р.

(ПРИЗВИЩЕ та ініціали)

Керівник СУББОТІН С.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент ШИТКОВА О.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ОКОПНОГО Віталія Романовича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення обміну повідомленнями. Software for Instant Messaging

керівник проєкту (роботи) д.т.н., професор, СУББОТІН Сергій Олександрович,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 24 ” квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	СУББОТІН С.О., завідувач кафедри		
Нормоконтроль	ЛИПОВЕЦЬ М.В., асистент		

7. Дата видачі завдання “ 15 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис)

Віталій ОКОПНИЙ
(Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Сергій СУББОТІН
(Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
95 с., 3 табл., 34 рис., 3 дод., 18 джерел.

С#, JETBRAINS, МОВА ПРОГРАМУВАННЯ, ПРОГРАМНЕ
ЗАБЕЗПЕЧЕННЯ, СЕРЕДОВИЩЕ РОЗРОБКИ, ТЕКСТОВЕ
ПОВІДОМЛЕННЯ.

Об'єкт дослідження – процес розробки програмного забезпечення для обміну текстовими повідомленнями.

Предмет дослідження – програмні засоби для підтримки процесу обміну повідомленнями.

Мета роботи – розробка програмного забезпечення для обміну повідомленнями.

Матеріали, методи та технічні засоби: мова програмування С#, середовище розробки JetBrains.

Результати. Розроблено програмне забезпечення для підтримки процесу обміну повідомленнями за допомогою мови програмування С# та середовища розробки JetBrains.

Висновки. Виконано проєктування програмного забезпечення для підтримки процесу обміну повідомленнями. Розроблено програмне забезпечення для підтримки процесу обміну повідомленнями. Здійснено тестування розробленого програмного забезпечення для підтримки процесу обміну повідомленнями.

Галузь використання – інформаційно-комунікаційні технології.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 95 pages, 3 tables, 34 figures, 3 appendixes, 18 sources.

C#, JETBRAINS, PROGRAMMING LANGUAGE, SOFTWARE, DEVELOPMENT ENVIRONMENT, TEXT MESSAGING.

The object of research is the process of developing software for exchanging of text messages.

The subject of the research is software for supporting the messaging process.

The purpose of this work is to develop the software to support the text messaging process.

Materials, methods and technical tools: C# programming language, JetBrains development environment.

Results. The software tools to support the text messaging process using the C# programming language and the JetBrains development environment have been developed.

Conclusions. The software to support the text messaging process has been designed. The software to support the text messaging process has been developed. The developed software to support the text messaging process has been tested.

Scope of use – information and communication technologies.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	11
1.1 Програмне забезпечення для обміну текстовими повідомленнями	11
1.2 Аналіз програмного забезпечення для підтримки процесу обміну текстовими повідомленнями	25
1.3 Висновки за розділом 1	34
2 Матеріали і методи	36
2.1 Вибір мови програмування	36
2.2 Вибір середовища розробки для створення програмного забезпечення для обміну повідомленнями	38
2.3 Висновки за розділом 2	41
3 Опис програми	42
3.1 Структура програмного забезпечення для обміну повідомленнями	42
3.2 Функціонування програмного забезпечення для обміну повідомленнями	44
3.3 Розробка бази даних програмного забезпечення для обміну повідомленнями	47
3.4 Проєктування інтерфейсу програмного забезпечення для обміну повідомленнями	49
3.5 Висновки за розділом 3	52
4 Експлуатація, тестування та експериментальне дослідження програми	53
4.1 Призначення й умови застосування програми	53
4.2 Характеристики програми для підтримки процесу обміну повідомленнями	54
4.3 Інструкція по експлуатації програми	55
4.3.1 Звернення до програми	55
4.3.2 Вхідні й вихідні дані	55

4.3.3 Повідомлення.....	55
4.4 Виконання програмного забезпечення для підтримки процесу обміну повідомленнями	56
4.5 Тестування програмного забезпечення для підтримки процесу обміну повідомленнями	59
4.6 Висновки за розділом 4	60
Висновки	61
Перелік джерел посилань	64
Додаток А Технічне завдання	66
Додаток Б Фрагмент тексту програми	70
Додаток В Слайди презентації	88

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ІМ – Instant messaging;

VOIP – Voice over IP;

MVC – Model-View-Controller;

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Актуальність використання програмного забезпечення обміну текстовими повідомленнями залишається високою у сучасному світі, оскільки такі програмні засоби дозволяють користувачам зручно та швидко обмінюватися інформацією, порівняно з телефонними дзвінками або електронною поштою [1]-[7].

Сучасне програмне забезпечення обміну текстовими повідомленнями пропонує широкий спектр функцій, таких як групові чати, відеодзвінки, обмін мультимедійними файлами та інші. Програмне забезпечення обміну текстовими повідомленнями також може інтегруватися з іншими програмними сервісами, такими як платіжні системи, соціальні мережі, календарі, електронна пошта та інші [1]-[7].

Великий попит на програмне забезпечення обміну текстовими повідомленнями також обґрунтовується поширенням мобільних пристроїв. У зв'язку зі зростанням рівня популярності, а в деяких випадках, і необхідності дистанційного навчання та роботи з віддаленими командами, програмне забезпечення обміну текстовими повідомленнями стає важливим інструментом для спілкування, співпраці та координації робочих та інших процесів [1]-[7].

Загалом, з урахуванням швидкого розвитку технологій та зміни способу спілкування людей, програмне забезпечення обміну текстовими повідомленнями залишається актуальним і важливим елементом сучасного цифрового життя [1]-[7].

Проте деяке програмне забезпечення обміну текстовими повідомленнями може використовувати ненадійні засоби захисту особистих даних користувачів, зокрема текстових повідомлень, що може призвести до витоку конфіденційної інформації. Інші програми обміну повідомленнями можуть мати унікальні функції, які можуть бути зайвими та ускладнювати спілкування між користувачами. Крім того, існує проблема стандартизації

форматів текстових повідомлень, що може призвести до проблеми комунікації між користувачами, які користуються апаратними засобами, що підтримуються різними програмними платформами [1]-[7].

Тому актуальною є розробка програмного забезпечення для обміну текстовими повідомленнями. У дипломній кваліфікаційній роботі бакалавра розв'язується актуальне завдання розробки програмного забезпечення обміну повідомленнями.

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для обміну повідомленнями;
- здійснити проєктування програмного забезпечення обміну повідомленнями;
- створити програмне забезпечення для обміну повідомленнями;
- виконати тестування розробленого програмного забезпечення для обміну повідомленнями.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Програмне забезпечення для обміну текстовими повідомленнями

Обмін текстовими повідомленнями представляє собою сервіс, що дозволяє людям спілкуватися в реальному часі за допомогою спеціального програмного забезпечення. Відмінною особливістю обміну текстовими повідомленнями від електронної пошти є те, що миттєві повідомлення дають змогу переглядати повідомлення однієї особи на екрані іншої особи майже миттєво після відправлення відповідного повідомлення [1], [2].

Миттєвий обмін повідомленнями (Instant messaging, IM) являє собою текстовий діалог між двома або більше людьми на комп'ютері або мобільному пристрої через чат, що має доступ до мережі Інтернет. Миттєвий обмін повідомленнями може відбуватися у формі чату, де проходять публічні дискусії в режимі реального часу, де кожен у каналі бачить усе, що говорять інші, також можливий варіант обміну миттєвими повідомленнями в особистому режимі [1], [2].

У своїй базовій формі мета обміну миттєвими повідомленнями полягає у двох основних завданнях: відстеженні присутності для надсилання повідомлень в залежності від того, хто наявний у чаті, і обміні текстовими повідомленнями [1], [2].

Програмне забезпечення підтримки процесу обміну миттєвими повідомленнями спирається на центральний сервер або сервери для відстеження присутності користувачів. Коли користувач увійшов до програмної системи обміну миттєвими повідомленнями, його ідентифікаційне ім'я розпізнається, й інші користувачі, які також перебувають в мережі та мають його у своєму списку друзів або контактів, можуть отримати сповіщення про його присутність. Потім програмне забезпечення підтримки процесу обміну миттєвими повідомленнями налагоджує пряме з'єднання між користувачами, щоб вони могли спілкуватися один з одним в реальному часі.

Програмне забезпечення обміну миттєвими повідомленнями набуло широкої популярності завдяки постійним змаганням між комерційними компаніями, які розробляють ці програми [1], [2]. Раніше обмін миттєвими повідомленнями використовувався переважно для спілкування користувачів між собою чи з друзями через Інтернет. Проте з розвитком Інтернет технологій процес обміну миттєвими повідомленнями та відповідне програмне забезпечення стали невід'ємною частиною бізнесу різного спрямування [1], [2].

У наш час обмін миттєвими повідомленнями є невід'ємною частиною програмного забезпечення різного призначення. Наприклад, соціальна мережа Facebook розширила свої можливості, надавши користувачам функцію обміну миттєвими повідомленнями [1], [2].

У сфері бізнесу миттєві повідомлення відіграють важливу роль у підтримці зв'язку в реальному часі [1], [2]. Бізнес завжди вимагає спілкування. Менеджерам потрібно обмінюватися ідеями, керівництво має бути на зв'язку зі співробітниками. Програмне забезпечення підтримки процесу обміну миттєвими повідомленнями забезпечує зручний та ефективний спосіб комунікації. Інформаційні технології та програмні засоби миттєвого обміну повідомленнями постійно розвиваються, щоб додати більше корисних можливостей та відповідних функцій для бізнесу та полегшити комунікацію між користувачами [1], [2].

Миттєві повідомлення можуть бути подані як текстові, так і звукові або відео повідомлення. Голосова передача даних через Інтернет (Voice over IP, VOIP) є технологією, що дозволяє користувачам спілкуватися один з одним за допомогою миттєвих повідомлень так само, як по телефону, за винятком того, що вона передає голос через Інтернет, а не через радіохвилі, як мобільні телефони [1], [2].

Групові чати за допомогою сучасного програмного забезпечення для обміну миттєвими повідомленнями дозволяють трьом або більше людям спілкуватися в одній чат кімнаті, що дає змогу проводити групові розмови та групове спілкування [1], [2].

Останнім часом миттєві повідомлення є способом дистанційного спілкування, що дозволяє користувачам обмінюватися текстовими або іншими повідомленнями у реальному часі через Інтернет. Миттєві повідомлення є неодмінною складовою щоденного життя та впливають на спосіб взаємодії з іншими людьми. Розуміння суті та важливості процесу миттєвого обміну повідомленнями має вирішальне значення для того, щоб ефективно користуватися цими програмними сервісами та розробляти відповідне програмне забезпечення [3], [4].

Миттєві повідомлення є способом обміну текстовими повідомленнями, зазвичай за допомогою спеціальних програм або онлайн платформ. Важливість миттєвих повідомлень полягає у можливості швидко та легко спілкуватися у реальному часі, незалежно від відстані між співрозмовниками. Миттєві повідомлення дозволяють людям залишатися на зв'язку, навіть якщо вони знаходяться далеко один від одного [3], [4].

Програмні засоби підтримки процесу обміну миттєвими повідомленнями змінили спосіб спілкування користувачів. Завдяки миттєвим повідомленням користувачі можуть спілкуватися у режимі реального часу, незалежно від їх місця знаходження [3], [4].

Програмні засоби обміну миттєвими повідомленнями дають можливість не лише спілкуватися через текстові повідомлення, але і обмінюватися різноманітними мультимедійними контентом. Користувачі можуть легко ділитися фотографіями, відео та документами, щоб розширити обговорення. Це особливо корисно в професійному оточенні для обміну файлами та спільної роботи над важливими проектами [3], [4].

Сучасне програмне забезпечення для обміну миттєвими повідомленнями дозволяє спілкуватися один з одним через відео в реальному часі. Програмне забезпечення підтримки відео конференцій є розширенням відео дзвінків, що дозволяє трьом або більше людям брати участь у відео дзвінках та проводити зустрічі або навчальні сесії в режимі реального часу, навіть якщо учасники знаходяться в різних місцях [1], [2].

У сучасній цифровій епосі віддалене наймання стає все більш популярним вибором для компаній, які шукають співробітників. Роботодавці все менше обмежуються пошуком кадрів на локальному рівні і, натомість, можуть звертатися до фахівців з інших країн. Раніше це часто не вважалося найкращим варіантом, оскільки спілкування з іноземними працівниками було складним і дорогим. Проте завдяки програмним засобам обміну миттєвими повідомленнями комунікація між роботодавцями та працівниками з-за кордону стала більш ефективною [1], [2]. Швидка взаємодія через програмні засоби обміну повідомленнями є ключовою особливістю сучасного бізнес-середовища, особливо в сфері аутсорсингу, що стрімко розвивається [1], [2].

Деякі програмні засоби миттєвого обміну повідомленнями перейшли до інтеграції з мережами за допомогою протоколу HTML, що зроблено з метою обійти певні обмеження, які можуть бути налаштовані для протоколу миттєвого обміну повідомленнями у корпоративних мережах. Співробітники, які хочуть продовжувати використовувати інструменти миттєвого обміну повідомленнями для особистих і робочих цілей, шукають способи обходу цих обмежень. Важливо врахувати, що фільтрація HTML-контенту може бути виконана, особливо на мережевих шлюзах. Якщо політика вашої компанії забороняє використання миттєвого обміну повідомленнями, переконайтеся, що список заборонених IP-адрес чутливий до обміну миттєвими повідомленнями. Багато програмних засобів миттєвого обміну повідомленнями активно використовують доступні порти для транзитних з'єднань. Іншими словами, програма миттєвого обміну повідомленнями буде шукати цей порт, якщо це дозволяють правила комунікації [2], [3].

Програмні засоби миттєвого обміну повідомленнями забезпечують прямий та легкий спосіб спілкування користувачів. Відзначимо, що поруч із зручністю миттєвого обміну даними існують ризики. Використання програмних засобів миттєвих повідомлень може призвести до передачі інформації у відкритому вигляді, поширення Інтернет посилань в повідомленнях, передачі файлів або соціальної інженерії [2], [3].

Інформація, яка передається через миттєві повідомлення за допомогою відповідного програмного забезпечення, зазвичай подається у формі звичайного тексту. Існують програми, які можуть зашифрувати мережевий потік миттєвих повідомлень, але це часто потребує додаткових програмних компонентів. Коли користувач надсилає конфіденційну інформацію через миттєві повідомлення, ця інформація стає вразливою для перехоплення у мережі [2], [3].

При експлуатації програмного забезпечення обміну миттєвими повідомленнями важливо бути обережним у користуванні системою миттєвих повідомлень, оскільки ніколи не відомо, з ким саме відбувається спілкування. Ці програмні системи дозволяють додавати у повідомлення посилання на сторінки в Інтернет, що може бути небезпечним. Посилання можуть вести на шкідливі вебсайти. Багато поштових клієнтів можуть перевіряти повідомлення на наявність шкідливих посилань, але програмні засоби підтримки процесу обміну миттєвими повідомленнями цього зазвичай не роблять. Хоча деякі клієнти миттєвих повідомлень можуть сповіщати користувачів про потенційно небезпечні посилання, вони не можуть блокувати їх [2], [3].

Також важливо враховувати, що програмні засоби обміну миттєвими повідомленнями дозволяють передавати файли. Часто це є просто зображення або інші файли, але іноді ці файли можуть містити шкідливі програми. Програмне забезпечення підтримки спілкування за допомогою миттєвих повідомлень за замовчуванням не перевіряють ці файли на наявність вірусів. Тому важливо це враховувати та обирати файлову передачу з подальшою перевіркою отриманих файлів [2], [3].

Атаки на основі програмних засобів соціальної інженерії дуже поширені в програмних системах обміну миттєвими повідомленнями через їх неформальний характер. Важливо розуміти, що неможливо завжди перевірити ідентичність особи, з якою відбувається обмін повідомленнями. Тому важливо бути обережним при обміні конфіденційною інформацією через програмні засоби обміну миттєвими повідомленнями [2], [3].

З метою захисту процесу спілкування через програмні засоби обміну миттєвими повідомленнями, можна заборонити використання зовнішніх клієнтів, які не потребують входу в центральну систему ідентифікації користувачів. Якщо потрібна програмна система обміну миттєвими повідомленнями для виробничої або іншої організації, варто розглянути можливість створення внутрішньої програмної системи обміну миттєвими повідомленнями. Якщо зовнішні програмні системи не можна обійти, слід звернути увагу на захист користувачів та їхніх програмних систем. Можна використовувати програми для шифрування вмісту миттєвих повідомлень, що підвищить безпеку спілкування. Також важливо налаштувати перевірку безпеки при передачі файлів через програмні засоби підтримки процесу обміну миттєвими повідомленнями, використовуючи антивірусні програми. Не менш важливою є інструктаж кінцевих користувачів щодо уникання небезпечних дій під час використання миттєвих повідомлень, що допоможе запобігти атакам соціальної інженерії [2], [3].

Ще одним важливим елементом програмних засобів обміну миттєвими повідомленнями є можливість створення списків контактів. Користувачі програмних засобів обміну миттєвими повідомленнями можуть формувати мережі контактів, які дозволяють їм легко знаходити осіб, з якими вони хочуть спілкуватися, і вести з ними комунікацію. Ця можливість допомагає людям будувати міцні взаємини та спілкуватися з друзями, колегами та партнерами по бізнесу [3], [4].

Програмні платформи обміну миттєвими повідомленнями мають ряд функцій, які поліпшують спілкування користувачів, зокрема, функції миттєвого обміну текстовими повідомленнями, обміну медіа файлами, підтримка списків контактів та оновлення статусу. Багато програмних платформ пропонують голосові та відео дзвінки, груповий чат і передачу файлів. Ці можливості постійно розвиваються, щоб відповідати змінним потребам користувачів [3], [4].

Передача повідомлень у реальному часі становить один з основних етапів процесу обміну миттєвими повідомленнями. Повідомлення надсилаються та приймаються миттєво, що дозволяє проводити гнучкі та динамічні розмови. Ця можливість не потребує очікування відповіді та забезпечує ефективне та безперебійне спілкування [3], [4].

Обмін мультимедійними файлами є ще одним значущим аспектом програмного забезпечення підтримки процесу обміну миттєвими повідомленнями. Користувачі можуть легко ділитися фотографіями, відео та іншими мультимедійними матеріалами зі своїми контактами для збагачення спілкування. Ця функція стає все більш важливою в сучасному візуальному світі, де обмін моментами та враженнями через фотографії стає все більш популярним [3], [4].

Списки контактів грають ключову роль на програмних платформах обміну миттєвими повідомленнями. Користувачі можуть створювати та керувати цими списками, що спрощує пошук та зв'язок з людьми, з якими вони хочуть спілкуватися. Ця функція дозволяє людям залишатися на зв'язку та спілкуватися з друзями, родиною та колегами [3], [4].

Оновлення статусу дозволяє користувачам програмного забезпечення підтримки процесу обміну повідомленнями повідомляти інших користувачів про свій стан доступності та поточні зайнятості. За допомогою цієї функції користувачі можуть вказати, чи можуть вони відповідати на повідомлення чи вони зайняті. Це сприяє управлінню очікуваннями та підтримує ефективне та своєчасне спілкування [3], [4].

З часом програмні платформи обміну миттєвими повідомленнями розширили свій функціонал на голосові та відео дзвінки. Ці можливості дозволяють здійснювати розмови у реальному часі за допомогою голосу та відео. Групові чати стають все більш популярними, дозволяючи багатьом учасникам приєднатися до однієї розмови, що сприяє співпраці та командній роботі [3], [4].

Крім того, передача файлів стала необхідною складовою програмних платформ обміну миттєвими повідомленнями. Користувачі можуть легко обмінюватися різноманітними файлами, такими як документи, презентації та інші важливі документи. Ця можливість особливо корисна у бізнес-середовищі, де ефективний обмін файлами має велике значення для продуктивності [3], [4].

Програмні засоби підтримки обміну миттєвими повідомленнями стали потужним засобом комунікації, що дає можливість людям спілкуватися легко та швидко без обмежень часу та відстані. Завдяки своїм можливостям і функціям, програмні засоби підтримки процесу обміну миттєвими повідомленнями змінили спосіб взаємодії між користувачами і стали неодмінною складовою їх щоденного життя [4], [5].

Концепція обміну миттєвими повідомленнями виникла на початкових етапах онлайн спілкування. У той час, коли Інтернет стимулював розвиток сучасних засобів комунікації, було розроблено базові програмні платформи обміну миттєвими повідомленнями, що заклали основу для їх широкого подальшого використання [4], [5].

Разом із розвитком Інтернету продовжує розвиватися й індустрія програмного забезпечення підтримки процесу обміну миттєвими повідомленнями. Розробляються потужні та легкі у використанні платформи, що перетворюють процес обміну миттєвими повідомленнями на загальноприйнятий програмний інструмент і сприяють експоненціальному зростанню та розвитку відповідного програмного забезпечення [4], [5].

Прогрес інформаційних технологій відіграв критичну роль у формуванні майбутнього програмного забезпечення для обміну миттєвими повідомленнями. З появою мобільних телефонів та мобільних застосунків, тепер можливо спілкуватися миттєво навіть у дорозі. Такі функції як показ часу читання повідомлення, голосові повідомлення та спеціальне шифрування поліпшують зручність для користувача та гарантують конфіденційність та безпеку спілкування [4], [5].

Широке застосування програмних засобів для підтримки процесу обміну миттєвими повідомленнями має значний вплив на особисте та професійне спілкування. Таке програмне забезпечення змінює спосіб взаємодії між користувачами, ліквідує бар'єри та перетворює динаміку взаємин [4], [5].

Завдяки миттєвим повідомленням за допомогою відповідного програмного забезпечення залишатися на зв'язку з близькими стало простіше, ніж будь-коли, незалежно від географічної відстані. Чи то текстові повідомлення, обмін фотографіями та відео, або проведення аудіо чи відео дзвінків, обмін миттєвими повідомленнями став важливим інструментом для підтримки взаємин [4], [5].

Промислові та інші підприємства використовують програмні засоби для обміну миттєвими повідомленнями як ефективний засіб спілкування. Команди можуть співпрацювати, обмінюватися ідеями та швидко приймати рішення в реальному часі. Платформи обміну миттєвими повідомленнями спрощують робочий процес, підвищують продуктивність і забезпечують ефективну командну роботу [4], [5]. З розвитком процесу обміну миттєвими повідомленнями з'являються нові функції та технології, які формують його майбутнє. Розуміння цих нових тенденцій і прогнозів може надати цінну інформацію для розробки відповідного програмного забезпечення [4], [5].

Прогнозується, що штучний інтелект і чат-боти гратимуть важливу роль у майбутньому обміні миттєвими повідомленнями. Ці технології можуть поліпшити роботу користувача шляхом персоналізованих пропозицій, автовідповідачів та інтелектуальних помічників на платформі обміну повідомленнями [4], [5]. У майбутньому забезпечення процесу обміну миттєвими повідомленнями може надати нові можливості, прогрес у сфері доповненої реальності та віртуальної реальності зробить обмін повідомленнями більш інтерактивним. Крім того, його можливості розширяться завдяки інтеграції з іншими технологіями, такими як Інтернет речей [4], [5].

Розвиток програмних засобів для обміну миттєвими повідомленнями значно змінив спосіб, яким спілкуються користувачі, усуваючи географічні обмеження і розширюючи можливості комунікації на світовому рівні. Від базових програмних рішень до новітніх технологій та тенденцій, обмін миттєвими повідомленнями продовжує формувати цифровий ландшафт сучасних інформаційних технологій та програмних засобів [4], [5].

Миттєві повідомлення, Інтернет чати та однорангові мережі дозволяють забезпечити необмежений та неперервний обмін інформацією та файлами. Програмні засоби для обміну миттєвими повідомленнями мають багатогранний вплив на бізнес, включаючи безпеку користувачів, крадіжку конфіденційних даних і порушення авторських прав. Миттєві повідомлення стали ефективним засобом спілкування. Не кожен клієнт обміну повідомленнями або відповідне програмне забезпечення підтримує всі функції обміну даними, але вони мають основні можливості, які додають або поліпшують інші параметри комунікації, такі як передача файлів, відео та аудіо даних [4], [5].

На відміну від програмних засобів для миттєвого обміну повідомленнями, використання засобів електронної пошти в деяких випадках є досить повільним, крім того, користувач не може знати, чи його адресати онлайн в даний момент та чи прочитано його лист. Іноді для надсилання або отримання електронних листів потрібно пройти кілька етапів. Тому обмін миттєвими повідомленнями за допомогою відповідних програмних засобів набув такої великої популярності. Програмні засоби для обміну миттєвими повідомленнями дозволяють створювати список контактів. Користувач може спілкуватися з будь-ким зі списку контактів, якщо у даний момент обидва учасники спілкування знаходяться у мережі [5], [6].

Проаналізуємо функції та можливості програмних засобів підтримки процесу обміну миттєвими повідомленнями [5], [6]:

- миттєвий обмін повідомленнями – надсилання та отримання повідомлень у реальному часі користувачам, що знаходяться у мережі;

- груповий чат – можливість спілкування з декількома друзями або колегами одночасно;
- можливість відправлення, перегляду та обговорення відео з учасниками спілкування;
- можливість відправлення, перегляду та обговорення фотографій з учасниками спілкування;
- можливість відтворення звукових ефектів;
- обмін файлами;
- мобільні функції – можливість відправлення повідомлень та файлів з мобільного пристрою [5], [6].

Онлайн сервіси створюють середовище, де користувачі можуть взаємодіяти з програмним сервісом через фізичний інтерфейс. Розробники програмного забезпечення створюють програмне забезпечення для чат-кімнат та розміщують їх на вебсервері. Чати дозволяють кільком користувачам обмінюватися повідомленнями, які бачать інші користувачі конкретної кімнати. Обмін миттєвими повідомленнями, по суті, є приватною чат-кімнатою для двох або більше осіб [5], [6].

Проаналізуємо етапи функціонування програмного забезпечення для обміну миттєвими повідомленнями. На початковому етапі користувач завантажує копію програмного клієнта для обміну повідомлення на свій апаратний засіб, потім встановлює програмне забезпечення для обміну повідомлення та запускає відповідний програмний клієнт. Програмний клієнт для обміну миттєвими повідомленнями намагається підключитися до сервера, використовуючи власні протоколи зв'язку. Після успішного підключення програмного клієнта обміну миттєвими повідомленнями до сервера користувач вводить своє ім'я користувача та пароль, щоб увійти на сервер. Сервер перевіряє дані користувача та надає йому доступ до програмної системи підтримки процесу обміну повідомленнями [5], [6].

Клієнт програмного забезпечення підтримки процесу обміну повідомленнями надсилає на сервер інформацію про підключення (IP-адресу

та номер порту, призначений клієнту) комп'ютера, а також надає імена всіх контактів з вашого списку. Сервер створює тимчасовий файл з інформацією про підключення та список контактів. Він також перевіряє, чи є хтось з списку контактів користувача у мережі [5], [6].

Якщо сервер програмного забезпечення підтримки процесу обміну повідомленнями знаходить контакт користувача, з яким він хоче спілкуватися, він надсилає повідомлення цьому користувачеві на комп'ютер з інформацією про його підключення. Також сервер повідомляє про підключення всім контактам зі списку користувача [5], [6].

Коли клієнт програмного забезпечення підтримки процесу обміну повідомленнями отримує інформацію про підключення іншого користувача зі списку контактів, статус цієї особи змінюється на Онлайн. Натискання на ім'я цієї людини у списку контактів відкриває вікно для введення тексту, після цього можна спілкуватися з цим користувачем, написавши відповідне повідомлення [5], [6]. Оскільки клієнт має IP-адресу та номер порту комп'ютера, на який надсилаються повідомлення, тому повідомлення надсилається безпосередньо до цього клієнта на його комп'ютер. Це означає, що немає сервера, через який пройшло би повідомлення, вся комунікація відбувається безпосередньо між обома клієнтами [5], [6]. Інший користувач отримує миттєве повідомлення та відповідає. Після закінчення розмови вікно повідомлень закривається, і відбувається перехід програмного забезпечення підтримки процесу обміну повідомленнями в автономний режим. Коли це відбувається, ваш клієнт програмного забезпечення підтримки процесу обміну повідомленнями надсилає повідомлення на сервер, щоб завершити сеанс. Сервер надсилає повідомлення клієнту програмного забезпечення підтримки процесу обміну повідомленнями кожної особи зі контактів, що зараз у мережі, щоб показати, що користувач вийшов з програмної системи обміну повідомленнями [5], [6].

Основні програмні інструменти миттєвого обміну повідомленнями використовують власні протоколи, які є унікальними для кожної платформи.

Це перешкоджає користувачам однієї програмної платформи обміну повідомленнями спілкуватися з тими, хто використовує іншу. Важливо зауважити, що миттєвий обмін повідомленнями не є повністю безпечним засобом спілкування. Повідомлення та інформація про підключення зберігаються на серверах, керованих провайдерами миттєвого обміну повідомленнями. Більшість з них забезпечують певний рівень шифрування, але це недостатньо безпечно для вимог щодо конфіденційності. Існує загроза перехоплення облікових записів користувачів миттєвих повідомлень і їх використання різними зловмисниками, з зараженими вірусами файлами, які відправляють миттєві повідомлення. Протокол передачі голосу через Інтернет вважається більш вразливим до атак, ніж текстові миттєві повідомлення та відповідне програмне забезпечення [5], [6].

Протягом останніх років можливості програмних засобів підтримки процесу миттєвого обміну повідомленнями значно розширилися. Більшість програмних платформ для обміну файлами включають сервіси миттєвого обміну повідомленнями, які інтегруються в електронну пошту, щоб користувачі могли отримувати сповіщення через електронну пошту про надходження миттєвих повідомлень. Багато постачальників послуг також пропонують доступ до своїх програмних сервісів за допомогою чатів та обміну файлами на мобільних телефонах або інших портативних пристроях. Можливості налаштування стали більш широкими, оскільки користувачі програмних засобів підтримки процесу миттєвого обміну можуть налаштовувати свої зображення профілю, інтерфейс екрана та інші параметри [5], [6].

Більшість програм для підтримки процесу миттєвого обміну повідомленнями дозволяють спілкуватися так само, як у телефонних дзвінках, а деякі пропонують можливості спілкування за допомогою відео чату [6].

Універсальні постачальники послуг програмного забезпечення для обміну миттєвими повідомленнями функціонують у мережі, тому не потрібно завантажувати додаткове програмне забезпечення. Крім того деякі програмні засоби пропонують мобільні версії. Розвиток програмних засобів для обміну

миттєвими повідомленнями виглядає перспективно. Велика кількість програмних інструментів для обміну миттєвими повідомленнями продовжують оновлюватися відповідними власниками, а постачальники послуг миттєвих повідомлень працюють над створенням більш інтегрованих інтерфейсів між користувачами. Бізнес-користувачі можуть легко проводити віртуальні зустрічі та співпрацювати над проектами [5], [6].

Визначено, що миттєвий обмін повідомленнями являє собою текстовий діалог між двома або більше людьми на комп'ютері або мобільному пристрої через чат, що має доступ до мережі Інтернет. Програмне забезпечення підтримки процесу обміну миттєвими повідомленнями спирається на центральний сервер або сервери для відстеження присутності користувачів та надання можливості прямого та зручного способу спілкування без обмежень часу та відстані. Відзначено, що розуміння суті та важливості процесу миттєвого обміну повідомленнями має вирішальне значення для того, щоб ефективно користуватися цими програмними сервісами та розробляти відповідне програмне забезпечення. Деякі програмні засоби миттєвого обміну повідомленнями перейшли до інтеграції з мережевими протоколами, що зроблено з метою обійти певні обмеження, які можуть бути налаштовані для протоколу миттєвого обміну повідомленнями у корпоративних мережах. Відзначено, що поруч із зручністю миттєвого обміну даними існують ризики, використання програмних засобів підтримки процесу обміну миттєвими повідомленнями може призвести до передачі інформації у відкритому вигляді, поширення Інтернет посилань в повідомленнях, передачі файлів стороннім особам. Якщо потрібна програмна система обміну миттєвими повідомленнями для виробничої або іншої організації, варто розглянути можливість створення внутрішньої програмної системи обміну миттєвими повідомленнями. Проаналізовано функції та можливості програмних засобів підтримки процесу обміну миттєвими повідомленнями. Проаналізовано етапи функціонування програмного забезпечення для обміну миттєвими повідомленнями.

1.2 Аналіз програмного забезпечення для підтримки процесу обміну текстовими повідомленнями

Проаналізуємо програмне забезпечення для підтримки процесу обміну текстовими повідомленнями [7]-[13].

Програмні платформи обміну миттєвими повідомленнями використовуються для різних цілей, таких як обслуговування клієнтів, підтримка співробітників та комунікація всередині команди. Функціональність програмного забезпечення для обміну миттєвими повідомленнями може варіюватися. Деякі програми зосереджені на командному спілкуванні, тоді як інші можуть бути багатоканальними платформами обміну повідомленнями, що поєднують внутрішні та зовнішні канали зв'язку [7]-[10].

Часто компанії інтегрують програмне забезпечення для обміну миттєвими повідомленнями зі своїми вебсайтами за допомогою плагінів, щоб забезпечити підтримку клієнтів. Це дозволяє відвідувачам вебсайту спілкуватися безпосередньо з командою обслуговування клієнтів через вікно чату [7]-[10].

Програмне забезпечення Zendesk для підтримки процесу обміну миттєвими повідомленнями дозволяє компаніям надавати зручну багатоканальну підтримку співробітникам та клієнтам. Програмне забезпечення Zendesk дозволяє користувачам надсилати текстові та голосові повідомлення, підтримує функції чату в реальному часі, а також чат-боти на платформі Facebook та інші можливості [8], [11].

Програмне забезпечення Zendesk (рис. 1.1) автоматично направляє запити клієнтів до найкращого доступного агента та надає автоматичні відповіді, що допомагає агентам ефективно відповідати на повторювані запитання та економити час. Крім того, Zendesk проводить опитування задоволеності, які допомагають отримати відгуки, які можна використовувати для поліпшення клієнтського та співробітницького досвіду [8], [11].

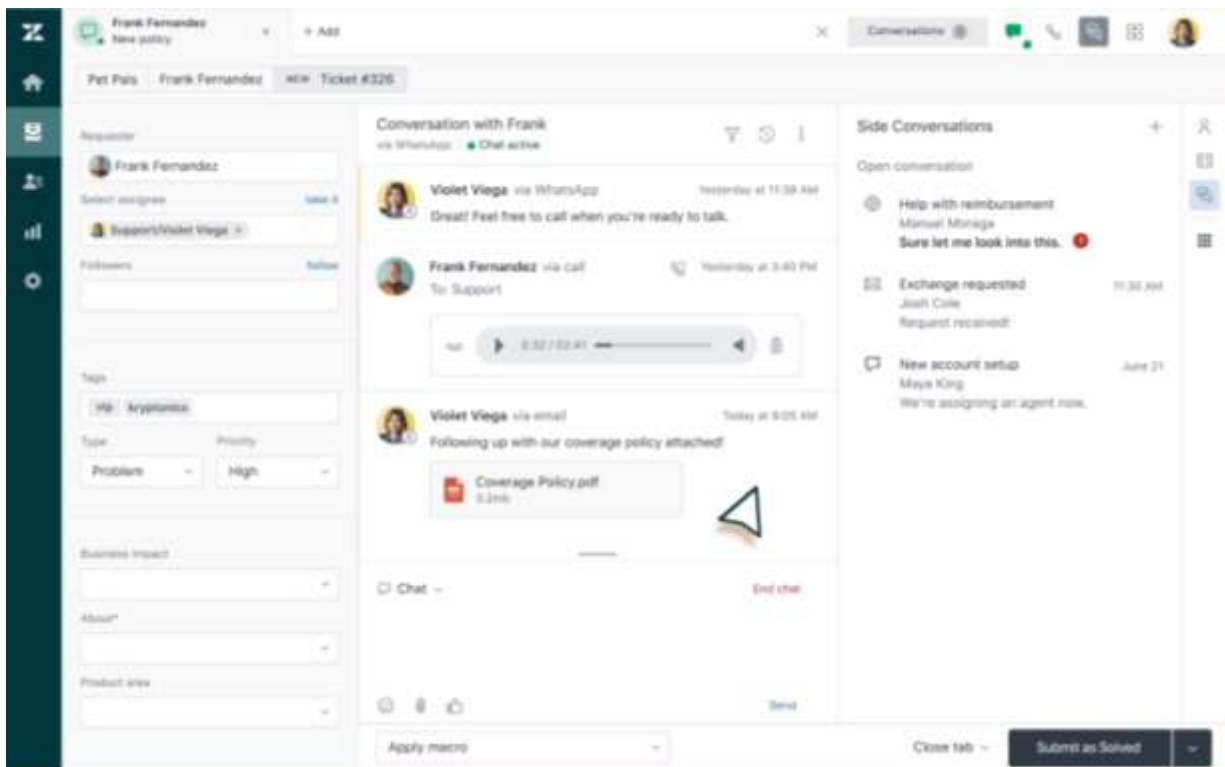


Рисунок 1.1 – Програмне забезпечення Zendesk [11]

Вбудована система обміну повідомленнями програмного забезпечення Zendesk дозволяє надавати підтримку клієнтам і співробітникам прямо з вебсайту, починаючи з відповідей на запитання та закінчуючи надсиланням активних повідомлень. Платформа підтримки чат ботів програмного забезпечення Zendesk, заснована на штучному інтелекті, дозволяє компаніям надавати цілодобову підтримку та відповідати на запитання клієнтів у природній мові [8], [11].

Крім того, програмне забезпечення Zendesk для взаємодії з працівниками та клієнтами починає працювати миттєво, завдяки чому Zendesk може легко масштабуватися для використання компаніями та організаціями будь-якого розміру [8], [11].

Основними характеристиками програмного забезпечення Zendesk є такі [8], [11]:

– багатоканальна маршрутизація: програмне забезпечення Zendesk дозволяє обробляти запити користувачів з різних каналів спілкування, таких як електронна пошта, чат, телефон, соціальні мережі тощо [8], [11];

– робоча область агента: програмне забезпечення Zendesk надає спеціальну робочу область для агентів з можливістю керування всіма клієнтськими запитами [8], [11];

– засоби сповіщення: агенти та адміністратори програмного забезпечення Zendesk отримують сповіщення про нові запити, оновлення та інші важливі події [8], [11];

– автоматичні відповіді: програмне забезпечення Zendesk дозволяє створювати передбачувані відповіді на типові запитання клієнтів або автоматизовані процеси обробки [8], [11];

– вбудований обмін повідомленнями: програмне забезпечення Zendesk підтримує можливість спілкування з клієнтами в режимі реального часу через чат, що полегшує швидке вирішення проблем [8], [11];

– опитування користувачів: програмне забезпечення Zendesk підтримує процес збору зворотного зв'язку від клієнтів для оцінки рівня їхнього задоволення обслуговуванням [8], [11];

– чат-боти на основі штучного інтелекту: програмне забезпечення Zendesk підтримує використання інтелектуальних чат-ботів для відповіді на типові запитання та автоматизації деяких операцій [8], [11];

– активний обмін повідомленнями: можливість запускати ініціативні спілкування з клієнтами для підвищення їхньої участі у процесі вирішення проблем [8], [11];

– інтеграція засобів спілкування за допомогою чату на вебсайт компанії для забезпечення швидкого доступу клієнтів до служби підтримки [8], [11].

Серед недоліків програмного забезпечення Zendesk можна відзначити складність налаштування. Для деяких користувачів програмного забезпечення Zendesk налаштування програми може бути складним процесом, особливо для тих, хто не має досвіду у використанні подібного програмного забезпечення. Іншим недоліком програмного забезпечення Zendesk є його вартість. Вартість Zendesk може бути високою для деяких компаній, особливо для малих підприємств, які мають обмежений бюджет на програмне забезпечення. Крім

того, хоча Zendesk пропонує багато функцій для підтримки клієнтів, іноді може бути обмежена в реалізації специфічних потреб певної компанії [8], [11].

Програмне забезпечення WeChat (рис. 1.2) являє собою універсальний засіб зв'язку, що працює на різних платформах. Програмне забезпечення WeChat підтримує особисті та групові чати та дозволяє відправляти голосові, фото, відео та текстові повідомлення через мобільні мережі. Користувач може користуватися WeChat на різних пристроях, включаючи веббраузери, що дає можливість зручно спілкуватися [8], [9], [12].



Рисунок 1.2 – Програмне забезпечення WeChat [12]

Програмне забезпечення WeChat є безкоштовним месенджером для мобільних пристроїв, комп'ютерів та вебкористувачів. Програмне забезпечення WeChat дозволяє спілкуватися текстовими та голосовими повідомленнями, а також за допомогою відеодзвінків. Крім того, програмне забезпечення WeChat доступно на різних мовах і має функцію мобільного платежу через цифровий гаманець [8], [9], [12].

Програмне забезпечення WeChat надає можливість підприємствам створювати свої облікові записи та спілкуватися з клієнтами через телефонні дзвінки або обмін миттєвими повідомленнями. Також компанії можуть налаштовувати автоматичні вітальні повідомлення та реагувати на телефонні дзвінки від клієнтів. У програмі WeChat доступні різноманітні можливості, такі як сигнал тривоги, автоматична відповідь, вбудований обмін повідомленнями, голосові та відеодзвінки, а також різні мобільні способи оплати. Крім того, WeChat підтримує переклад мови, дозволяє поширювати власний QR-код, створювати контент для пошуку в самій програмі, запускати рекламні кампанії та створювати програмні модулі, які можна продавати безпосередньо через власні електронні магазини [8], [9], [12].

Програмне забезпечення WeChat надає засоби пошуку контактів за допомогою WeChat ID або номера мобільного телефону. Засоби програми WeChat дозволяють користувачам поділитися улюбленими моментами зі свого життя з друзями, використовуючи фотографії та відео, а також дозволяють переглядати події друзів зі списку контактів у будь-який час [8], [9], [12]. Зручно спілкуйтеся з друзями та родиною за допомогою WeChat. Платформа дозволяє здійснювати безкоштовні відео та аудіо дзвінки з друзями або групами друзів з будь-якого місця [8], [9], [12].

Програмне забезпечення WeChat має такі особливості [8], [9], [12]:

- підтримка засобів миттєвого спілкування: програмне забезпечення WeChat дозволяє користувачам надсилати текстові повідомлення, голосові та відео виклики, а також миттєві фотографії та відео, щоб спілкуватися з друзями та родиною у реальному часі [8], [9], [12];

- пошук та додавання нових контактів: користувачі програмного забезпечення WeChat можуть швидко знаходити та додавати нових друзів за допомогою WeChat ID або номерів мобільних телефонів [8], [9], [12];

- моменти: функція Моменти дозволяє користувачам ділитися улюбленими моментами свого життя, такими як фотографії та відео, зі своїми друзями, а також переглядати моменти своїх друзів [8], [9], [12];

- безкоштовні виклики: програмне забезпечення WeChat надає можливість здійснювати безкоштовні відео та голосові виклики з друзями або групами друзів [8], [9], [12];

- підтримка різних платформ: програмне забезпечення WeChat доступно на різних платформах, включаючи мобільні пристрої, настільні комп'ютери, що робить його зручним для використання в будь-який час та в будь-якому місці [8], [9], [12];

- автоматичні вітальні повідомлення: користувачі програмного забезпечення WeChat можуть налаштовувати автоматичні вітальні повідомлення та телефонні дзвінки для зв'язку з клієнтами [8], [9], [12].

Серед недоліків програмного забезпечення WeChat можна відзначити складність налаштування. Для деяких користувачів програмного забезпечення WeChat налаштування програми може бути складним процесом, особливо для тих, хто не має досвіду у використанні подібного програмного забезпечення. Крім того, хоча WeChat пропонує багато функцій для підтримки клієнтів, іноді може бути обмежена в реалізації специфічних потреб певної компанії [8], [12].

Програмне забезпечення обміну повідомленнями Ryver дозволяє підтримувати процеси делегування та виконання внутрішніх бізнес-завдань. Користувачі програмного забезпечення обміну повідомленнями Ryver можуть створювати приватні групи, вести приватні розмови один на один і створювати форуми для конкретних аудиторій [7]–[9], [13].

Програмне забезпечення обміну повідомленнями Ryver є програмою для співпраці в команді з можливістю обміну повідомленнями. За допомогою Ryver користувачі можуть створювати канали чату для конкретних груп учасників команди, а також проводити приватні один на один розмови. Цей інструмент співпраці також підтримує голосові та відео дзвінки [7]–[9], [13].

Програмне забезпечення Ryver (рис. 1.3) дозволяє організовувати спілкування команди в єдиній програмі з груповим чатом, управлінням завданнями, а також аудіо та відео дзвінками [7]–[9], [13].

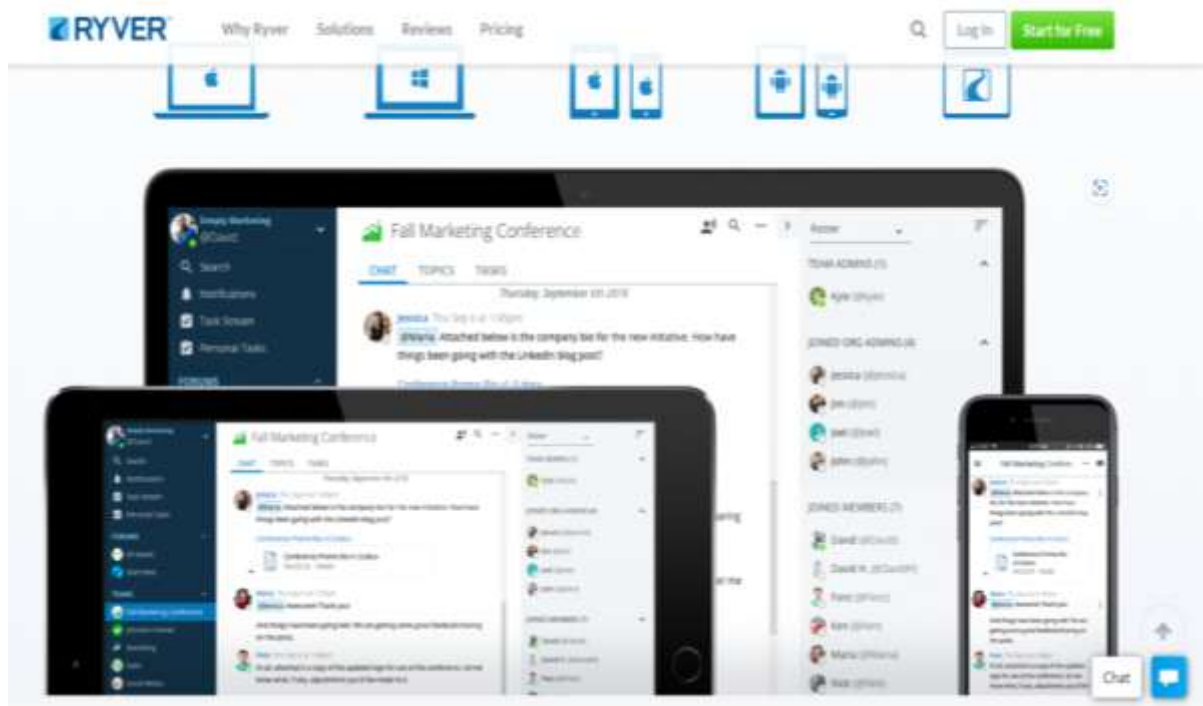


Рисунок 1.3 – Програмне забезпечення Ryver [13]

Користувачі програмного забезпечення Ryver можуть відстежувати як індивідуальні, так і групові завдання через панель завдань. Програмне забезпечення Ryver підтримує різноманітні інтеграції, включаючи можливість прямого обміну файлами з Google Drive або Dropbox [7]–[9], [13].

Особливості програмного забезпечення Ryver включають можливість проводити голосові та відеодзвінки, обмінюватися файлами, отримувати сигнали тривоги, взаємодіяти з робочим столом, спілкуватися з командою, проводити голосування, участь у відеоконференціях, спільний доступ до екрану та функція автоматичного сповіщення. Поміж функцій інтеграції з іншими програмним забезпеченням можна відзначити можливість роботи з Dropbox і Google Drive, а також можливість підключення додаткових інтеграцій через сервіс Zapier за додаткову плату [7]–[9], [13].

Програмне забезпечення Ryver має такі особливості [7]–[9], [13]:

- підтримка голосових та відео дзвінків: програмне забезпечення Ryver забезпечує можливість користувачів проводити голосові та відео конференції з членами команди [7]–[9], [13];

- підтримка можливості обміну файлами: програмне забезпечення Ryver дозволяє користувачам обмінюватися файлами, спрощуючи спільну роботу над проектами [7]–[9], [13];

- підтримка функцій оповіщення: програмне забезпечення Ryver надає користувачам можливість отримувати повідомлення-тривоги для важливих подій [7]–[9], [13];

- робочий стіл: програмне забезпечення Ryver надає зручний інтерфейс для організації робочих завдань та спільної роботи над ними [7]–[9], [13];

- засоби спілкування з командою: програмне забезпечення Ryver надає можливість створювати групові чати та приватні бесіди для ефективного спілкування з колегами [7]–[9], [13];

- засоби підтримки процесу голосування: програмне забезпечення Ryver надає функціонал для проведення голосувань серед учасників команди з метою прийняття рішень [7]–[9], [13];

- відеоконференції: програмне забезпечення Ryver забезпечує користувачам можливість організації відеоконференцій для взаємодії в реальному часі [7]–[9], [13];

- спільний доступ до екрану: програмне забезпечення Ryver дозволяє користувачам спільно переглядати екрани для взаємного навчання або розв'язання проблем [7]–[9], [13];

- функція автоматичного сповіщення: програмне забезпечення Ryver надає можливість користувачам налаштувати автоматичні сповіщення для певних подій або дій [7]–[9], [13];

- інтеграція з Dropbox і Google Drive: програмне забезпечення Ryver підтримує інтеграцію з популярними сервісами зберігання файлів для зручного обміну документами та іншими матеріалами [7]–[9], [13].

Серед недоліків програмного забезпечення Ryver можна відзначити складність налаштування. Для деяких користувачів програмного забезпечення Ryver налаштування програми може бути складним процесом, особливо для тих, хто не має досвіду у використанні подібного програмного забезпечення.

Крім того, хоча Ryver пропонує багато функцій для підтримки клієнтів, іноді може бути обмежена в реалізації специфічних потреб певної компанії або групи користувачів [7]–[9], [13].

Порівняльну характеристику програмного забезпечення для підтримки процесу обміну текстовими повідомленнями наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для підтримки процесу обміну текстовими повідомленнями

Критерій порівняння	Zendesk [11]	WeChat [12]	Ryver [13]
Зручність засобів пошуку та додавання нових контактів	+-	+	+-
Інтеграція з іншим програмним забезпеченням	+-	+	+
Підтримка різних платформ	+-	+	-
Засоби створення автоматичних відповідей	+	-	+-
Безкоштовність	-	+	-
Засоби підтримки чат ботів	+-	-	+

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для підтримки процесу обміну текстовими повідомленнями. Проте деякі програмні засоби для підтримки процесу обміну текстовими повідомленнями можуть використовувати ненадійні засоби захисту особистих даних користувачів, зокрема текстових повідомлень, що може призвести до витоку конфіденційної інформації. Інші програми обміну повідомленнями можуть мати унікальні функції, які можуть бути зайвими та ускладнювати спілкування між користувачами. Крім того, існує проблема стандартизації форматів текстових повідомлень, що може призвести до проблеми комунікації між користувачами, які користуються

апаратними засобами, що підтримуються різними програмними платформами. Тому актуальною є розробка програмного забезпечення для обміну повідомленнями між користувачами.

При розробці програмного забезпечення обміну повідомленнями необхідно забезпечити такі функціональні вимоги:

- можливість реєстрації та авторизації у програмному забезпеченні обміну повідомленнями;
- можливість створення та видалення чатів;
- можливість адміністрування чатів користувачем, що їх створив, або людиною, яка наділена відповідними правами;
- можливість роботи з різними чатами, зокрема, можливість відправлення повідомлень до різних чатів;
- засоби перегляду повідомлень чатів, де користувач був учасником;
- засоби синхронізації дій користувача з іншими користувачами (зокрема, синхронізація інформації про видалення користувача з чату).

1.3 Висновки за розділом 1

Визначено, що миттєвий обмін повідомленнями являє собою текстовий діалог між двома або більше людьми на комп'ютері або мобільному пристрої через чат, що має доступ до мережі Інтернет. Програмне забезпечення підтримки процесу обміну миттєвими повідомленнями спирається на центральний сервер або сервери для відстеження присутності користувачів та надання можливості прямого та зручного способу спілкування без обмежень часу та відстані. Відзначено, що розуміння суті та важливості процесу миттєвого обміну повідомленнями має вирішальне значення для того, щоб ефективно користуватися цими програмними сервісами та розробляти відповідне програмне забезпечення. Деякі програмні засоби миттєвого обміну повідомленнями перейшли до інтеграції з мережевими протоколами, що зроблено з метою обійти певні обмеження, які можуть бути налаштовані для

протоколу миттєвого обміну повідомленнями у корпоративних мережах. Відзначено, що поруч із зручністю миттєвого обміну даними існують ризики, використання програмних засобів підтримки процесу обміну миттєвими повідомленнями може призвести до передачі інформації у відкритому вигляді, поширення Інтернет посилань в повідомленнях, передачі файлів стороннім особам. Якщо потрібна програмна система обміну миттєвими повідомленнями для виробничої або іншої організації, варто розглянути можливість створення внутрішньої програмної системи обміну миттєвими повідомленнями. Проаналізовано функції та можливості програмних засобів підтримки процесу обміну миттєвими повідомленнями. Проаналізовано етапи функціонування програмного забезпечення для обміну миттєвими повідомленнями.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для підтримки процесу обміну текстовими повідомленнями. Проте деякі програмні засоби для підтримки процесу обміну текстовими повідомленнями можуть використовувати ненадійні засоби захисту особистих даних користувачів, зокрема текстових повідомлень, що може призвести до витоку конфіденційної інформації. Інші програми обміну повідомленнями можуть мати унікальні функції, які можуть бути зайвими та ускладнювати спілкування між користувачами. Крім того, існує проблема стандартизації форматів текстових повідомлень, що може призвести до проблеми комунікації між користувачами, які користуються апаратними засобами, що підтримуються різними програмними платформами. Тому актуальною є розробка програмного забезпечення для обміну повідомленнями між користувачами.

Сформульовано функціональні вимоги до програмного забезпечення обміну повідомленнями.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для обміну повідомленнями було використано мову програмування C# [14], [15].

Мова програмування C# є об'єктно-орієнтованою та призначена для створення програм для платформи Microsoft .NET Framework. Вона базується на синтаксисі, схожому до C++ та Java. C# включає статичну типізацію, можливість поліморфізму, перевантаження операторів, включаючи явні та неявні оператори приведення, а також підтримує використання делегатів, атрибутів, подій, загальних типів та методів, ітераторів, анонімних функцій з підтримкою закриття, LINQ та елементів винятків. Також мова дозволяє використовувати коментарі у форматі XML [14], [15].

Мова програмування C# використовує багато концепцій з інших мов, таких як C++, Delphi, Smalltalk, а особливо Java і C. При цьому вона відкидає певні шаблони, які можуть ускладнювати створення програмних систем. Наприклад, C++ не підтримує множинне успадкування класів, прямий доступ або керування пам'яттю. Тим не менше, важливо зазначити, що C# дає розробникам можливість отримати прямий доступ до пам'яті через написання спеціального "небезпечного" коду [14], [15].

Інша важлива відмінність між C# і C++ полягає в тому, як компілюється та виконується код. Коли незалежний від платформи C++ компілюється в інструкції процесора певного типу на рівні компілятора, то незалежний від платформи C# трансліюється в проміжний код (IL) на рівні середовища виконання. Середовище виконання (CLR) відповідає за виконання та переклад цього коду в інструкції процесора [14], [15].

Цей підхід має кілька переваг: CLR може оптимізувати процес виконання шляхом кешування скомпільованого коду та виконання оптимізованих для процесора інструкцій, щоб уникнути повторної компіляції. Механізм перевірки написаного керованого коду та управління пам'яттю за

допомогою збирачів сміття дозволяє ефективно використовувати ресурси системи та уникати витоків пам'яті [14], [15].

.NET Framework є платформою для розробки програмного забезпечення від Microsoft, яка включає в себе середовище виконання CLR (Common Language Runtime) і обширну бібліотеку класів BCL (Base Class Library). Програми, написані на будь-якій мові, що підтримує платформу .NET, можуть користуватися класами та методами BCL, такими як створення об'єктів класів, виклик їхніх методів, успадкування від необхідних класів BCL та інші [14], [15].

Windows Communication Foundation (WCF) – це інфраструктура для розробки розподілених програм, орієнтованих на служби. WCF дозволяє обмінюватися даними між різними службами у формі асинхронних повідомлень. Служба може бути частиною постійного сервісу, розміщеного в ІІS, або може бути службою, що працює у програмі. Вона може бути і клієнтом служби, який запитує дані від служби. Повідомлення може бути простим, таким як відправка одного символу або слова в форматі XML, або складним, таким як потік бінарних даних [14], [15].

Простота використання є ключовою особливістю деяких технологій, що використовуються у процесі розробки програмного забезпечення. Це можна досягти шляхом приховання деталей мережевого підключення до зовнішньої служби та надання резервного об'єкта класу [14], [15].

Мова програмування C# є однією з основних мов програмування, що використовуються для розробки програмного забезпечення для платформи .NET Framework. C# має синтаксис, схожий до інших мов програмування, таких як C++ та Java, і вона надає розробникам широкі можливості для створення різноманітних програмних застосунків, включаючи вебзастосунки, програми для персональних комп'ютерів, мобільні застосунки та програми іншого призначення [14], [15].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для обміну повідомленнями наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для обміну повідомленнями

Критерій порівняння мов програмування	Мова програмування		
	C#	Java	C++
Інтеграція з іншими технологіями розробки ПЗ	+	+	+-
Зручність для створення ПЗ для обміну повідомленнями	+	-	-
Швидкодія	+	+-	+
Використання платформи .NET Framework	+	-	-
Широка спільнота розробників	+	+	+-

Отже, для реалізації для розробки програмного забезпечення для обміну повідомленнями обрано мову програмування C#, яка є об'єктно-орієнтованою мовою та надає розробникам широкі можливості для створення різноманітних програмних застосунків, включаючи вебзастосунки, програми для персональних комп'ютерів, мобільні застосунки та ін.

2.2 Вибір середовища розробки для створення програмного забезпечення для обміну повідомленнями

В якості середовища розробки для створення програмного забезпечення для обміну повідомленнями було обрано середовище JetBrains Rider [16]–[18].

Середовище розробки JetBrains Rider – це повноцінне кросплатформне середовище розробки .NET, яке підтримує різноманітні типи проектів, такі як .NET Framework, Mono та .NET Core. Середовище розробки JetBrains Rider підтримує більшість мов, які використовуються в .NET-розробці, такі як C#,

VB.NET, F#, а також синтаксиси для ASP.NET, XAML, XML, JavaScript, TypeScript, JSON, HTML, CSS і SQL. Завдяки ряду потужних функцій, які працюють з різними мовами, ви зможете створювати високоякісний код швидше, ніж коли-небудь раніше [16]–[18].

JetBrains Rider створений компанією JetBrains, яка також відома своїми іншими популярними інструментами, такими як IntelliJ IDEA, PyCharm і WebStorm. Rider об'єднує в собі можливості ReSharper, відомого плагіна JetBrains для .NET у Visual Studio, та платформу IntelliJ. Це дозволяє Rider надавати відмінне середовище розробки для проектів .NET, одночасно надаючи багаті та універсальні функції, які доступні в екосистемі продуктів JetBrains [16]–[18].

JetBrains Rider є інтегрованим середовищем розробки, яке працює на різних платформах, таких як Windows, macOS і Linux, для розробки на .NET. Розроблений компанією JetBrains, він базується на популярному плагіні ReSharper для Visual Studio, що забезпечує декілька тисяч перевірок коду в реальному часі, рефакторинги та інші корисні функції. Rider підтримує різні платформи .NET, включаючи .NET Framework, .NET Core та Mono, і є відмінним вибором для розробників, які потребують універсального та потужного середовища розробки [16]–[18].

Середовище розробки JetBrains Rider підтримує засоби інтеграції з базами даних. В складі середовища розробки JetBrains Rider є інструменти бази даних JetBrains DataGrip, які забезпечують розширені можливості редагування SQL, виконання запитів та інтелектуальну роботу з базами даних безпосередньо в середовищі розробки [16]–[18].

Будучи побудованим на основі платформи IntelliJ, Rider підтримує використання різноманітних плагінів, розроблених для інших продуктів JetBrains. Це дозволяє значно розширити функціональність та можливості інтеграції інструменту розробки Rider [16]–[18].

Середовище розробки JetBrains Rider підтримує різні операційні систем, таких як Windows, macOS і Linux, а також можливість роботи з

різними платформами .NET, включаючи .NET Framework, .NET Core та Mono, що дозволяє розробляти кросплатформне програмне забезпечення [16]–[18].

Наявність потужного інструменту для налагодження програм на платформах .NET, Mono та .NET Core забезпечує розробникам зручність процесу створення програмного забезпечення [16]–[18]. Середовище розробки ПЗ JetBrains Rider підтримує роботу з інтегрованою корпоративною базою даних IDE DataGrip. Це надає розробникам засоби для редагування, виконання SQL-запитів та роботи з базами даних безпосередньо в середовищі Rider. Rider має прямий доступ до систем контролю версій, таких як Git, Subversion, Mercurial та інших, у своєму інтегрованому середовищі розробки, що полегшує управління кодовою базою та співпрацю над нею [16]–[18].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для обміну повідомленнями наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для обміну повідомленнями

Критерій порівняння середовищ розробки	Середовища розробки		
	JetBrains Rider	Visual Studio	PyCharm
Зручність використання для створення програм на мові C#	+	+	–
Кросплатформність	+	–	+
Швидкодія середовища розробки	+	+–	+
Зручність налагодження та профілювання	+	+	+–
Широкий набір функцій	+	+	–

Таким чином, для створення програмного забезпечення для обміну повідомленнями обрано середовище розробки JetBrains Rider, що є кросплатформним середовищем розробки, яке підтримує різноманітні типи проєктів, такі як .NET Framework, Mono та .NET Core та є зручним засобом розробки програмного забезпечення на мові C#.

2.3 Висновки за розділом 2

Для розробки програмного забезпечення для обміну повідомленнями обрано мову програмування C#, яка є об'єктно-орієнтованою мовою та надає розробникам широкі можливості для створення різноманітних програмних застосунків, включаючи вебзастосунки, програми для персональних комп'ютерів, мобільні застосунки та ін.

Для створення програмного забезпечення для обміну повідомленнями обрано середовище розробки JetBrains Rider, що є кросплатформним середовищем розробки, яке підтримує різноманітні типи проєктів, такі як .NET Framework, Mono та .NET Core та є зручним засобом розробки програмного забезпечення на мові C#.

3 ОПИС ПРОГРАМИ

3.1 Структура программного обеспечения для обмена сообщениями

Структуру программного обеспечения для обмена сообщениями наведено на рис. 3.1.

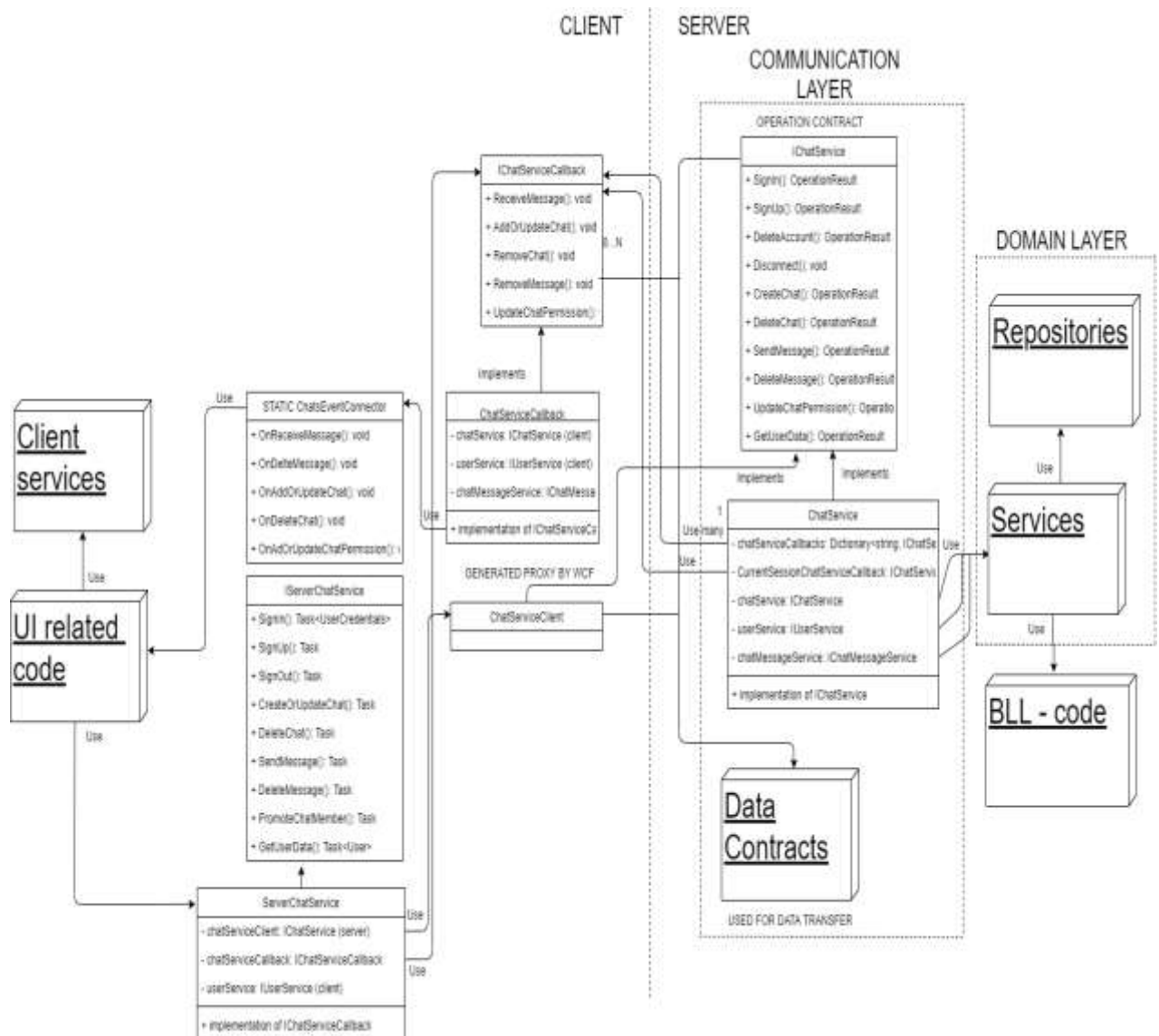


Рисунок 3.1 – Структура программного обеспечения для обмена сообщениями

Як видно з рис. 3.1 програмне забезпечення для обміну повідомленнями складається з клієнтської та серверної частин.

Серверна частина програмного забезпечення для обміну повідомленнями структурно складається з комунікаційного шару, доменного шару, шару бізнес-логіки та шару доступу до даних.

Комунікаційний шар серверної частини програмного забезпечення для обміну повідомленнями включає інтерфейс, за допомогою якого клієнти можуть взаємодіяти з програмним сервісом обміну повідомленнями, а також інтерфейс для обміну інформацією у мережі.

Доменний шар серверної частини програмного забезпечення для обміну повідомленнями включає моделі, що використовуються в основному коді сервера; опис і реалізацію сервісів, які керують отриманими даними від та обробленими класами бізнес-логіки.

Шар бізнес-логіки серверної частини програмного забезпечення для обміну повідомленнями складається з класів, які виконують специфічну логіку, характерну для підтримки процесу обміну повідомленнями.

Шар доступу до даних серверної частини програмного забезпечення для обміну повідомленнями потрібен для налаштування процесів взаємодії з базою даних.

Клієнтська частина програмного забезпечення для обміну повідомленнями містить шар представлення, який забезпечує інтерфейс користувача, та шар взаємодії з сервером.

Діаграму компонентів програмного забезпечення для обміну повідомленнями наведено на рис. 3.2.

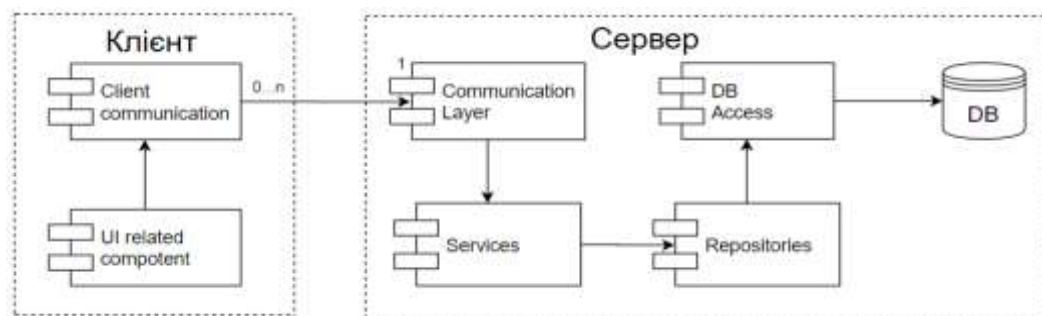


Рисунок 3.2 – Діаграма компонентів програмного забезпечення для обміну повідомленнями

Така логічна структура програмного забезпечення для обміну повідомленнями значно знижує зв'язність компонентів програми і спрощує модулів при необхідності розвитку модифікації програмної системи.

3.2 Функціонування програмного забезпечення для обміну повідомленнями

Функціонування програмного забезпечення для обміну повідомленнями подамо за допомогою схеми, зображеної на рис. 3.3.

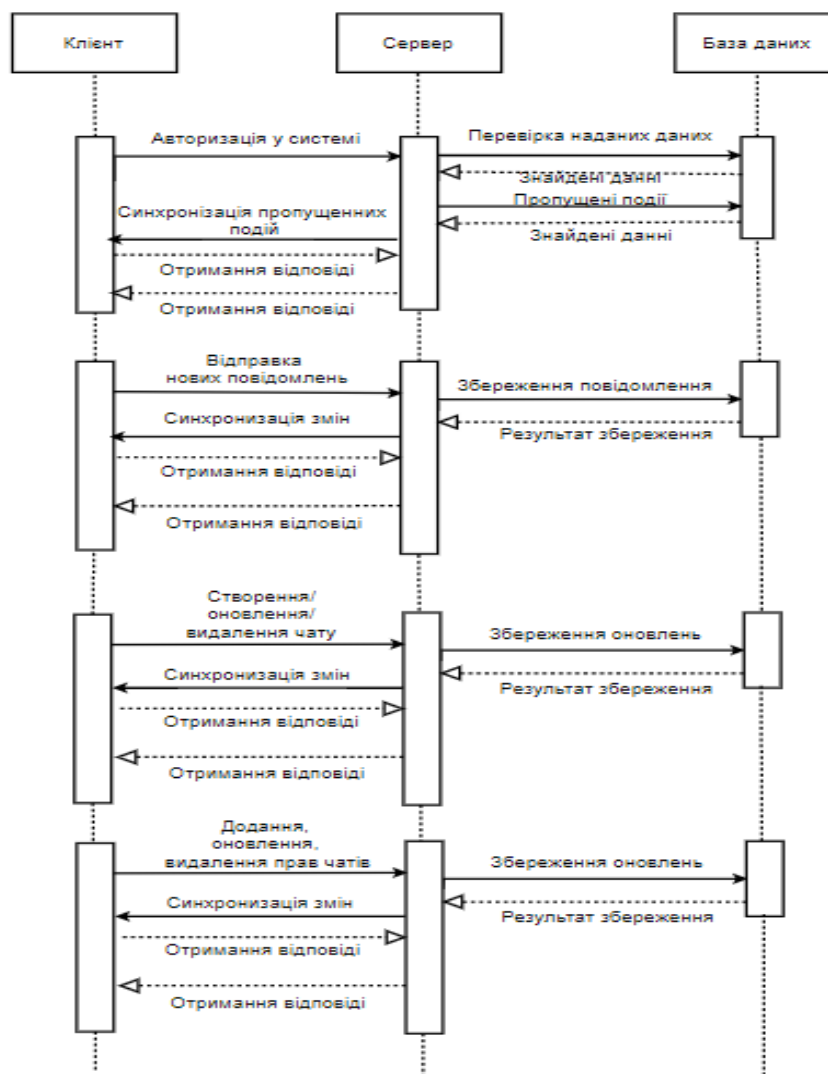


Рисунок 3.3 – Схема функціонування програмного забезпечення для обміну повідомленнями

Програмне забезпечення для обміну повідомленнями розроблено з урахуванням того, що, як правило, запити до сервера (такі як підключення, створення або видалення кімнати спілкування) викликають зворотну синхронізацію даних із сервером: клієнт виконує певну операцію, сервер обробляє її та, якщо вона успішна, повідомляє всіх зацікавлених користувачів про цю подію за допомогою механізму зворотного виклику (включаючи ініціатора). Це забезпечує стійкий механізм оновлень з єдиною точкою входу для оновлень, тому нам не потрібно додатково контролювати відображення повідомлень або реагувати на помилки виконання.

Функціонування програмного забезпечення для обміну повідомленнями може забезпечити роботу трьох груп користувачів.

Звичайний користувач – особа, яка може надсилати та читати повідомлення. Діаграма прецедентів ролі користувача програмного забезпечення для обміну повідомленнями наведена на рис. 3.4.

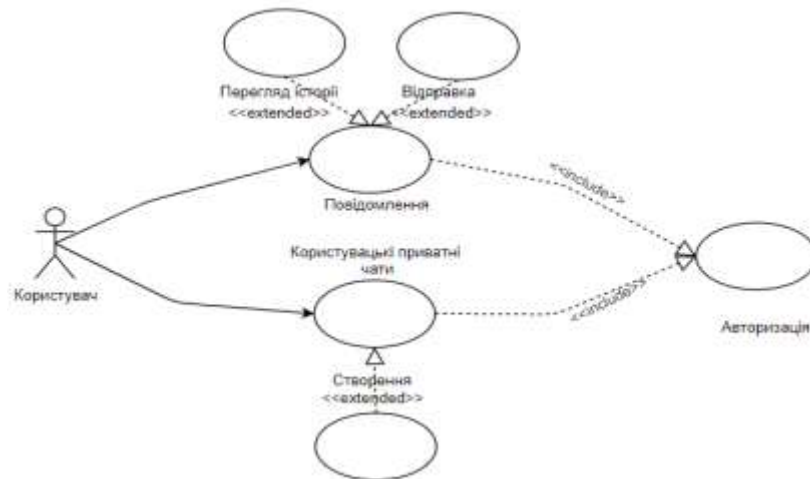


Рисунок 3.4 – Діаграма прецедентів програмного забезпечення для обміну повідомленнями, роль Користувач

Адміністратор – особа, яка має права звичайного користувача, а також додаткові права, пов'язані з адмініструванням чатів. Діаграма прецедентів ролі адміністратора програмного забезпечення для обміну повідомленнями наведена на рис. 3.5.

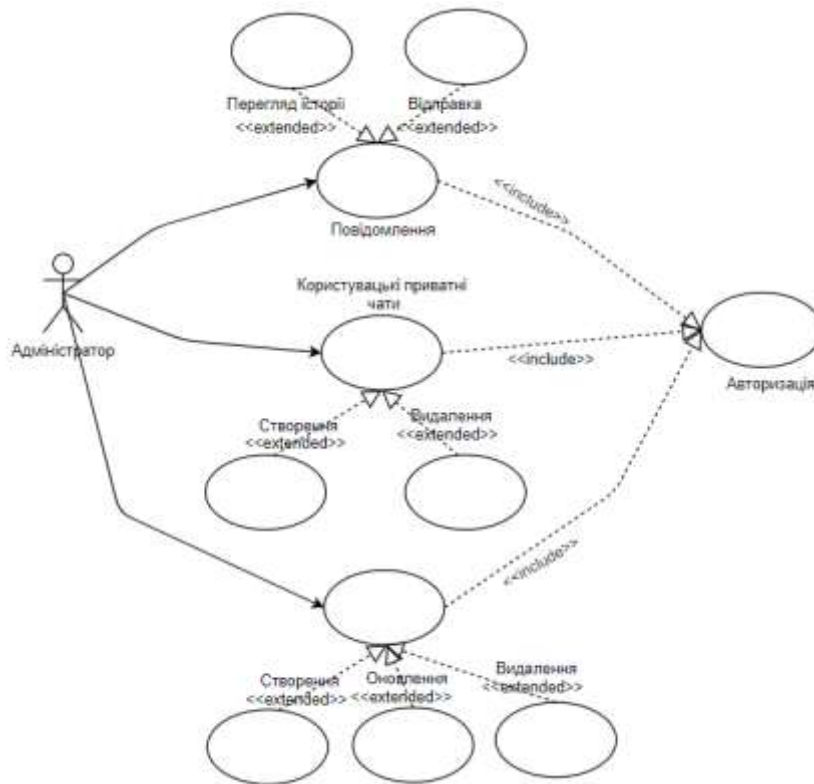


Рисунок 3.5 – Діаграма прецедентів програмного забезпечення для обміну повідомленнями, роль Адміністратор

Читач – особа, яка може читати повідомлення чату, але не надсилати їх. Діаграма прецедентів ролі читача програмного забезпечення для обміну повідомленнями наведена на рис. 3.6.

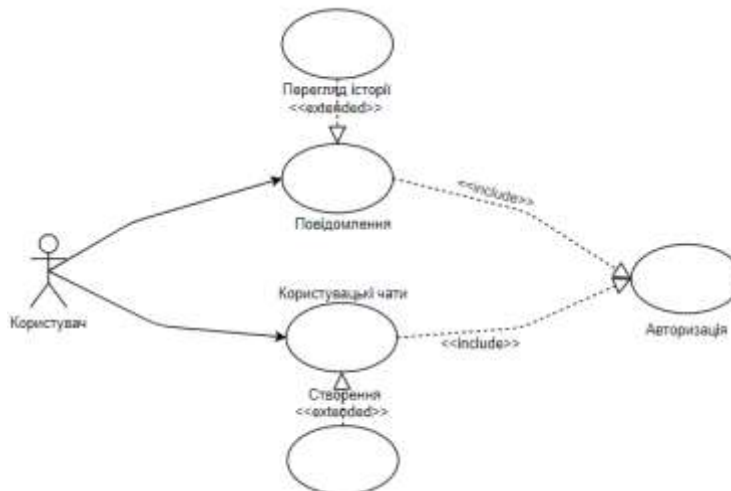


Рисунок 3.6 – Діаграма прецедентів програмного забезпечення для обміну повідомленнями, роль Читач

Діаграму станів програмного забезпечення для обміну повідомленнями наведено на рис. 3.7.

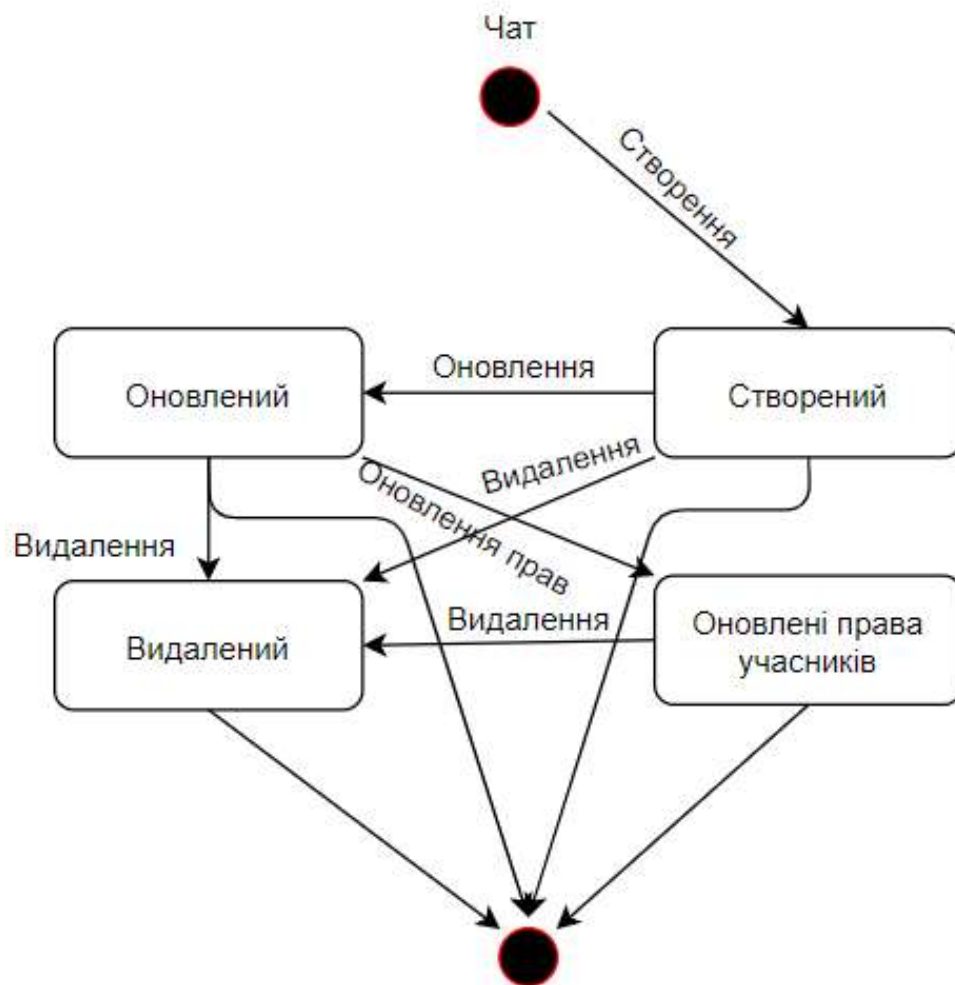


Рисунок 3.7 – Діаграма станів програмного забезпечення для обміну повідомленнями

3.3 Розробка бази даних програмного забезпечення для обміну повідомленнями

База даних програмного забезпечення для обміну повідомленнями призначена для зберігання даних про користувачів чатів, повідомлення, які вони відправляють, чати, де спілкуються користувачі та їх параметри (користувачі, повідомлення та ін.).

Схему бази даних програмного забезпечення для обміну повідомленнями наведено на рис. 3.8.

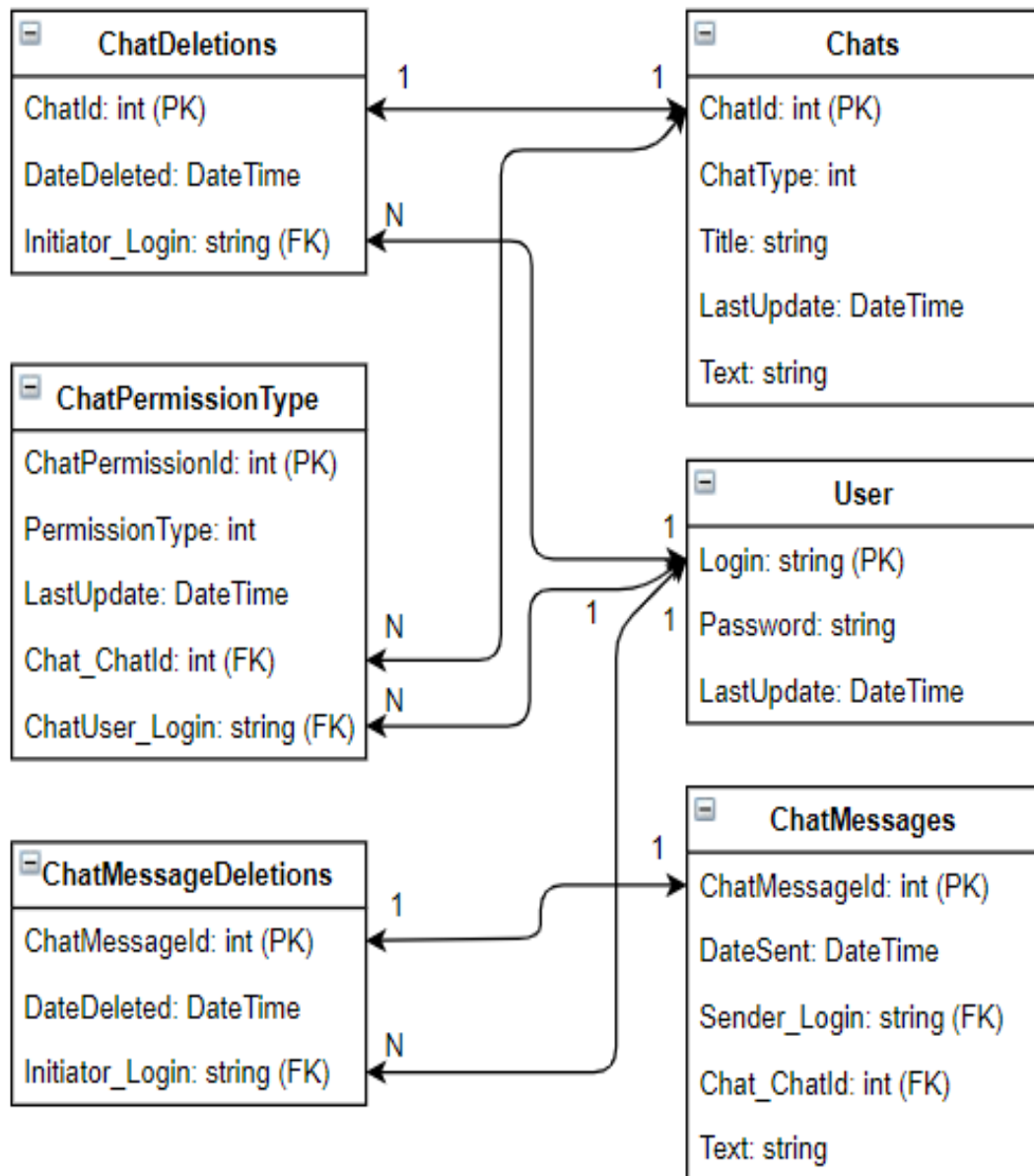


Рисунок 3.8 – Схема бази даних програмного забезпечення для обміну повідомленнями

Схема бази даних програмного забезпечення для обміну повідомленнями відображає концептуальні зв'язки між обраними елементами у відповідній предметній області. Основними сутностями бази даних обрано

такі: користувач (User), чати, повідомлення у чатах, типи чату, видалені чати, видалені повідомлення чатів.

3.4 Проєктування інтерфейсу програмного забезпечення для обміну повідомленнями

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для обміну повідомленнями виконано з урахуванням вимог технічного завдання. При розробленні програмного забезпечення для підтримки процесу обміну повідомленнями враховані функціональні вимоги до програми:

- можливість реєстрації та авторизації у програмному забезпеченні обміну повідомленнями;
- можливість створення та видалення чатів;
- можливість адміністрування чатів користувачем, що їх створив, або людиною, яка наділена відповідними правами;
- можливість роботи з різними чатами, зокрема, можливість відправлення повідомлень до різних чатів;
- засоби перегляду повідомлень чатів, де користувач був учасником;
- засоби синхронізації дій користувача з іншими користувачами (зокрема, синхронізація інформації про видалення користувача з чату).

Інтерфейс форми авторизації у програмному забезпечення для обміну повідомленнями наведено на рис. 3.9.

Інтерфейс головної форми програмного забезпечення для обміну повідомленнями наведено на рис. 3.10.

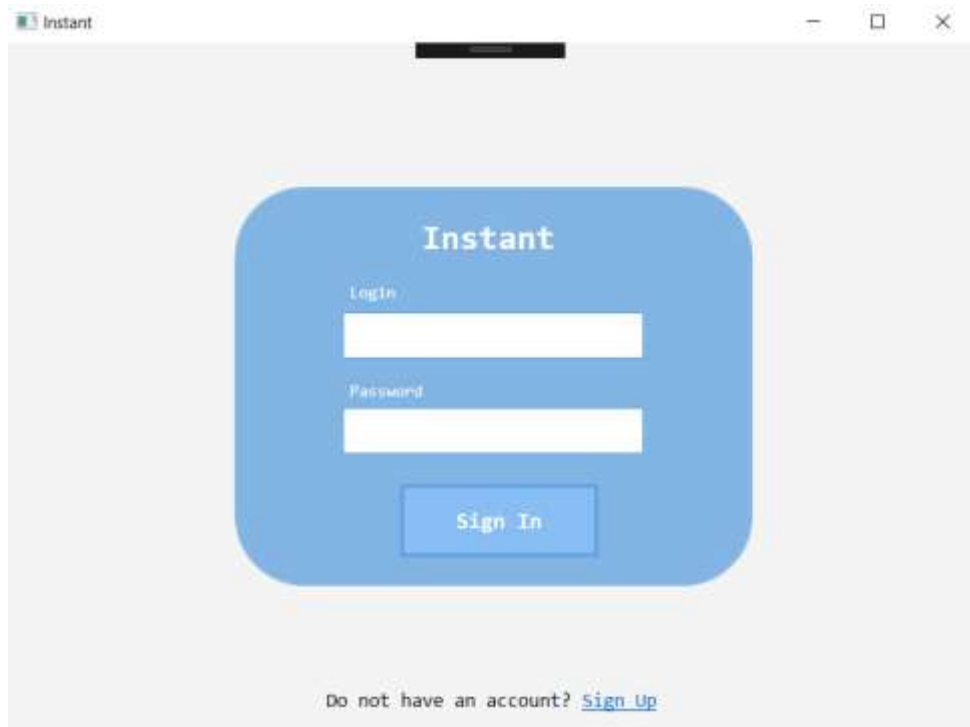


Рисунок 3.9 – Інтерфейс форми авторизації програмного забезпечення для обміну повідомленнями

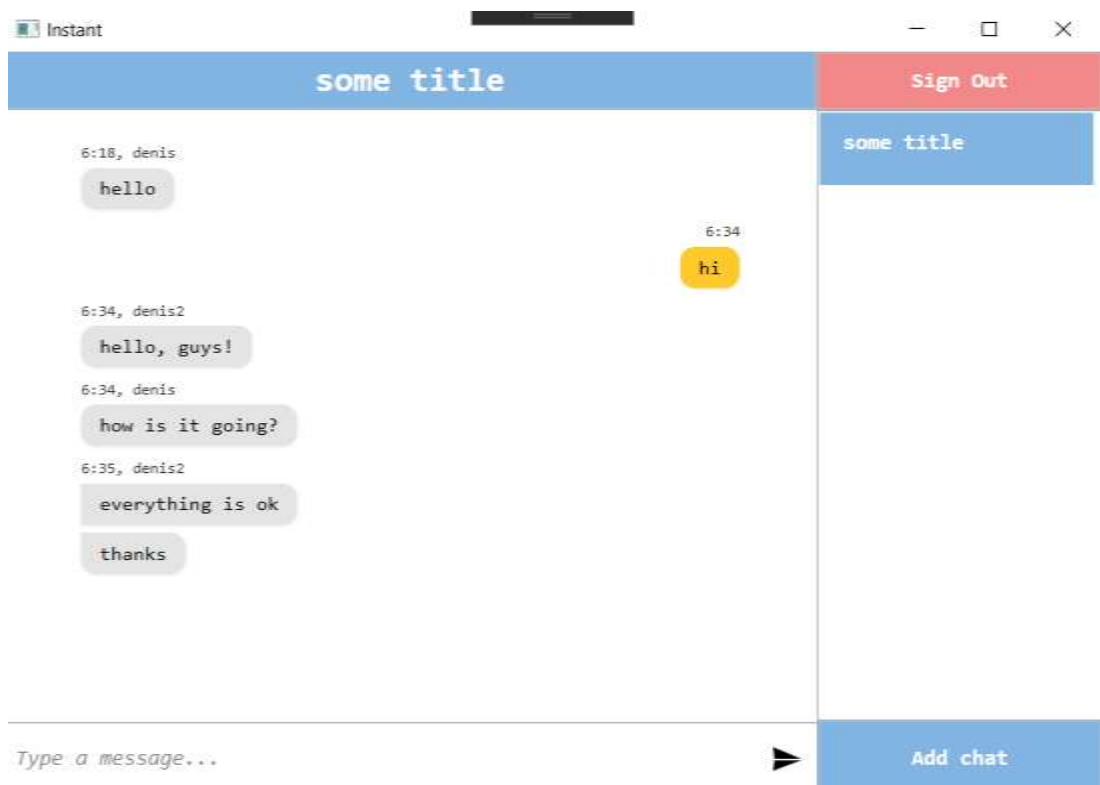


Рисунок 3.10 – Інтерфейс головної форми програмного забезпечення для обміну повідомленнями

Форму створення чату у програмному забезпеченні для обміну повідомленнями наведено на рис. 3.11.



The 'Create chat' form is a blue rounded rectangle. At the top, it has the title 'Create chat'. Below the title, there is a section 'Select chat type:' with two radio buttons: 'Private chat' (selected) and 'Public group'. Underneath, there is a text input field labeled 'Enter name for the group:'. Below that is another text input field labeled 'Enter group member login:' with a blue '+' button to its right. Below this is a text area labeled 'Chat members:'. At the bottom, there are two buttons: a red 'Cancel' button and a blue 'Create' button.

Рисунок 3.11 – Форма створення чату у програмному забезпеченні для обміну повідомленнями

Форму оновлення чату у програмному забезпеченні для обміну повідомленнями наведено на рис. 3.12.



The 'Update chat' form is a blue rounded rectangle. At the top, it has the title 'Update chat'. Below the title, there is a text input field labeled 'Enter name for the group:' with the text 'some title' entered. Below this is a text area labeled 'Chat members:' containing the text 'denis', 'denis1', and 'denis2'. At the bottom, there are four buttons: a red 'Leave' button, a red 'Delete chat' button, a red 'Cancel' button, and a blue 'Update' button.

Рисунок 3.12 – Форма оновлення чату у програмному забезпеченні для обміну повідомленнями

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для обміну повідомленнями.

Запропоновано структуру програмного забезпечення для обміну повідомленнями. Визначено, що програмне забезпечення для обміну повідомленнями складається з клієнтської та серверної частин. Серверна частина програмного забезпечення для обміну повідомленнями структурно складається з комунікаційного шару, доменного шару, шару бізнес-логіки та шару доступу до даних. Клієнтська частина програмного забезпечення для обміну повідомленнями містить шар представлення, який забезпечує інтерфейс користувача, та шар взаємодії з сервером.

Розроблено базу даних програмного забезпечення для обміну повідомленнями, яка відображає концептуальні зв'язки між обраними елементами у відповідній предметній області обміну повідомлення. Визначено, що основними сутностями бази даних є такі: користувач (User), чати, повідомлення у чатах, типи чату, видалені чати, видалені повідомлення чатів.

Описано функціонування програмного забезпечення для обміну повідомленнями.

Описано особливості реалізації програмного забезпечення для обміну повідомленнями. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для обміну повідомленнями.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для підтримки процесу обміну повідомленнями. Програма написана на мові C#, яка є об'єктно-орієнтованою мовою та надає розробникам широкі можливості для створення різноманітних програмних застосунків, включаючи вебзастосунки, програми для персональних комп'ютерів, мобільні застосунки та ін. В якості середовища розробки для створення програмного забезпечення для обміну повідомленнями обрано JetBrains Rider, що є кросплатформним середовищем розробки, яке підтримує різноманітні типи проєктів, такі як .NET Framework, Mono та .NET Core та є зручним засобом розробки програмного забезпечення на мові C#.

Програмне забезпечення для підтримки процесу обміну повідомленнями забезпечує виконання таких функцій:

- можливість реєстрації та авторизації у програмному забезпеченні обміну повідомленнями;
- можливість створення та видалення чатів;
- можливість адміністрування чатів користувачем, що їх створив, або людиною, яка наділена відповідними правами;
- можливість роботи з різними чатами, зокрема, можливість відправлення повідомлень до різних чатів;
- засоби перегляду повідомлень чатів, де користувач був учасником;
- засоби синхронізації дій користувача з іншими користувачами (зокрема, синхронізація інформації про видалення користувача з чату).

Для експлуатації програмного забезпечення для підтримки процесу обміну повідомленнями необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом 800 Мб і більше;
- не менше 1Гб вільного простору на запам'ятовуючому пристрою апаратного засобу для збереження файлів;

- наявність доступу до мережі Інтернет;
- наявність клавіатури для введення текстових повідомлень;
- встановлена операційна система та веббраузер на стороні клієнта.

Функціональні характеристики розробленого програмного забезпечення для обміну повідомленнями наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для обміну повідомленнями наведено у додатку Б.

4.2 Характеристики програми для підтримки процесу обміну повідомленнями

Програмне забезпечення для підтримки процесу обміну повідомленнями складається з двох основних компонентів — серверної та клієнтської частин.

Серверна частина програмного забезпечення для обміну повідомленнями структурно складається з комунікаційного шару, доменного шару, шару бізнес-логіки та шару доступу до даних.

Комунікаційний шар серверної частини програмного забезпечення для обміну повідомленнями включає інтерфейс, за допомогою якого клієнти можуть взаємодіяти з програмним сервісом обміну повідомленнями, а також інтерфейс для обміну інформацією у мережі.

Доменний шар серверної частини програмного забезпечення для обміну повідомленнями включає моделі, що використовуються в основному коді сервера; опис і реалізацію сервісів, які керують отриманими даними від та обробленими класами бізнес-логіки.

Шар бізнес-логіки серверної частини програмного забезпечення для обміну повідомленнями складається з класів, які виконують специфічну логіку, характерну для підтримки процесу обміну повідомленнями.

Шар доступу до даних серверної частини програмного забезпечення для обміну повідомленнями потрібен для налаштування процесів взаємодії з базою даних.

Клієнтська частина програмного забезпечення для обміну повідомленнями містить шар представлення, який забезпечує інтерфейс користувача, та шар взаємодії з сервером.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для підтримки процесу обміну повідомленнями на серверній частини необхідно запустити виконуваний файл програми.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для підтримки процесу обміну повідомленнями є текстові повідомлення користувача, його реєстраційні дані, необхідні для входу у програму та ін.

Вихідними даними програмного забезпечення для підтримки процесу обміну повідомленнями є створені візуальні форми користувача для взаємодії з програмою, текстові повідомлення, що отримає користувач від інших користувачів.

4.3.3 Повідомлення

Користувач програмного забезпечення для підтримки процесу обміну повідомленнями може отримати такі повідомлення: повідомлення про помилку при авторизації або реєстрації у програмі, повідомлення про

проблему відправки текстового повідомлення (можлива відсутність доступу до Інтернету), повідомлення про проблему створення нового чату та ін.

4.4 Виконання програмного забезпечення для підтримки процесу обміну повідомленнями

Після запуску програмного забезпечення для підтримки процесу обміну повідомленнями користувачу відображається форма авторизації, де користувач повинен увести свої дані для входу в систему (рис. 4.1).



Рисунок 4.1 – Інтерфейс форми авторизації програмного забезпечення для підтримки процесу обміну повідомленнями

Після авторизації користувачу відображається головна форма, де він може вводити текст повідомлень та створювати нові чати.

Інтерфейс головної форми програмного забезпечення для обміну повідомленнями наведено на рис. 4.2.

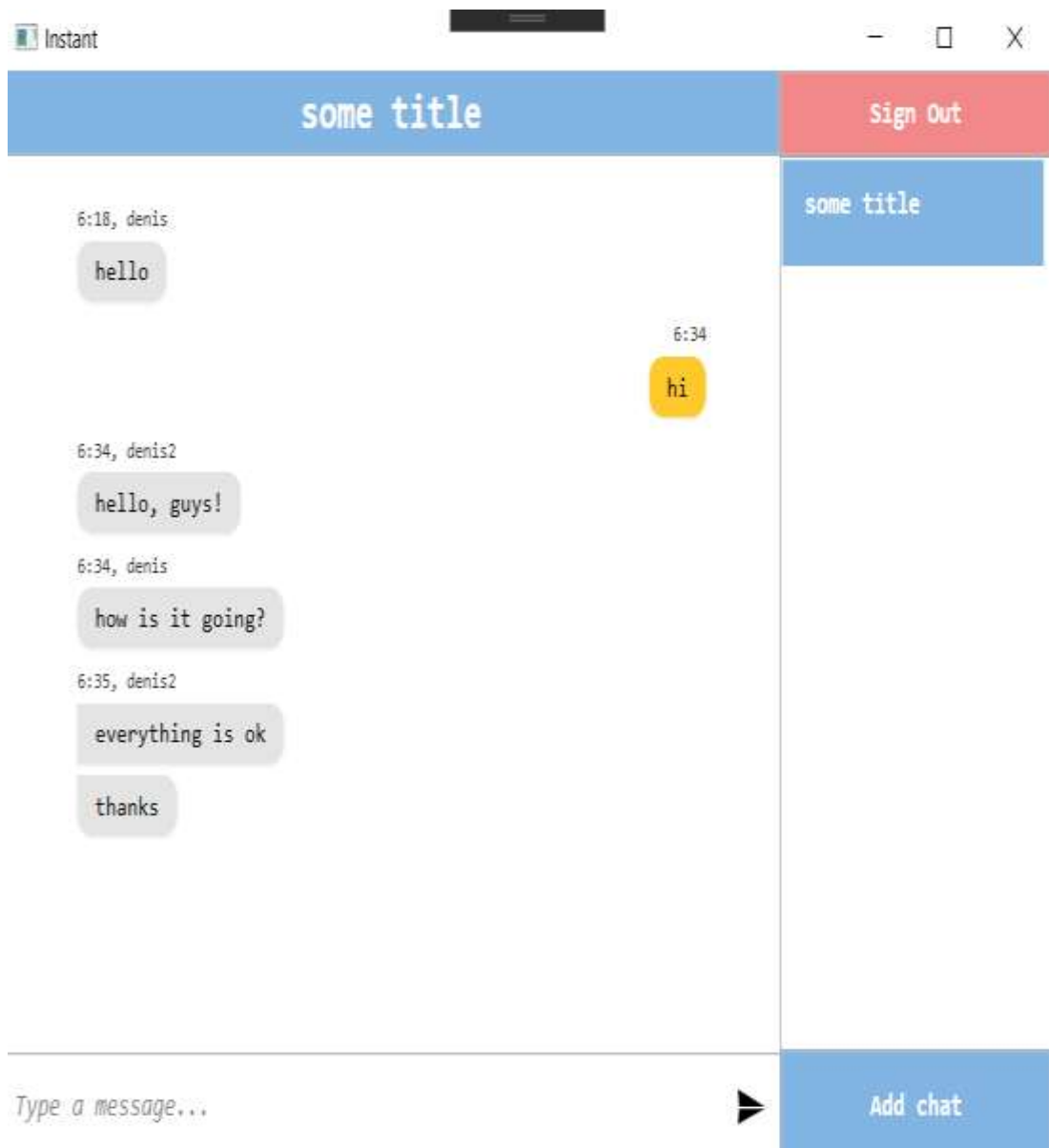


Рисунок 4.2 – Інтерфейс головної форми програмного забезпечення для обміну повідомленнями

Для створення нового чату користувачу програмного забезпечення для обміну повідомленнями необхідно натиснути відповідну кнопку.

У вікні, що з'явилося, необхідно обрати тип чату (приватний чи публічний), ввести його назву та за необхідності додати учасників спілкування.

Форму створення чату у програмному забезпеченні для обміну повідомленнями наведено на рис. 4.3.



The 'Create chat' form is a blue rounded rectangle. At the top, it has the title 'Create chat'. Below the title, there is a section 'Select chat type:' with two radio buttons: 'Private chat' and 'Public group', with 'Public group' selected. Underneath is a text input field labeled 'Enter name for the group:'. Below that is another text input field labeled 'Enter group member login:' with a blue '+' button to its right. Below the login field is a larger text area labeled 'Chat members:'. At the bottom, there are two buttons: a red 'Cancel' button and a blue 'Create' button.

Рисунок 4.3 – Форма створення чату у програмному забезпеченні для обміну повідомленнями

При необхідності користувач, що створив відповідний чат, може змінити його параметри, зокрема видалити певних користувачів, змінити назву чату, а, при необхідності, видалити і весь чат (рис. 4.4).



The 'Update chat' form is a blue rounded rectangle. At the top, it has the title 'Update chat'. Below the title, there is a text input field labeled 'Enter name for the group:' containing the text 'some title'. Below that is a text area labeled 'Chat members:' containing the text 'denis', 'denis1', and 'denis2'. Below the text area are three red buttons: 'Leave', 'Delete chat', and 'Cancel'. At the bottom right, there is a blue 'Update' button.

Рисунок 4.4 – Форма оновлення чату у програмному забезпеченні для обміну повідомленнями

4.5 Тестування програмного забезпечення для підтримки процесу обміну повідомленнями

Виконано тестування програмного забезпечення для підтримки процесу обміну повідомленнями.

Фрагмент чеклисту для тестування програмного забезпечення для підтримки процесу обміну повідомленнями наведено на рис. 4.5.

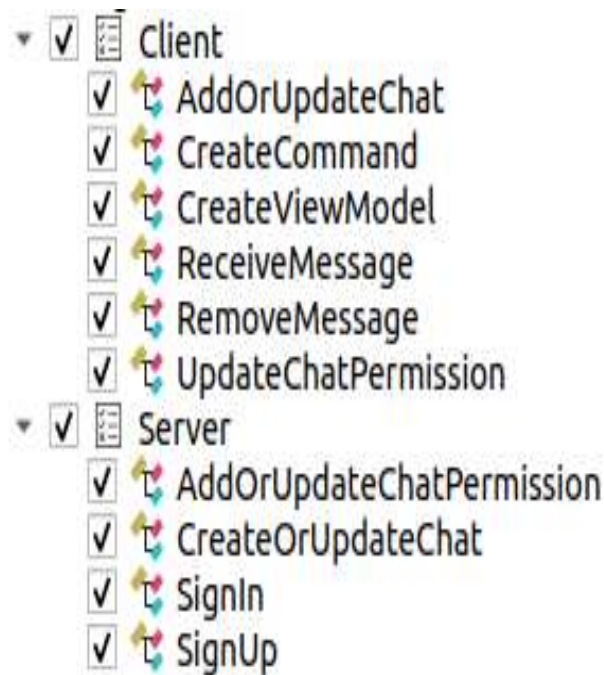


Рисунок 4.5 – Фрагмент чеклисту для тестування програмного забезпечення для підтримки процесу обміну повідомленнями

Виявлено, що програма для підтримки процесу обміну повідомленнями функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу обміну повідомленнями.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою процесу обміну повідомленнями.

4.6 Висновки за розділом 4

Описано програмне забезпечення для підтримки процесу обміну повідомленнями. Виконано тестування розробленого програмного забезпечення для підтримки процесу обміну повідомленнями. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу обміну повідомленнями.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для обміну текстовими повідомленнями.

Визначено, що миттєвий обмін повідомленнями являє собою текстовий діалог між двома або більше людьми на комп'ютері або мобільному пристрої через чат, що має доступ до мережі Інтернет. Програмне забезпечення підтримки процесу обміну миттєвими повідомленнями спирається на центральний сервер або сервери для відстеження присутності користувачів та надання можливості прямого та зручного способу спілкування без обмежень часу та відстані. Відзначено, що розуміння суті та важливості процесу миттєвого обміну повідомленнями має вирішальне значення для того, щоб ефективно користуватися цими програмними сервісами та розробляти відповідне програмне забезпечення. Деякі програмні засоби миттєвого обміну повідомленнями перейшли до інтеграції з мережевими протоколами, що зроблено з метою обійти певні обмеження, які можуть бути налаштовані для протоколу миттєвого обміну повідомленнями у корпоративних мережах. Відзначено, що поруч із зручністю миттєвого обміну даними існують ризики, використання програмних засобів підтримки процесу обміну миттєвими повідомленнями може призвести до передачі інформації у відкритому вигляді, поширення Інтернет посилань в повідомленнях, передачі файлів стороннім особам. Якщо потрібна програмна система обміну миттєвими повідомленнями для виробничої або іншої організації, варто розглянути можливість створення внутрішньої програмної системи обміну миттєвими повідомленнями. Проаналізовано функції та можливості програмних засобів підтримки процесу обміну миттєвими повідомленнями. Проаналізовано етапи функціонування програмного забезпечення для обміну миттєвими повідомленнями.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для підтримки процесу обміну

текстовими повідомленнями. Проте деякі програмні засоби для підтримки процесу обміну текстовими повідомленнями можуть використовувати ненадійні засоби захисту особистих даних користувачів, зокрема текстових повідомлень, що може призвести до витоку конфіденційної інформації. Інші програми обміну повідомленнями можуть мати унікальні функції, які можуть бути зайвими та ускладнювати спілкування між користувачами. Крім того, існує проблема стандартизації форматів текстових повідомлень, що може призвести до проблеми комунікації між користувачами, які користуються апаратними засобами, що підтримуються різними програмними платформами. Тому актуальною є розробка програмного забезпечення для обміну повідомленнями між користувачами.

Сформульовано функціональні вимоги до програмного забезпечення обміну повідомленнями.

Для розробки програмного забезпечення для обміну повідомленнями обрано мову програмування C#, яка є об'єктно-орієнтованою мовою та надає розробникам широкі можливості для створення різноманітних програмних застосунків, включаючи вебзастосунки, програми для персональних комп'ютерів, мобільні застосунки та ін.

Для створення програмного забезпечення для обміну повідомленнями обрано середовище розробки JetBrains Rider, що є кросплатформним середовищем розробки, яке підтримує різноманітні типи проєктів, такі як .NET Framework, Mono та .NET Core та є зручним засобом розробки програмного забезпечення на мові C#.

Розроблено програмне забезпечення для обміну повідомленнями.

Запропоновано структуру програмного забезпечення для обміну повідомленнями. Визначено, що програмне забезпечення для обміну повідомленнями складається з клієнтської та серверної частин. Серверна частина програмного забезпечення для обміну повідомленнями структурно складається з комунікаційного шару, доменного шару, шару бізнес-логіки та шару доступу до даних. Клієнтська частина програмного забезпечення для

обміну повідомленнями містить шар представлення, який забезпечує інтерфейс користувача, та шар взаємодії з сервером.

Розроблено базу даних програмного забезпечення для обміну повідомленнями, яка відображає концептуальні зв'язки між обраними елементами у відповідній предметній області обміну повідомлення. Визначено, що основними сутностями бази даних є такі: користувач (User), чати, повідомлення у чатах, типи чату, видалені чати, видалені повідомлення чатів.

Описано функціонування програмного забезпечення для обміну повідомленнями.

Описано особливості реалізації програмного забезпечення для обміну повідомленнями. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для обміну повідомленнями.

Виконано тестування розробленого програмного забезпечення для підтримки процесу обміну повідомленнями. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу обміну повідомленнями.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Instant messaging [Electronic resource]. – Access mode: <https://www.cleverism.com/lexicon/instant-messaging/>.
2. Instant messaging [Electronic resource]. – Access mode: <https://www.britannica.com/topic/instant-messaging>.
3. Instant Messaging [Electronic resource]. – Access mode: <https://www.sciencedirect.com/topics/computer-science/instant-messaging>.
4. The Evolution of Instant Messaging: A Comprehensive Guide [Electronic resource]. – Access mode: <https://nativemsg.com/resources/text-marketing/the-evolution-of-instant-messaging-a-comprehensive-guide/>.
5. Introduction to Instant Messaging [Electronic resource]. – Access mode: <https://www.sciencedirect.com/science/article/abs/pii/B9781597490177500069>.
6. How Instant Messaging Works [Electronic resource]. – Access mode: <https://computer.howstuffworks.com/e-mail-messaging/instant-messaging.htm>.
7. Top Free Business Instant Messaging Software [Electronic resource]. – Access mode: <https://www.g2.com/categories/business-instant-messaging/free>.
8. Instant messaging software [Electronic resource]. – Access mode: <https://www.zendesk.com/service/messaging/instant-messaging-software/>.
9. Best Instant Messaging Platforms for your Business [Electronic resource]. – Access mode: <https://www.twilio.com/en-us/blog/instant-messaging-platforms>.
10. Instant Messaging Software [Electronic resource]. – Access mode: <https://financesonline.com/c/instant-messaging-software/>.
11. Zendesk [Electronic resource]. – Access mode: <https://www.zendesk.com/>.
12. WeChat [Electronic resource]. – Access mode: <https://www.wechat.com/en/>.
13. Ryver [Electronic resource]. – Access mode: <https://ryver.com/>.

14. Design Patterns in C# [Electronic resource]. – Access mode: <https://www.pluralsight.com/paths/design-patterns-in-c>.

15. C#, The Language For All Platforms [Electronic resource]. – Access mode: <https://hackaday.com/2020/02/09/c-the-language-for-all-platforms-now-including-windows-3-11-and-dos/>.

16. JetBrains Rider [Electronic resource]. – Access mode: <https://www.jetbrains.com/help/rider/Introduction.html>.

17. What is JetBrains Rider? A Comprehensive Overview [Electronic resource]. – Access mode: <https://medium.com/@shani12devani/what-is-jetbrains-rider-a-comprehensive-overview-3ca07a096a4>.

18. JetBrains Rider IDE [Electronic resource]. – Access mode: <https://thetechnologyvault.com/jetbrains-rider-ide>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для підтримки процесу обміну повідомленнями.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмне забезпечення обміну повідомленнями», затверджене наказом Національного університету «Запорізька політехніка» № 168 від 24 квітня 2024 р.

A.2 Призначення розробки

Програмний продукт призначений для підтримки процесу обміну повідомленнями.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для підтримки процесу обміну повідомленнями забезпечує виконання таких функцій:

- можливість реєстрації та авторизації у програмному забезпеченні обміну повідомленнями;
- можливість створення та видалення чатів;
- можливість адміністрування чатів користувачем, що їх створив, або людиною, яка наділена відповідними правами;
- можливість роботи з різними чатами, зокрема, можливість відправлення повідомлень до різних чатів;
- засоби перегляду повідомлень чатів, де користувач був учасником;

– засоби синхронізації дій користувача з іншими користувачами (зокрема, синхронізація інформації про видалення користувача з чату).

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для підтримки процесу обміну повідомленнями повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення для підтримки процесу обміну повідомленнями повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для підтримки процесу обміну повідомленнями необхідна наявність персонального комп'ютера або іншого апаратного засобу з доступом до мережі Інтернет.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення для підтримки процесу обміну повідомленнями необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом 800 Мб і більше;
- не менше 1Гб вільного простору на запам'ятовуючому пристрою апаратного засобу для збереження файлів;
- наявність доступу до мережі Інтернет;

- наявність клавіатури для введення текстових повідомлень;
- встановлена операційна система та веббраузер на стороні клієнта.

A.3.6 Вимоги до маркування і пакування

Програма для підтримки процесу обміну повідомленнями може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для підтримки процесу обміну повідомленнями».

A.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для підтримки процесу обміну повідомленнями є текстові повідомлення користувача, його реєстраційні дані, необхідні для входу у програму та ін.

Вихідними даними програмного забезпечення для підтримки процесу обміну повідомленнями є створені візуальні форми користувача для взаємодії з програмою, текстові повідомлення, що отримає користувач від інших користувачів.

ДОДАТОК Б
Фрагмент тексту програми

```

public static class Msgfs
{
    public static bool IsMsgfVld(ChtrMsgf chtrMsgf)
    {
        return !string.IsNullOrEmpty(chtrMsgf.Text)
            && !string.IsNullOrWhiteSpace(chtrMsgf.Text);
    }
}

public static class KrstvhS
{
    public static bool IsKrstvhDtNq(Krstvh krstvh, IList<Krstvh>
krstvhs)
    {
        bool snQ = krstvhs.All(u => u.Lgn != krstvh.Lgn);
        return snQ;
    }

    public static bool AreKrstvhCrdntlsValid(string lgn, string
pswrd)
    {
        return !string.IsNullOrEmpty(lgn) &&
!string.IsNullOrWhiteSpace(lgn)
            && !string.IsNullOrEmpty(pswrd) &&
!string.IsNullOrWhiteSpace(pswrd);
    }
}

public enum ChtrPrmsTypes
{
    None = 0,
    Bnd = 1,
    Chtch = 2,
    ChtchWr = 3,
    Mdrt = 4,
    Amstr = 5
}

public enum PrtrsIttp
{
    Usph = 1,
    Flr = 2
}

public enum KrstvhEvtp
{

```

```

        KrstvhKntked = 1,
        KrstvhDiskntked = 2
    }

    public Prtrslt Sgnf(AthrzdKrstvhOptns optns)
    {
        Prtrslt prtrslt;

    try
    {
        var dmnoptns = mpr.Map<Dmnoptns.AthrzdKrstvhOptns>(optns);
        var dmnrstvh = this.krstvhCrvC.Sgnf(dmnoptns);

        var klbck = this.KrntsnsChtrCrvC.Klbck;
        if (this.chtrCrvC.Klbcks.KntnsVl(klbck))
        {
            throw new ArgumentException($"Can not kntk client (krstvh
#{dmnrstvh.Lgn}). Client is alchtchy kntked");
        }

        lock (snobj)
        {
            this.chtrCrvC.Klbcks.Add(dmnrstvh.Lgn, klbck);
        }

        var dmnrstvhChtrUpdates =
this.chtrCrvC.GetMsdChtrUpdates(dmnrstvh.Lgn, optns.LastOnline);
        var msdChtrUpdatesOptns =
this.mpr.Map<Ilst<KrtptdtChtrOptns>>(dmnrstvhChtrUpdates);
        foreach (var msdChtrPdtOpt in msdChtrUpdatesOptns)
        {
            var dmnrstvhChtrMmbrs =
this.chtrCrvC.GetMmbrsOfChtr(msdChtrPdtOpt.ChtrId);
            msdChtrPdtOpt.MmbrsLgns = dmnrstvhChtrMmbrs
                .Select(mhc => mhc.Lgn)
                .ToList();
            KrtptdtChtrFrkl(dmnrstvh.Lgn, msdChtrPdtOpt);
        }

        var dmnrstvhMsdMsgfs = this.chtrCrvC.GetMsdMsgfs(dmnrstvh.Lgn,
optns.LastOnline);
        var msdMsgfs = mpr.Map<Ilst<SendMsgfOptns>>(dmnrstvhMsdMsgfs);
        foreach (var Msgf in msdMsgfs)
        {
            klbck.ReceiveMsgf(Msgf);
        }
    }

```

```

        var dmnMsdDltdMsgfs =
this.chtrCrvc.GetMsdDltdMsgfs(dmnKrstvh.Lgn, optns.LastOnline);
        foreach (var Msgf in dmnMsdDltdMsgfs)
        {
            klbck.RmvMsgf(Msgf.MsgfId);
        }

        var dmnMsdDltdChtrs =
this.chtrCrvc.GetMsdDltdChtrs(dmnKrstvh.Lgn, optns.LastOnline);
        foreach (var chtr in dmnMsdDltdChtrs)
        {
            klbck.RemoveChtr(chtr.ChtrId);
        }

        var dmnMsdChtrPrmss =
        this.chtrPrmsCrvc.GetMsdChtrPrmss(optns.Lgn,
optns.LastOnline);
        var chtrPrmsOptns =

this.mpr.Map<Ilst<DobapdtChtrPrmsOptns>>(dmnMsdChtrPrmss);
        foreach (var chtrPrms in chtrPrmsOptns)
        {
            klbck.UpdateChtrPrms(chtrPrms);
        }

        var krstvhData = mpr.Map<Krstvh>(dmnKrstvh);
        prtrslt = new Prtrslt(Prtrsltttp.Usph, xtrDt: krstvhData);
    }
    catch (Exception exception)
    {
        prtrslt = new Prtrslt(
            prtrsltType: Prtrsltttp.Flr,
            Msgf: exception.ToString());
    }

    return prtrslt;
}

public void Diskntk(AthrzdKrstvhOptns optns)
{
    try
    {
        var dmnoptns = mpr.Map<Dmnoptns.AthrzdKrstvhOptns>(optns);
        var dmnKrstvh = this.krstvhCrvc.Sgnf(dmnoptns);

        this.chtrCrvcKlbcks.Remove(dmnKrstvh.Lgn);
    }
}

```

```

        catch (Exception exception)
        {
        }
    }

    public Prtrslt GetKrstvhData(string krstvhLgn)
    {
        Prtrslt prtrslt;

        try
        {
            var dmnrKrstvh = this.krstvhCrvc.GetKrstvh(krstvhLgn);
            var krstvh = this.mpr.Map<Krstvh>(dmnrKrstvh);
            prtrslt = new Prtrslt(
                prtrsltType: PrtrsltType.Usph,
                xtrDt: krstvh);
        }
        catch (Exception exception)
        {
            prtrslt = new Prtrslt(
                prtrsltType: PrtrsltType.Flr,
                Msgf: exception.ToString());
        }

        return prtrslt;
    }

    public Prtrslt KrtpdtChtr(KrtpdtChtrOptns optns)
    {
        Prtrslt prtrslt;

        try
        {
            var dmnoptns = mpr.Map<Dmnoptns.KrtpdtChtrOptns>(optns);
            var dmnrChtr = this.chtrCrvc.KrtpdtChtr(dmnoptns);
            var rsltOptns = this.mpr.Map<KrtpdtChtrOptns>(dmnrChtr);
            rsltOptns.MmbrsLgns = optns.MmbrsLgns;
            if (!rsltOptns.MmbrsLgns.Kntns(optns.IntrLgn))
            {
                rsltOptns.MmbrsLgns.Add(optns.IntrLgn);
            }

            foreach (var krstvhLgn in rsltOptns.MmbrsLgns)
            {
                var chtrMmbr = this.krstvhCrvc.GetKrstvh(krstvhLgn);
                KrtpdtChtrFrkl(
                    chtrMmbr.Lgn,
                    rsltOptns);
            }
        }
    }

```

```

    }

    prtrslt = new Prtrslt(Prtrslttp.Usph);
}
catch (Exception exception)
{
    prtrslt = new Prtrslt(
        prtrsltType: Prtrslttp.Flr,
        Msgf: exception.ToString());
}

    return prtrslt;
}

private void KrtpdtChtrFrkl(
    string krstvhLgn,
    KrtpdtChtrOptns optns)
{
    try
    {
        if (this.chtrCrvcKlbcks.KntnsKey(krstvhLgn))
        {
            var dmnChtrPrms =
this.chtrPrmsCrvc.GetChtrPrms(krstvhLgn, optns.ChtrId);
            var chtrPrmsOptns =
this.mpr.Map<DobapdtChtrPrmsOptns>(dmnChtrPrms);
            optns.XtrDt = chtrPrmsOptns;

            var krstvhsChtrCrvcKlbck =
this.chtrCrvcKlbcks[krstvhLgn];
            krstvhsChtrCrvcKlbck.DobapdtChtr(optns);
        }
    }
    catch (Exception exception)
    {
        Console.WriteLine(exception);
    }
}

public Prtrslt DeleteChtr(DeleteChtrOptns optns)
{
    Prtrslt prtrslt;

    try
    {
        var chtrMmbrs =
this.chtrCrvc.GetMmbrsOfChtr(optns.ChtrId);

```

```

        this.chtrCrvc.DeleteChtr(optns.ChtrId,
optns.IntrLgn);
        prtrslt = new Prtrslt(Prtrsltttp.Usph);

        foreach (var chtrMmbr in chtrMmbrs)
        {
            if
(this.chtrCrvcKlbcks.KntnsKey(chtrMmbr.Lgn))
            {
                var krstvhChtrCrvcKlbck =
this.chtrCrvcKlbcks[chtrMmbr.Lgn];

krstvhChtrCrvcKlbck.RemoveChtr(optns.ChtrId);
            }
        }
    }
    catch (Exception exception)
    {
        Console.WriteLine(exception);
        prtrslt = new Prtrslt(
            prtrsltType: Prtrsltttp.Flr,
            Msgf: exception.ToString());
    }

    return prtrslt;
}

public Prtrslt SendMsgf(SendMsgfOptns sendMsgfOptns)
{
    Prtrslt prtrslt;

    try
    {
        var dmnMsgf =
mpr.Map<DmnModels.ChtrMsgf>(sendMsgfOptns);
        this.chtrCrvc.SendMsgf(dmnMsgf);

        var chtrMmbrs =
this.chtrCrvc.GetMmbrsOfChtr(sendMsgfOptns.ChtrId);
        foreach (var mmbr in chtrMmbrs)
        {
            try
            {
                if
(this.chtrCrvcKlbcks.KntnsKey(mmbr.Lgn))
                {
                    var krstvhChtrCrvcKlbck =
this.chtrCrvcKlbcks[mmbr.Lgn];

```

```

krstvhChtrCrvcKlbck.ReceiveMsgf(sendMsgfOptns);
    }
    }
    catch (Exception exception)
    {
        Console.WriteLine(exception);
    }
}

prtrslt = new Prtrslt(Prtrsltttp.Usph);
}
catch (Exception exception)
{
    prtrslt = new Prtrslt(
        prtrsltType: Prtrsltttp.Flrr,
        Msgf: exception.ToString());
}

return prtrslt;
}

public Prtrslt DeleteMsgf(DeleteMsgfOptns optns)
{
    Prtrslt prtrslt;

    try
    {
        int chtrId =
this.chtrMsgfCrvc.GetChtrIdByMsgfId(optns.MsgfId);
        var chtrMmbrs =
this.chtrCrvc.GetMmbrsOfChtr(chtrId);
        this.chtrMsgfCrvc.DeleteMsgf(optns.MsgfId,
optns.IntrLgn);
        prtrslt = new Prtrslt(Prtrsltttp.Usph);

        foreach (var chtrMmbr in chtrMmbrs)
        {
            var krstvhChtrCrvcKlbck =
this.chtrCrvcKlbcks[chtrMmbr.Lgn];
            krstvhChtrCrvcKlbck.RmvMsgf(optns.MsgfId);
        }
    }
    catch (Exception exception)
    {
        Console.WriteLine(exception);
        prtrslt = new Prtrslt(
            prtrsltType: Prtrsltttp.Flrr,

```

```

        Msgf: exception.ToString());
    }

    return prtrslt;
}

public Prtrslt
UpdateKrstvhPrmsInChtr(DobapdtChtrPrmsOptns optns)
{
    Prtrslt prtrslt;

    try
    {
        var dmnoptns =
mpr.Map<Dmnoptns.DobapdtChtrPrmsOptns>(optns);
        this.chtrPrmsCrvc.DobapdtChtrPrms(dmnoptns);

        if
(this.chtrCrvcKlbcks.KntnsKey(optns.ChtrMmbrLgn))
        {
            var dmnrsltPrms =
this.chtrPrmsCrvc.GetChtrPrms(
                optns.ChtrMmbrLgn,
                optns.ChtrId);
            var rsltPrmsOptns =
this.mpr.Map<DobapdtChtrPrmsOptns>(dmnrsltPrms);
            var krstvhChtrCrvcKlbck =
this.chtrCrvcKlbcks[optns.ChtrMmbrLgn];

            krstvhChtrCrvcKlbck.UpdateChtrPrms(rsltPrmsOptns);
            if (optns.PrmsType == ChtrPrmsTypes.Bnd
                || optns.PrmsType == ChtrPrmsTypes.None)
            {

                krstvhChtrCrvcKlbck.RemoveChtr(dmnoptns.ChtrId);
            }
        }

        prtrslt = new Prtrslt(Prtrsltttp.Usph);
    }
    catch (Exception exception)
    {
        prtrslt = new Prtrslt(
            prtrsltType: Prtrsltttp.Flr,
            Msgf: exception.ToString());
    }

    return prtrslt;
}

```

```

    }
}

```

```

public class InstantServerDbCntxt : DbCntxt
{
    public DbSet<Krstvh> Krstvhs { get; set; }
    public DbSet<Chtr> Chtrs { get; set; }
    public DbSet<ChtrMsgf> ChtrMsgfs { get; set; }
    public DbSet<ChtrPrms> ChtrPrmss { get; set; }
    public DbSet<ChtrDltn> ChtrDltns { get; set; }
    public DbSet<ChtrMsgfDltn> ChtrMsgfDltns { get; set; }
}

```

```

namespace Instant.Server.Dmn.Implementation.Repositories
{
    public class ChtrRpstr : IChtrRpstr
    {
        private chtchonly InstantServerDbCntxt dbCntxt;

        public ChtrRpstr(InstantServerDbCntxt dbCntxt)
        {
            this.dbCntxt = dbCntxt;
        }

        public Chtr GetChtr(int chtrId)
        {
            var chtr = this.dbCntxt.Chtrs.Find(chtrId);
            return chtr;
        }

        public Ilst<Chtr> GetChtrs()
        {
            var chtrs = this.dbCntxt.Chtrs.ToList();
            return chtrs;
        }

        public Ilst<Krstvh> GetMmbrsOfChtr(int chtrId)
        {
            var chtr = GetChtr(chtrId);
            _ = chtr ?? throw new ArgumentException($"Can not get
mmbrs of chtr #{chtrId.ToString()}, it isnt present");

            var krstvhIds = this.dbCntxt.ChtrPrmss
                .Where(uch => uch.Chtr.ChtrId == chtr.ChtrId

```

```

        && (uch.PrmsType != (int) ChtrPrmsTypes.None
        || uch.PrmsType != (int) ChtrPrmsTypes.Bnd))
        .Select(uch => uch.ChtrKrstvh.Lgn)
        .ToList();

    var krstvhhs = this.dbCntxt.Krstvhhs
        .Where(u => krstvhIds
            .Any(dIu => dIu == u.Lgn))
        .ToList();

    return krstvhhs;
}

public IList<Chtr> GetChtrsOfKrstvh(string krstvhLgn)
{
    var chtrIds = this.dbCntxt.ChtrPrmss
        .Where(uch => uch.ChtrKrstvh.Lgn == krstvhLgn)
        .Select(uch => uch.Chtr.ChtrId)
        .ToList();

    var chtrs = chtrIds
        .Select(GetChtr)
        .ToList();

    return chtrs;
}

public void SaveChtr(Chtr chtr)
{
    _ = chtr ?? throw new ArgumentNullException("Can not
save chtr, because null");
    dbCntxt.Chtrs.Dobapdt(chtr);
    dbCntxt.SaveChanges();
}

public IList<ChtrDltn> GetChtrDltns(string krstvhLgn)
{
    var chtrDltns = this.dbCntxt.ChtrDltns
        .Where(chd => chd.Chtr.ChtrPrmss
            .Any(chp => chp.ChtrKrstvh.Lgn == krstvhLgn))
        .ToList();
    return chtrDltns;
}

public void DeleteChtr(Chtr chtr, string intrLgn)
{
    var intr = this.dbCntxt.Krstvhhs.Find(intrLgn);
    var chtrDltn = new ChtrDltn();

```

```

        chtrDltn.ChtrId = chtr.ChtrId;
        chtrDltn.Chtr = chtr;
        chtrDltn.Intr = intr;
        chtrDltn.DateDltd = DateTime.Now;
        this.dbCntxt.ChtrDltns.Dobapdt(chtrDltn);
        this.dbCntxt.SaveChanges();
    }

    public bool IsChtrDltd(int chtrId)
    {
        return this.dbCntxt.ChtrDltns
            .FirstOrDefault(chd => chd.ChtrId == chtrId) ==
null;
    }
}

public class KrstvhRpstr : IKrstvhRpstr
{
    private chtchonly InstantServerDbCntxt dbCntxt;

    public KrstvhRpstr(InstantServerDbCntxt dbCntxt)
    {
        this.dbCntxt = dbCntxt;
    }

    public Krstvh GetKrstvh(string krstvhLgn)
    {
        var krstvh = dbCntxt.Krstvhs.FirstOrDefault(u => u.Lgn ==
krstvhLgn);
        return krstvh;
    }

    public IList<Krstvh> GetAllKrstvhs()
    {
        var krstvhs = dbCntxt.Krstvhs.ToList();
        return krstvhs;
    }

    public IList<Chtr> GetChtrsOfKrstvh(string krstvhLgn)
    {
        var krstvh = GetKrstvh(krstvhLgn);
        _ = krstvh ?? throw new ArgumentException($"Can not get
chtrs of krstvh #{krstvhLgn.ToString()}, no such krstvh");

        var chtrIds = dbCntxt.ChtrPrmss
            .Where(uch => uch.ChtrKrstvh.Lgn == krstvh.Lgn)

```

```

        .Select(uch => uch.Chtr.ChtrId)
        .ToList();

    var chtrs = dbCntxt.Chtrs
        .Where(ch => chtrIds
            .Any(dch => dch == ch.ChtrId))
        .ToList();

    return chtrs;
}

public void SaveKrstvh(Krstvh krstvh)
{
    _ = krstvh ?? throw new ArgumentNullException("Can not
save krstvh, because null");
    this.dbCntxt.Krstvhs.Dobapdt(krstvh);

    this.dbCntxt.SaveChanges();
}

public void DeleteKrstvh(Krstvh krstvh)
{
    _ = krstvh ?? throw new ArgumentNullException("Can not
delete krstvh, because null");
    this.dbCntxt.Krstvhs.Remove(krstvh);

    this.dbCntxt.SaveChanges();
}

public void DeleteKrstvh(string krstvhLgn)
{
    var existingKrstvh = GetKrstvh(krstvhLgn);

    _ = existingKrstvh ?? throw new
ArgumentNullException($"Can not delete vh, because krstvh
#{krstvhLgn.ToString()} null");
    DeleteKrstvh(existingKrstvh);

    this.dbCntxt.SaveChanges();
}
}

public class ChtrMsgfCrvc : IChtrMsgfCrvc
{
    private chtchonly IChtrMsgfRpstr chtrMsgfRpstr;
    private chtchonly IChtrRpstr chtrRpstr;
    private chtchonly IKrstvhRpstr krstvhRpstr;
    private chtchonly IChtrPrmsCrvc chtrPrmsCrvc;

```

```

private chtchonly IMpr mpr;

public ChtrMsgfCrvc(
    IChtrMsgfRpstr chtrMsgfRpstr,
    IChtrRpstr chtrRpstr,
    IKrstvhRpstr krstvhRpstr,
    IChtrPrmsCrvc chtrPrmsCrvc,
    IMpr mpr)
{
    this.chtrMsgfRpstr = chtrMsgfRpstr;
    this.chtrRpstr = chtrRpstr;
    this.krstvhRpstr = krstvhRpstr;
    this.chtrPrmsCrvc = chtrPrmsCrvc;
    this.mpr = mpr;
}

public IList<ChtrMsgf> GetMsMsgfs(int chtrId, DateTime
lastOnline)
{
    var Msgfs = this.chtrMsgfRpstr.GetMsgfs(chtrId);
    var msdMsgfs = Msgfs
        .Where(m => m.DateSent >= lastOnline)
        .ToList();

    var dmnMsMsgfs = mpr.Map<IList<ChtrMsgf>>(msdMsgfs);
    return dmnMsMsgfs;
}

public void SaveMsgf(ChtrMsgf chtrMsgf)
{
    var entityMsgf =
this.mpr.Map<Data.Entities.ChtrMsgf>(chtrMsgf);
    entityMsgf.Chtr =
this.chtrRpstr.GetChtr(chtrMsgf.ChtrId);
    entityMsgf.Sender =
this.krstvhRpstr.GetKrstvh(chtrMsgf.SenderLgn);
    this.chtrMsgfRpstr.SaveMsgf(entityMsgf);
}

public IList<ChtrMsgf> GetMsDltdMsgfs(int chtrId, DateTime
lastOnline)
{
    var chtrMsgfDltnsInChtr =
this.chtrMsgfRpstr.GetChtrMsgfDltns(chtrId);
    var msdChtrMsgfDltns = chtrMsgfDltnsInChtr
        .Where(mhcd => mhcd.DateDltd > lastOnline)
        .ToList();
    var Msgfs = this.chtrMsgfRpstr.GetMsgfs(chtrId);

```

```

        var msdDltdMsgfs = Msgfs
            .Where(mhc => msdChtrMsgfDltns
                .Any(mmhcd => mhc.ChtrMsgfId ==
mmhcd.ChtrMsgfId))
            .ToList();

        var dmnMsDltdMsgfs =
this.mpr.Map<IList<ChtrMsgf>>(msdDltdMsgfs);
        return dmnMsDltdMsgfs;
    }

    public void DeleteMsgf(int chtrMsgfId, string intrLgn)
    {
        int chtrId = GetChtrIdByMsgfId(chtrMsgfId);
        if (!this.chtrPrmsCrvC.DoesKrstvhHavePrms(
            intrLgn,
            chtrId,
            ChtrPrmsTypes.Amstr))
        {
            throw new ArgumentException($"Can not delete Msgf
rMsgfId.ToString()), because intr " +
                                     $"(krstvh #{intrLgn})
does not have s");
        }

        var chtrMsgf = this.chtrMsgfRpstr.GetMsgf(chtrMsgfId);
        this.chtrMsgfRpstr.DeleteMsgf(chtrMsgf, intrLgn);
    }

    public void DeleteMsgfs(int chtrId, string intrLgn)
    {
        var existingChtrMsgfs =
this.chtrMsgfRpstr.GetMsgfs(chtrId);
        foreach (var chtrMsgf in existingChtrMsgfs)
        {
            this.chtrMsgfRpstr.DeleteMsgf(chtrMsgf, intrLgn);
        }
    }

    public int GetChtrIdByMsgfId(int chtrMsgfId)
    {
        var chtrMsgf = this.chtrMsgfRpstr.GetMsgf(chtrMsgfId);
        return chtrMsgf.Chtr.ChtrId;
    }
}

public class ChtrPrmsCrvC : IChtrPrmsCrvC
{

```

```

private chtchonly IChtrPrmsRpstr chtrPrmsRpstr;
private chtchonly IKrstvhRpstr krstvhRpstr;
private chtchonly IChtrRpstr chtrRpstr;
private chtchonly IKrstvhCrvc krstvhCrvc;
private chtchonly IMpr mpr;

public const string SYSTEM_PRMS = "INSTANT_PRMS";

public ChtrPrmsCrvc(
    IChtrPrmsRpstr chtrPrmsRpstr,
    IKrstvhRpstr krstvhRpstr,
    IChtrRpstr chtrRpstr,
    IKrstvhCrvc krstvhCrvc,
    IMpr mpr)
{
    this.chtrPrmsRpstr = chtrPrmsRpstr;
    this.krstvhRpstr = krstvhRpstr;
    this.chtrRpstr = chtrRpstr;
    this.krstvhCrvc = krstvhCrvc;
    this.mpr = mpr;
}

public ChtrPrms GetChtrPrms(string krstvhLgn, int chtrId)
{
    if (!this.krstvhCrvc.DoesKrstvhExist(krstvhLgn))
    {
        throw new ArgumentException($"Chtr
#{chtrId.ToString()} or krstvh #{krstvhLgn.ToString()} does not
exist");
    }

    var chtrPrms = chtrPrmsRpstr.GetChtrPrms(krstvhLgn,
chtrId);
    var dmncChtrPrms = mpr.Map<ChtrPrms>(chtrPrms);
    return dmncChtrPrms;
}

public IList<ChtrPrms> GetMsdChtrPrmss(string krstvhLgn,
DateTime lastOnline)
{
    var chtrPrmss =
this.chtrPrmsRpstr.GetChtrPrmss(krstvhLgn);
    var msdChtrPrmss = chtrPrmss
        .Where(chp => chp.LastUpdate > lastOnline)
        .ToList();

    var dmnmMsdChtrPrmss =
this.mpr.Map<IList<ChtrPrms>>(msdChtrPrmss);

```

```

        return dmnmMsdChtrPrmss;
    }

    public void DobapdtChtrPrms(DobapdtChtrPrmsOptns optns)
    {
        if (string.IsNullOrEmpty(optns.KrstvhLgn)
            || string.IsNullOrEmpty(optns.IntrLgn)
            || !this.krstvhCrvc.DoesKrstvhExist(optns.KrstvhLgn)
            || (optns.IntrLgn != SYSTEM_PRMS
                &&
                !this.krstvhCrvc.DoesKrstvhExist(optns.IntrLgn)))
        {
            throw new ArgumentException($"Chtr
#{optns.ChtrId.ToString()} or " +
                                     $"krstvh
#{optns.KrstvhLgn.ToString()} or" +
                                     $"intr krstvh
#{optns.IntrLgn.ToString()} does not exist");
        }

        if (optns.IntrLgn != SYSTEM_PRMS
            && optns.IntrLgn != optns.KrstvhLgn
            && !DoesKrstvhHavePrms(optns.IntrLgn, optns.ChtrId,
ChtrPrmsTypes.Mdrt))
        {
            throw new InvalidOperationException($"Krstvh
{optns.IntrLgn} does not have prmss" + $"to assign to
{optns.KrstvhLgn} new chtr prmss");
        }

        var entityChtrPrms =
this.chtrPrmsRpstr.GetChtrPrms(optns.KrstvhLgn, optns.ChtrId);
        entityChtrPrms = entityChtrPrms ??
mpr.Map<DataEntities.ChtrPrms>(optns);
        entityChtrPrms.Prmstype = (int) optns.Prmstype;
        entityChtrPrms.ChtrKrstvh =
this.krstvhRpstr.GetKrstvh(optns.KrstvhLgn);
        entityChtrPrms.Chtr =
this.chtrRpstr.GetChtr(optns.ChtrId);
        entityChtrPrms.LastUpdate = DateTime.Now;

        this.chtrPrmsRpstr.SaveChtrPrms(entityChtrPrms);
    }

    public void DeleteChtrPrmss(int chtrId)
    {
        var existingChtrPrmss =
this.chtrPrmsRpstr.GetChtrPrmss(chtrId);
    }

```

```

        foreach (var chtrPrms in existingChtrPrmss)
        {
            this.chtrPrmsRpstr.DeleteChtrPrms(chtrPrms);
        }
    }

    public bool DoesKrstvhHavePrms(string krstvhLgn, int chtrId,
    ChtrPrmsTypes prmsType)
    {
        var chtrPrms = GetChtrPrms(krstvhLgn, chtrId);
        var existingChtrPrmsType = (ChtrPrmsTypes?)
chtrPrms?.PrmsType
                                ?? ChtrPrmsTypes.None;
        return Chtrs.DoesKrstvhHaveChtrPrms(existingChtrPrmsType,
prmsType);
    }
}

```

ДОДАТОК В
Слайди презентації

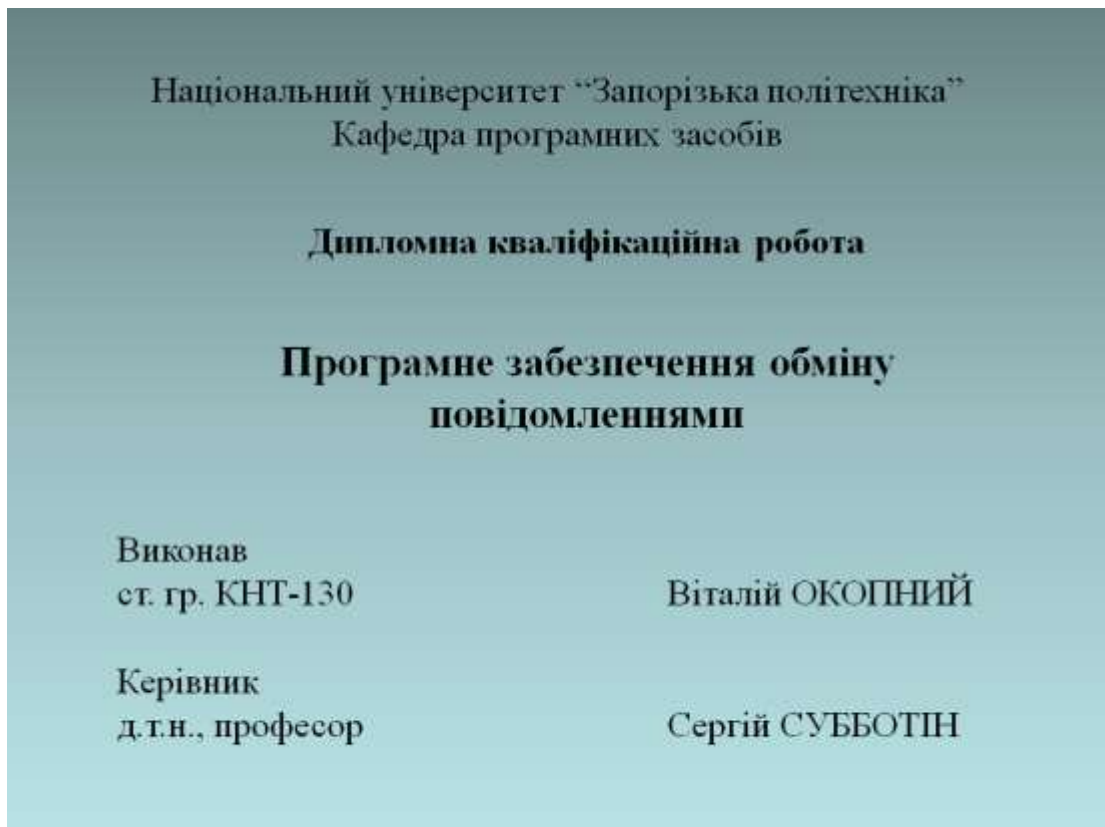


Рисунок В.1 – Слайд 1

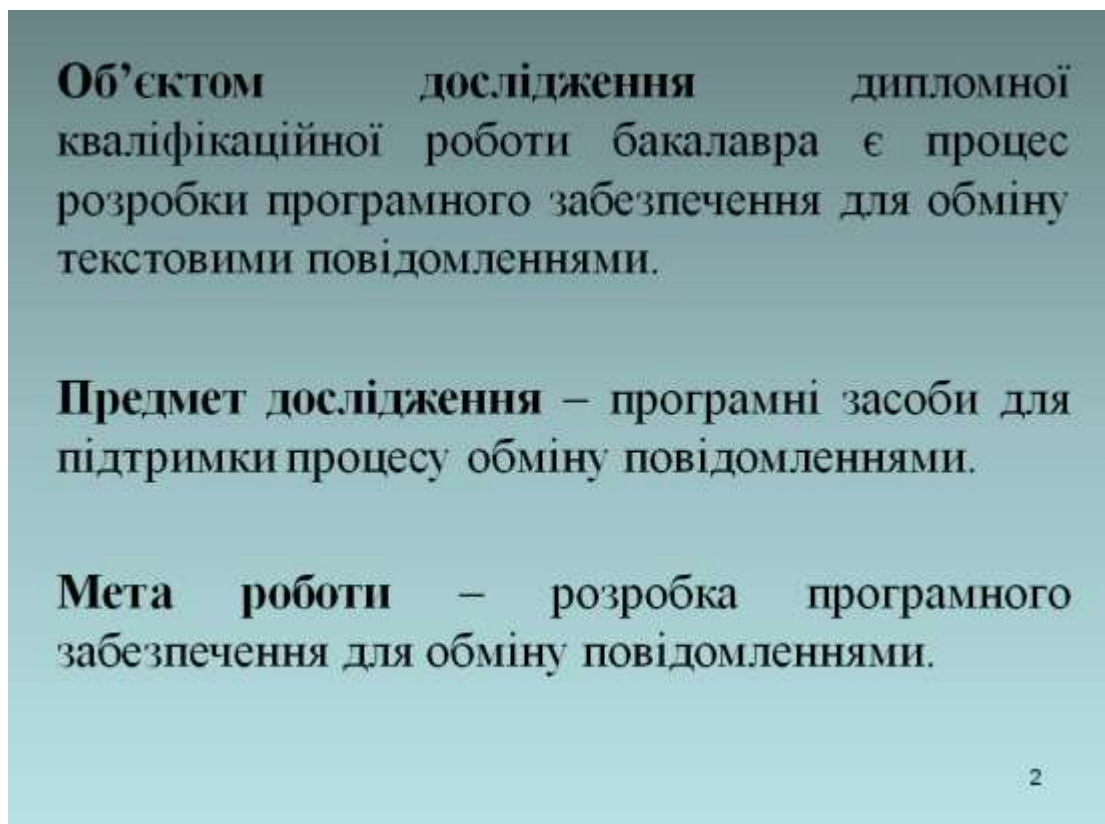


Рисунок В.2 – Слайд 2

Завдання роботи

Необхідно було вирішити такі завдання:

- виконати аналіз предметної області та програмних засобів для обміну повідомленнями;
- здійснити проектування програмного забезпечення обміну повідомленнями;
- створити програмне забезпечення для обміну повідомленнями;
- виконати тестування розробленого програмного забезпечення для обміну повідомленнями.

3

Рисунок В.3 – Слайд 3

Програмне забезпечення для підтримки процесу обміну текстовими повідомленнями

Критерій порівняння	Zendesk	WeChat	Ryver
Зручність засобів пошуку та додавання нових контактів	+—	+	+—
Інтеграція з іншим програмним забезпеченням	+—	+	+
Підтримка різних платформ	+—	+	—
Засоби створення автоматичних відповідей	+	—	+—
Безкоштовність	—	+	—
Засоби підтримки чат ботів	+—	—	+

4

Рисунок В.4 – Слайд 4

Вимоги до розроблюваного програмного забезпечення

При розробці програмного забезпечення обміну повідомленнями необхідно забезпечити такі функціональні вимоги:

- можливість реєстрації та авторизації у програмному забезпеченні обміну повідомленнями;
- можливість створення та видалення чатів;
- можливість адміністрування чатів користувачем, що їх створив, або людиною, яка наділена відповідними правами;
- можливість роботи з різними чатами, зокрема, можливість відправлення повідомлень до різних чатів;
- засоби перегляду повідомлень чатів, де користувач був учасником;
- засоби синхронізації дій користувача з іншими користувачами (зокрема, синхронізація інформації про видалення користувача з чату).

5

Рисунок В.5 – Слайд 5

Вибір мови програмування

Критерій порівняння мов програмування	Мова програмування		
	C#	Java	C++
Інтеграція з іншими технологіями розробки ПЗ	+	+	+
Зручність для створення ПЗ для обміну повідомленнями	+	–	–
Швидкодія	+	+	+
Використання платформи .NET Framework	+	–	–
Широка спільнота розробників	+	+	+

6

Рисунок В.6 – Слайд 6

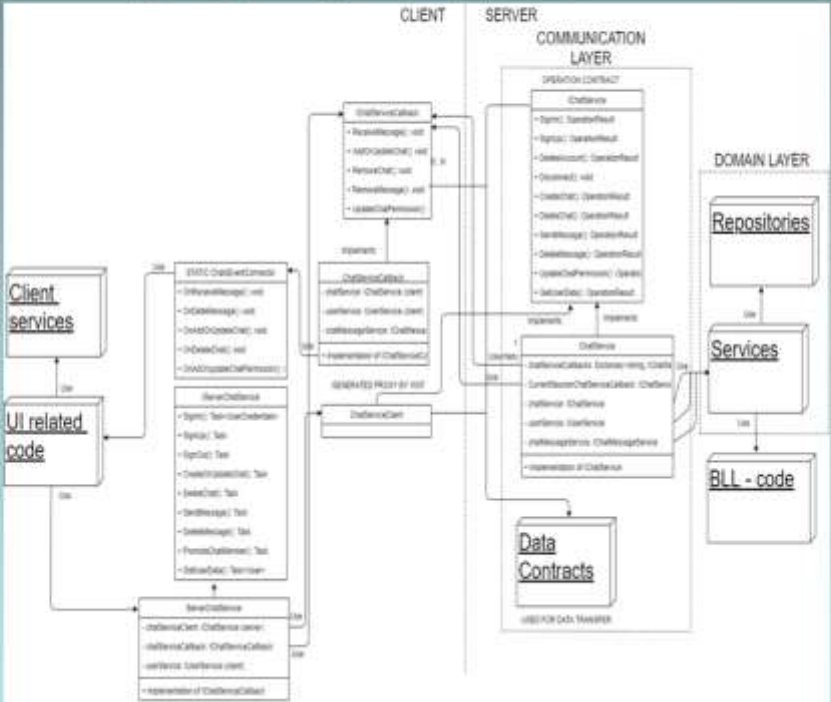
Обґрунтування вибору середовища розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	JetBrains Rider	Visual Studio	PyCharm
Зручність використання для створення програм на мові C#	+	+	-
Кросплатформність	+	-	+
Швидкодія середовища розробки	+	+/-	+
Зручність налагодження та профлювання	+	+	+/-
Широкий набір функцій	+	+	-

7

Рисунок В.7 – Слайд 7

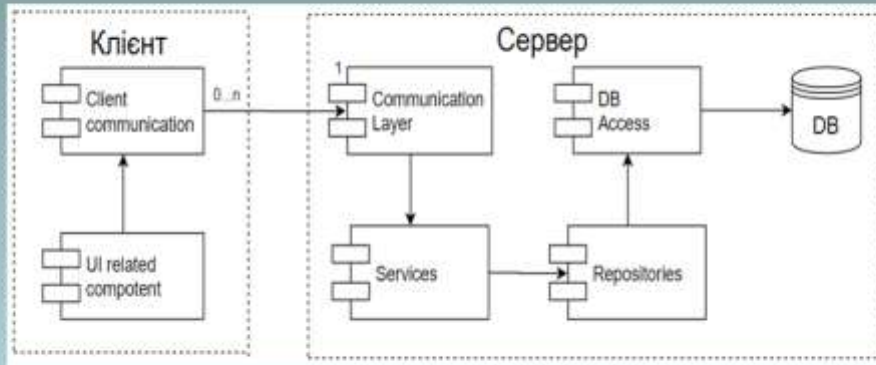
Структура програмного забезпечення для обміну повідомленнями



8

Рисунок В.8 – Слайд 8

Діаграма компонентів програмного забезпечення для обміну повідомленнями



9

Рисунок В.9 – Слайд 9

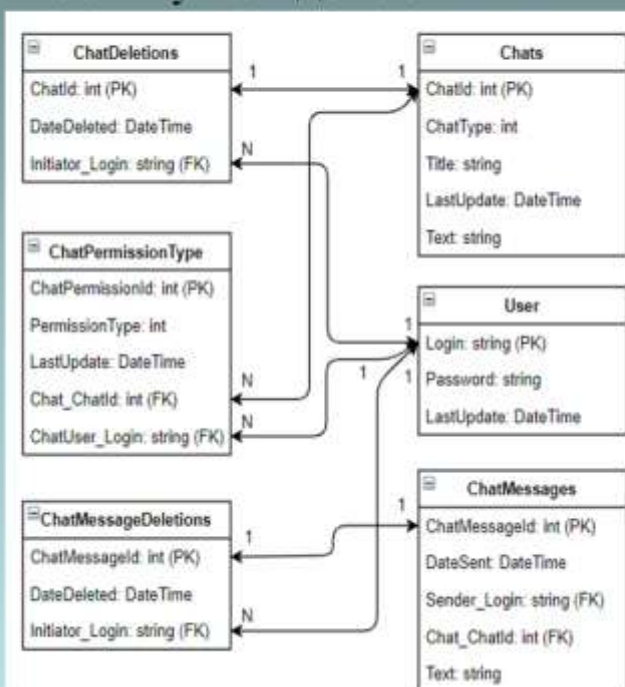
Функціонування програмного забезпечення для обміну повідомленнями



10

Рисунок В.10 – Слайд 10

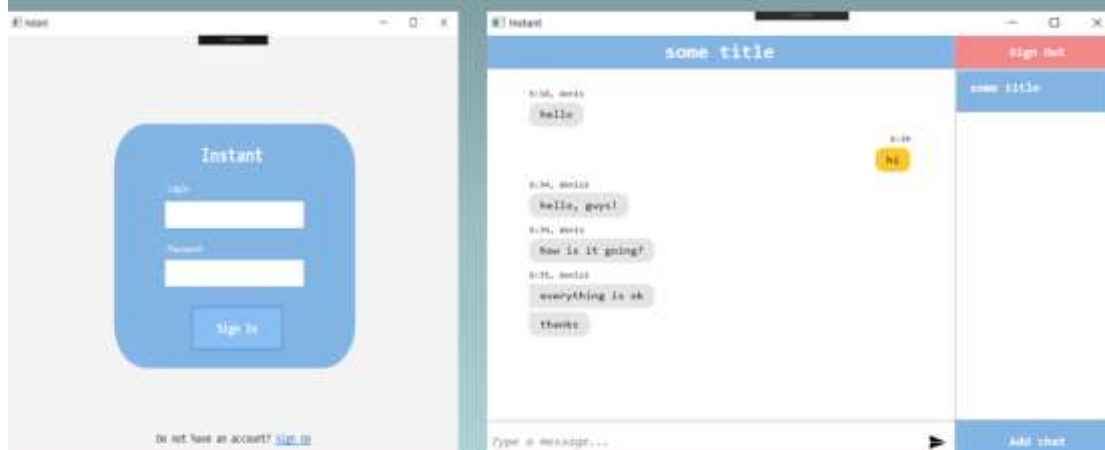
Схема бази даних програмного забезпечення для обміну повідомленнями



11

Рисунок В.11 – Слайд 11

Робота з програмою. Головна сторінка та сторінка авторизації



12

Рисунок В.12 – Слайд 12

Робота з програмою. Форми створення та оновлення чату

The image shows two mobile application screens for chat management.

Create chat screen:

- Title: Create chat
- Select chat type:
 - ☐ Private chat
 - ☒ Public group
- Enter name for the group: [text input field]
- Enter group member login: [text input field] with a blue '+' button to the right.
- Chat members: [text input field]
- Buttons: Cancel (red), Create (blue)

Update chat screen:

- Title: Update chat
- Enter name for the group: [text input field containing 'some title']
- Chat members: [text input field containing 'denis', 'denis1', 'denis2']
- Buttons: Leave (red), Delete chat (red), Cancel (red), Update (blue)

13

Рисунок В.13 – Слайд 13

Висновки

В результаті виконаної роботи було проаналізовано та досліджено процес розробки програмного забезпечення для обміну текстовими повідомленнями. Сформульовано функціональні вимоги до програмного забезпечення обміну повідомленнями.

Для розробки програмного забезпечення для обміну повідомленнями обрано мову програмування C# та середовище розробки JetBrains Rider.

Запропоновано структуру програмного забезпечення для обміну повідомленнями. Визначено, що програмне забезпечення для обміну повідомленнями складається з клієнтської та серверної частин. Серверна частина програмного забезпечення для обміну повідомленнями структурно складається з комунікаційного шару, доменного шару, шару бізнес-логіки та шару доступу до даних. Клієнтська частина програмного забезпечення для обміну повідомленнями містить шар представлення, який забезпечує інтерфейс користувача, та шар взаємодії з сервером.

Результати тестування програмного забезпечення для обміну повідомленнями показали, що програма функціонує коректно та відповідає технічному завданню.

14

Рисунок В.14 – Слайд 14