Київський національний торговельно-економічний університет

Кафедра інженерії програмного забезпечення та кібербезпеки

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

НА ТЕМУ:

Розробка клієнт-серверного мережевого чату

Студента 4 курсу, 6 групи, спеціальності 121 «Інженерія програмного забезпечення» _____ підпис студента — Бондар Володимир Олександрович

Науковий керівник кандидат технічних наук, доцент _____ підпис керівника — Харченко Олександр Анатолійович

Гарант освітньої програми кандидат технічних наук, доцент _____ підпис керівника — Цензура Микола Олександрович

КИЇВ – 2021

Київський національний торговельно-економічний університет

Факультет інформаційних технологій Кафедра _____
_____
Освітній ступінь_____
Спеціальність _____
Спеціалізація / освітня програма _____

Затверджую
Зав. кафедри інженерії
програмного забезпечення
та кібербезпеки
Криворучко О.В.
"20" жовтня 2020 р.

Завдання
на випускна кваліфікаційна робота студентові

Бондарю Володимиру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи _____
_____ Розробка клієнт-серверного мережевого чату_____
_____
Затверджена наказом ректора від "30" жовтня 2020 р. № 3225
2. Строк здачі студентом закінченої роботи _____
_____
3. Цільова установка та вихідні дані до роботи
Мета роботи - аналіз та обґрунтування підходів до проектування та створення мережевого чату.
Об'єкт дослідження – пошук оптимальних рішень для вирішення поставленої задачі._____
_____
_____
Предмет дослідження - можливості застосування засобів проектування та розробки програмного продукту._____
_____
_____

4. Консультанти роботи із зазначенням розділів, які консультують:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв

5. Зміст випускної кваліфікаційної роботи (перелік питань за кожним розділом)

ВСТУП, РОЗДІЛ 1 ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ДЛЯ СТВОРЕННЯ ТА АНАЛІЗУ ЧАТУ, Історія розвитку, TCP / IP та клієнт-серверна архітектура, Мета розробки мережевого чату, Середовище розробки мережевого чату, Висновок до розділу 1, РОЗДІЛ 2 ОСНОВНІ ЕТАПИ ПРОЕКТУВАННЯ ТА СТВОРЕННЯ МЕРЕЖЕВОГО ЧАТУ, Загальні відомості про програмний продукт, Призначення і область застосування, Функціональне призначення, Опис логічної структури, Структура проекту, Технічні та програмні засоби які були використані, Створення інтерфейсу користувача за допомогою XAML, Висновок до розділу 2, РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОЕКТУ «HOUSE CHAT», Загальні відомості про програму, Характеристика програми, Встановлення програми, Запуск програми, Клієнт «Chat.Client.exe», Процес обміну ключовою інформацією, Обмеження на використання програми, Опис головного інтерфейсу програми та робота з нею, Результат роботи програми, Код програми «House Chat», Висновок до розділу 3, ВИСНОВКИ, СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ, Електронні ресурси.

6. Календарний план виконання роботи

№ пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		за планом	фактично
1	2	3	4
1.	*Вибір теми випускної кваліфікаційної роботи*	*21.09.2020*	*21.09.2020*
2.	*Вступ та перелік літературних джерел*	*14.12.2020*	*14.12.2020*
3.	*Розділ 1. «ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ДЛЯ СТВОРЕННЯ ТА АНАЛІЗУ ЧАТУ»*	*19.02.2021*	*19.02.2021*
4.	*Розділ 2. «ОСНОВНІ ЕТАПИ ПРОЕКТУВАННЯ ТА СТВОРЕННЯ МЕРЕЖЕВОГО ЧАТУ»*	*05.03.2021*	*05.03.2021*
5.	*Розділ 3. «РЕАЛІЗАЦІЯ ПРОЕКТУ "HOUSE CHAT"»*	*10.04.2021*	*10.04.2021*
6.	*Висновки*	*24.04.2021*	*24.04.2021*
7.	*Здача випускної кваліфікаційної роботи на кафедру (перша перевірка)*	*14.05.2021*	*14.05.2021*
8.	*Підготовка автореферату та презентації доповіді*	*17.05.2021*	*17.05.2021*
9.	*Попередній захист випускної кваліфікаційної роботи*	*18.05.2021 - 21.05.2021*	
10.	*Зовнішнє рецензування випускної кваліфікаційної роботи*	*01.06.2021*	
11.	*Здача прошитого випускної кваліфікаційної роботи на кафедру*	*02.06.2021*	
12.	*Публічний захист випускної кваліфікаційної роботи*	*за розкладом роботи ЕК*	

7. Дата видачі завдання «20» жовтня 2020 р.

8. Науковий керівник випускної кваліфікаційної роботи

Харченко О.А.

(прізвище, ініціали, підпис)

9. Гарант освітньої програми ____Цензура М.О._____

(прізвище, ініціали, підпис)

10. Завдання прийняв до виконання студент ___Бондар В.О._____

(прізвище, ініціали, підпис)

11. Відгук керівника випускної кваліфікаційної роботи

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Науковий керівник випускної кваліфікаційної роботи _____

(підпис, дата)

Відмітка про попередній захист _____

(ПІБ, підпис, дата)

12. Висновок про випускну кваліфікаційну робота

Випускна кваліфікаційна робота студента _____

(прізвище, ініціали)

може бути допущена до захисту екзаменаційній комісії.

Гарант освітньої програми _____

(прізвище, ініціали, підпис)

Завідувач кафедри _____

(підпис, прізвище, ініціали)

«____»_____20 ___ р.

Анотація

Дипломна робота на тему «Розробка клієнт-серверного мережевого чату» містить 41 сторінку та 14 рисунків. Перелік списку використаних джерел налічує 11 посилань.

Метою роботи є аналіз та обґрунтування підходів до проектування та створення мережевого чату.

Під **об'єктом дослідження випускної кваліфікаційної роботи** розуміємо задачі пошуку оптимальних рішень.

Предметом дослідження випускної кваліфікаційної роботи постають можливості застосування засобів проектування та розробки програмного продукту.

В першому розділі були враховані вимоги до повноцінного локального чату. Ці вимоги оптимально націлені на досягнення максимального задоволення потреб користувача. Надійність програми в цілому залежить від надійності операційної системи. Для забезпечення стійкого стану програмного засобу необхідно перевіряти коректність вхідних і вихідних даних.

У другому розділі визначено основне завдання даного проекту – забезпечити миттєвий та безпечний зв'язок двох і декількох людей одночасно, шляхом передачі невеликої кількості інформації безпосередньо один одному.

У третьому розділі були описані загальні відомості про програму, характеристика програми, головний інтерфейс, представлений код, а також було продемонстровано роботу програми.

Annotation

Thesis on the topic "Development of client-server network chat" contains 41 pages and 14 figures. The list of used sources includes 11 links.

The aim of the work is to analyze and substantiate approaches to the design and creation of network chat.

Under **the object of study of the final qualifying work** we understand the task of finding optimal solutions.

The subject of the final qualification work is the possibility of using tools for designing and developing a software product.

The first section took into account the requirements for a full-fledged local chat. These requirements are optimally aimed at achieving maximum satisfaction of user needs. The reliability of the program in general depends on the reliability of the operating system. To ensure the stable state of the software, it is necessary to check the correctness of the input and output data.

The second section defines the main task of this project - to provide instant and secure communication between two and more people at the same time, by transmitting a small amount of information directly to each other.

The third section described the general information about the program, the characteristics of the program, the main interface, the presented code, and also demonstrated the operation of the program.

ЗМІСТ

ВСТУП 4
РОЗДІЛ 1 6
ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ДЛЯ СТВОРЕННЯ ТА АНАЛІЗУ ЧАТУ 6
1.1 Історія розвитку 6
1.2 TCP / IP та клієнт-серверна архітектура 8
1.3 Мета розробки мережевого чату 10
1.4 Середовище розробки мережевого чату 11
Висновок до розділу 1 15
РОЗДІЛ 2 16
ОСНОВНІ ЕТАПИ ПРОЕКТУВАННЯ ТА СТВОРЕННЯ МЕРЕЖЕВОГО ЧАТУ 16
2.1 Загальні відомості про програмний продукт 16
2.2 Призначення і область застосування 16
2.3 Функціональне призначення 16
2.4 Опис логічної структури 17
2.5 Структура проекту 18
2.6 Технічні та програмні засоби які були використані 18
2.7 Створення інтерфейсу користувача за допомогою XAML 20
Висновок до розділу 2 23
РОЗДІЛ 3 24
РЕАЛІЗАЦІЯ ПРОЕКТУ «HOUSE CHAT» 24
3.1 Загальні відомості про програму 24
3.2 Характеристика програми 24
3.3 Встановлення програми 24
3.4 Запуск програми 25
3.5 Клієнт «Chat.Client.exe» 28
3.6 Процес обміну ключовою інформацією 28
3.7. Обмеження на використання програми 28

3.8 Опис головного інтерфейсу програми та робота з нею 29
3.9 Результат роботи програми 30
3.10 Код програми «House Chat» 31
Висновок до розділу 3 39
ВИСНОВКИ 40
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 41
Електронні ресурси 41

ВСТУП

Розвиток інформаційних технологій та елементів штучного інтелекту допомагають розвитку суспільства в різних сферах, а саме: власникам бізнесу надає різні можливі варіанти виходу на ринок, незалежно від того, в якій сфері вони працюють; для страхування – використання швидких та простих сценаріїв, як подача страхового позову, претензії, чи запит на обслуговування, дозволить залучати все нових клієнтів за рахунок підвищення якості обслуговування поточних; стосовно комунальних послуг – при виникненні надзвичайних ситуацій та аварій розсилати повідомлення про проблеми і терміни їх вирішення, або запит рахунку за послуги та можливість його оплати безпосередньо в чаті, перевірка заборгованості, або ж заявка на ремонт та в інших сферах нашого життя. Мережевий чат є одним з найбільш перспективних нововведень. У користувачів відпадає необхідність дзвонити в службу підтримки для вирішення питань, більшість з яких є однотипними. У випадках, коли компанія НР потребує персоналізованої допомоги, чат-бот може переключити його на оператора колл-центру. Такий підхід не тільки прискорює процес взаємодії з клієнтом, але і підвищує ефективність бізнесу.

Багато транснаціональних компаній уже використовують цю технологію для продажів, збору інформації та зворотного зв'язку зі споживачами. Однак отримати вигоду від таких впроваджень може і малий бізнес. Брак часу, мала кількість співробітників, слабо розвинені технічні потужності – все це позначається на якості і ефективності взаємодії з клієнтами. Мережеві чати дозволяють уникнути цих проблем і сформувати лояльне ставлення до бренду підприємства.

Огляд останніх джерел досліджень і публікацій. За останні роки з'явилася велика кількість наукових робіт, у яких розглядаються можливості використання

мережевих чатів та чат-ботів. Відомості щодо теоретичного й практичного застосування чатів, доцільність використовувати перевірені методи їх проектування, можна знайти в працях О.І. Провотаря, Х.А. Клочко, Л.В. Ясницкого та ін.

Постановка завдання. Завдання роботи полягає у вирішенні питання проектування та застосування мережевого чату, вибору методів та інструментів для оптимізації розміщення в будь-якій мережі.

Метою роботи є аналіз та обґрунтування підходів до проектування та створення мережевого чату.

Під ***об'єктом дослідження випускної кваліфікаційної роботи*** розуміємо задачі пошуку оптимальних рішень.

Предметом дослідження випускної кваліфікаційної роботи постають можливості застосування засобів проектування та розробки програмного продукту.

Основними ***завданнями є*** проектування та розробка чату з можливістю підключення до будь-якої мережі.

Методи, які застосовано для розв'язання задач: основи теорії графів, системний підхід для аналізу інформації, метод параметричної оптимізації для оптимізації алгоритму, структурно-функціональний метод для здійснення постановки задачі.

Проблема пошуку оптимального рішення постає базовою у процесі розв'язання оптимізаційних задач у різних галузях людської діяльності: економіці, логіці, техніці, медицині, мережевих технологіях тощо. Розв'язок таких задач часто можна формалізувати до опису математичної моделі критеріїв оптимальності засобами теорії графів та математичної статистики і розглядати питання оптимізації як алгоритмізований пошук найкоротших шляхів між вершинами графа з урахуванням фізичного значення його вершин та дуг. Тому актуальною є розробка програмного продукту розв'язання оптимізаційних задач.

РОЗДІЛ 1
ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ДЛЯ СТВОРЕННЯ ТА АНАЛІЗУ ЧАТУ

1.1 Історія розвитку

Сучасна інформаційна ера – це новий крок до технологічного розвитку людства, який стрімко розвивається кожну секунду нашого буття. Технологічний, а також інформаційний прогрес, особливо за останні роки, зробив величезний крок вперед. Сучасне людство швидко перетворюється в інформаційне суспільство, а особливо це відбувається в країнах, що швидко розвиваються, які роблять великий акцент саме на розвиток техніки та інформаційних технологій.

У багатьох людей є настільні персональні комп'ютери або ноутбуки, з доступом до всесвітньої мережі Інтернет з величезною кількістю розважальних і цікавих ресурсів, або до локальної мережі, яка наповнена різними розважальними сервісами.

Відповідно, люди почали знайомитись і спілкуватись на відстані, щоб полегшити це завдання, в наш час існує дуже багато рішень, дана випускна кваліфікаційна робота була присвячена розробці засобу спілкування між людьми.

З розвитком інформаційних технологій стали можливими глобальні комунікації. Історичним попередником чатів, безсумнівно, був телефон. Ні пошта, ні телеграф не дозволяли спілкуватися в реальному часі, і не були доступні в домашніх умовах. Винахід і поширення телефону по планеті викликало справжню революцію в засобах і способах спілкування. Можливість поговорити зі співрозмовником на іншій стороні Землі здавалася справжнім дивом.

У другій половині XX століття почали бурхливо розвиватися комп'ютери. Однак довгий час вони були великими і дуже дорогими, що перешкоджало, щоб витрачати дорогоцінний машинний час на забави з обміном повідомлень замість

розрахунків атомних бомб. До того ж до кінця 60-х років вони не були пов'язані один з одним. Предок Інтернету, мережа ARPANET, в 1969 році налічувала лише чотири пов'язаних один з одним наукових комп'ютера. Трохи пізніше, в 1971 році, була придумана електронна пошта, яка стала надзвичайно популярна через свою зручність. Поступово з'явилися нові служби повідомлень, такі, як списки поштової розсилки, групи новин та дошки оголошень. Однак у той час мережа ARPANET ще не могла легко взаємодіяти з іншими мережами, побудованими на інших технічних стандартах, що ускладнювало її поширення. Але, тим не менш, ця проблема незабаром була вирішена після переходу мереж на протокол обміну даними TCP / IP, який успішно застосовується до цих пір. Саме в 1983 році термін «Інтернет» закріпився за мережею ARPANET.

Програми для обміну текстовими рядками, незважаючи на простоту самої ідеї, з'явилися не відразу. Приблизно в 1974 році для мейнфрейма PLATO була розроблена програма Talkomatic, яка потенційно дозволяла спілкуватися між тисячами терміналів системи. У 1980-х з'явилася система Freelancing Round table. Однак по-справжньому популярним став розроблений в 1988 році протокол, названий Internet Relay Chat (IRC), що приблизно можна перекласти як «ретранслюєма інтернет-розмова». Десь в цей же час з'явилося і поширилося саме поняття «чат». Спілкування в IRC швидко стало популярним через простоту процесу і дружності середовища. У 1991 році під час операції «Буря в пустелі» була організована IRC-трансляція новин – повідомлення з усього світу збиралися в одному місці і в режимі реального часу передавалися в IRC. Є відомості, що подібним чином IRC використовувався і під час спроби здійснити державний переворот в СРСР, коли користувачі з Москви моментально повідомляли всьому світу про те, що відбувається на вулицях міста. Для клієнтів IRC було написано безліч ботів, що автоматизують багато рутинних операцій. Найвідомішим з клієнтів IRC став mIRC; завдяки простій і ефективній системі команд для нього було написано безліч скриптів, які також дозволяють виконувати широкий спектр дій. Боти і mIRC-боти використовуються для різних ігор в таких каналах як: «Мафія», «Вікторина» та інших.

Згодом, крім звичайних текстових чатів, були вигадані відео та голосові чати. Відеочати – це обмін текстовими повідомленнями з можливим транслюванням зображення з веб-камери. Спочатку це були не відео, а скоріше, фоточати через низьку пропускну спроможність каналів відправлявся не відеопотік, а картинка з деякими інтервалами, що однак, давало можливість досить оперативно спостерігати зміну емоцій у співрозмовника і було значним проривом. Пізніше став транслюватися відеопотік, хоча і з меншою роздільною здатністю. Веб-камери є простими та дешевими, хоча зворотній бік цього – низька роздільна здатність відео і його погана якість. Зображення виходить зашумлене та з поганою передачею кольору. Однак для спілкування, такої якості більш ніж достатньо.

Голосові чати теж з'явилися з розвитком ідей обміну повідомленнями. В даний час в комп'ютерних іграх широко застосовується система TeamSpeak та Discord, що дозволяє спілкуватися голосом між членами команди, не відволікаючись від управління грою. А спілкування по Viber, Telegram, WhatsApp, Skype більше нагадує розмову по телефону, ніж чат, хоча можливість відправки звичайних текстових повідомлень в ньому теж присутня.

Дана робота присвячена розробці та створенню зручного, корисного і безпечного в першу чергу для користувача додатку з можливістю мережевого спілкування.

1.2 TCP / IP та клієнт-серверна архітектура

Використовуючи мережу Internet, ми дуже часто зустрічаємо різні програми та онлайн сервіси. Ними можуть бути поштові клієнти, чати, форуми, FTP клієнти. Всі ці програми використовують для своєї роботи різного роду протоколи: FTP, POP, SMTP, HTTP. Але базовим для них є єдиний протокол - TCP / IP.

- TCP / IP – набір (стек) протоколів передачі даних. TCP / IP – це позначення всієї мережі, яка працює на основі двох протоколів – TCP і IP.
- TCP (Transfer Control Protocol) – протокол, який служить для встановлення надійного з'єднання між двома пристроями, передачі інформації і підтвердження її отримання

– IP (Internet Protocol) – Інтернет протокол, який відповідає за правильність доставки повідомлень за певною адресою. При цьому дані розбиваються на пакети, які можуть доставлятися по-різному.

В основі клієнт-серверної архітектури лежать два компоненти: клієнт і сервер.

Клієнт – комп'ютер на стороні користувача, який відправляє запит до сервера для надання інформації або виконання певних дій.

Сервер – більш потужний комп'ютер або обладнання, призначене для вирішення певних завдань з виконання програмних кодів, виконання сервісних функцій за запитом клієнтів, надання користувачам доступу до певних ресурсів.

Модель такої системи полягає в тому, що клієнт відправляє запит на сервер, де він обробляється, і готовий результат відправляється клієнтові. Сервер може обслуговувати кілька клієнтів одночасно. Якщо одночасно приходить більше одного запиту, то вони встановлюються в чергу і виконуються сервером послідовно. Іноді запити можуть мати пріоритети. Запити з більш високими пріоритетами повинні виконуватися раніше (рис.1.1).

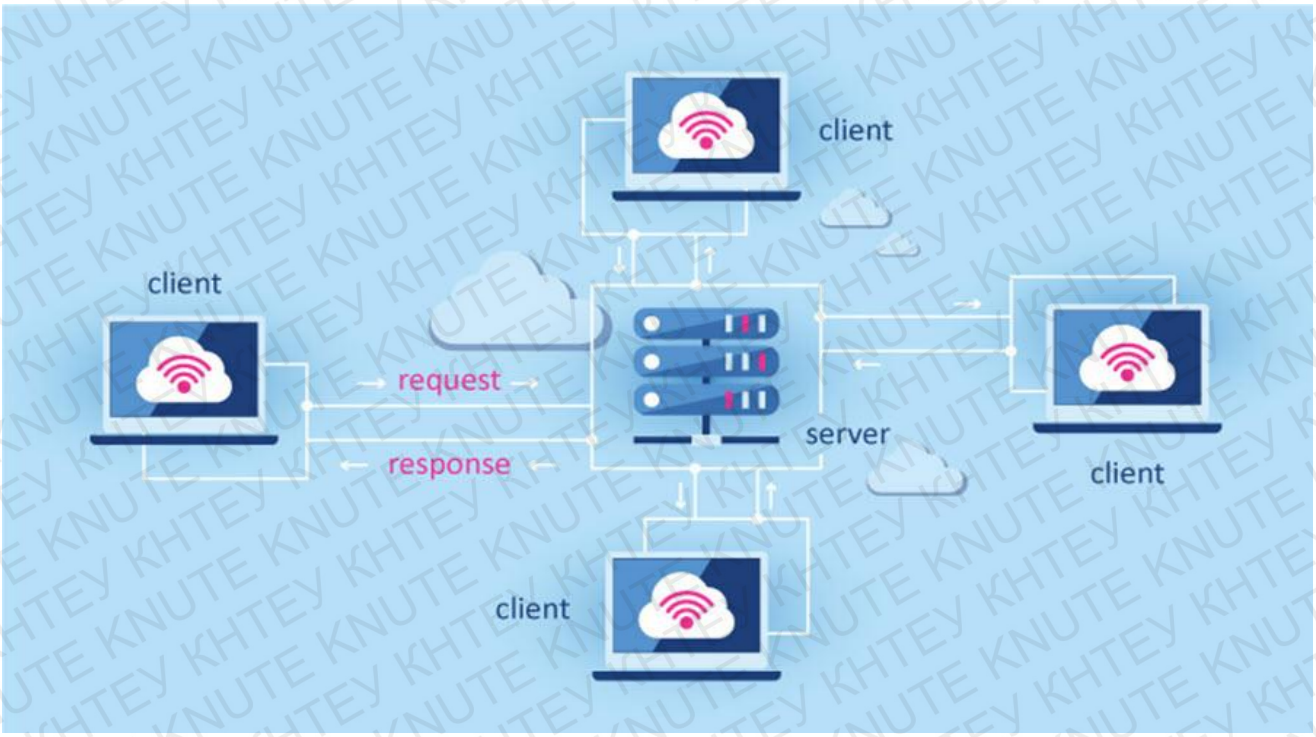

Рис.1.1 Клієнт-серверна архітектура

Функції, які реалізуються на сервері:

- обробка клієнтського запиту;
- відправка результату (відповіді) клієнту.

Функції, які реалізуються на стороні клієнта:

- відображення інтерфейсу користувача;
- формулювання запиту до сервера і його відправка;
- отримання результатів запиту і відправка додаткових команд (запитів на додавання, оновлення або видалення даних).

Чат – засіб обміну повідомленнями в комп'ютерній мережі в режимі реального часу, а також програмне забезпечення, що дозволяє організовувати таке спілкування. Характерною особливістю є комунікація саме в реальному часі або близька до цього, що відрізняє чат від форумів. Тобто, якщо на форумі можна написати питання і чекати, поки хто-небудь вважатиме за потрібне на нього відповісти (або в той же час, можна отримати кілька відповідей відразу від різних користувачів), то в чаті спілкування відбувається тільки з тими, хто присутній в ньому зараз.

Під словом чат зазвичай розуміється групове спілкування, хоча до них можна віднести і обмін текстом «один на один» за допомогою програм миттєвого обміну повідомленнями, наприклад, XMPP, ICQ або навіть SMS.

1.3 Мета розробки мережевого чату

Основною метою створення мережевого чату – є миттєвий обмін повідомленнями. Використання даного програмного продукту передбачає скоротити час виконання будь-якої роботи або просто добре провести час спілкуючись в колі друзів.

Мережевий чат може бути використаний як для малого кола людей, тобто в домашніх умовах, так і на виробництві невеликого підприємства може застосовувати виключно для важливих справ пов'язаних безпосередньо з роботою, або для розважальних цілей.

Перелік питань, що підлягають розробці у випускній кваліфікаційній роботі:

- мережевий чат повинен забезпечувати миттєвий обмін повідомленнями, бути корисним у використанні і споживати найменшу кількість ресурсів персонального комп'ютера;
- програма повинна відображати список підключених користувачів, забезпечувати стабільний потік обміну інформацією, мати налаштування приватності;
- програмний продукт повинен мати зрозумілий та лаконічний інтерфейс, що не вимагає спеціальної підготовки користувачів, повинен сприяти більш ефективному розподілу та економії часу;

1.4 Середовище розробки мережевого чату

Мова програмування C# була розроблена в 1993-2001 роках групою інженерів компанії Microsoft під керівництвом Андерса Хейлсберг і Скотта Вільтаумота, як мова розробки додатків для платформи Microsoft.

До 2000 року у Microsoft були готові промислові версії нових технологій і рішень для обміну повідомленнями та даними, а також для створення Internet-додатків. Була випущена і нова платформа для розробки під нові рішення - .NET. У ній об'єдналися відразу кілька мов програмування, що було в новинку для того часу.

Ще одним нововведенням платформи .NET була технологія активних серверних сторінок ASP.NET (Active Server Page). З її допомогою можна було відносно швидко розробити веб-додатки, які взаємодіють з базами даних. Спеціально для ASP.NET була створена мова програмування C#. Та й сама ASP.NET була повністю написана на ньому.

Порівняння з іншими мовами

C# – це мова з C-подібним синтаксисом. Тут він близький в цьому відношенні до C ++ і Java.

Об'єктно-орієнтована мова програмування C#, вона багато перейняла у Java та C ++. Як і Java, C # спочатку призначалась для веб-розробки, і приблизно 75% її синтаксичних можливостей такі, як у Java. C# також називають «очищеною версією

Java». Ще 10% C# запозичив з C ++ і 5% - з Visual Basic. Решта 10% C # – це реалізація власних ідей розробників. Об'єктно-орієнтований підхід дозволяє будувати за допомогою C # великі, але в той же час гнучкі, масштабовані і розгортаються додатки.

C# вже давно підтримує багато корисних функцій:

- інкапсуляція,
- успадкування,
- поліморфізм,
- перевантаження операторів,
- статична типізація.

При цьому вона все ще активно розвивається, і з кожною новою версією з'являється все більше цікавого – наприклад лямбда, динамічне зв'язування, асинхронні методи.

У порівнянні з іншими мовами C# досить молода, але в той же час вона вже пройшла великий шлях. Перша версія мови вийшла разом з релізом Microsoft Visual Studio .NET в лютому 2002 року. Поточною версією мови є версія C# 8.0, яка вийшла у вересні 2019 року разом з релізом .NET Core 3.

Роль платформи .NET

Коли говорять C#, нерідко мають на увазі технології платформи .NET (Windows Forms, WPF, ASP.NET, Xamarin). І навпаки, коли говорять .NET, нерідко мають на увазі C#. Однак, хоча ці поняття пов'язані, ототожнювати їх невірно. Мова C# була створена спеціально для роботи з фреймворком .NET, проте саме поняття .NET дещо ширше.

Колись Білл Гейтс сказав, що .NET – це найкраще, що зробила компанія Microsoft. У нього є вагомі підстави так вважати. Фреймворк .NET представляє потужну платформу для створення додатків. Ось його особливості:

- Підтримка декількох мов. В основі .NET – загальномовне середовище виконання Common Language Runtime (CLR), завдяки чому платформа підтримує кілька мов: поряд з C# це VB.NET, C++, F#, а також різні діалекти інших мов, прив'язані до .NET, наприклад, Delphi. NET. Код на будь-якому з

цих мов компілюється в збірку спільною мовою CIL (Common Intermediate Language) – свого роду асемблер платформи .NET. Тому можна зробити окремі модулі однієї програми на різних мовах.

- Потужна бібліотека класів. .NET представляє єдину для всіх підтримуваних мов бібліотеку класів. Який би додаток ми не збиралися писати на C# – текстовий редактор, чат або складний веб-сайт – так чи інакше ми задіємо бібліотеку класів .NET.
- Різноманітність технологій. Загальномовне середовище виконання CLR і базова бібліотека класів – це основа для цілого стека технологій, які розробники можуть задіяти при створенні різних додатків. Наприклад, для баз даних в цьому стеці є технологія ADO.NET і Entity Framework Core. Для графічних додатків з насиченим інтерфейсом – технології WPF і UWP. Для більш простих графічних додатків – Windows Forms. Для розробки мобільних додатків – Xamarin. Для створення веб-сайтів – ASP.NET.

.NET Framework і .NET Core

.NET довгий час розвивався під назвою .NET Framework – переважно як платформа для Windows. Але з 2019 вона більше не розвивається – останньою версією цієї платформи стала .NET Framework 4.8.

У 2014 Microsoft почав випускати альтернативну платформу – .NET Core, яка повинна була увібрати в себе всі можливості застарілого .NET Framework і додати нову функціональність. Тому слід розрізняти .NET Framework, який призначений переважно для Windows, і багатоплатформений.NET Core.

Переваги і недоліки мови C#

У «шарпа» виділяють багато переваг:

- підтримка переважної більшості продуктів Microsoft;
- безкоштовність ряду інструментів для невеликих компаній і деяких індивідуальних розробників – Visual Studio, хмара Azure, Windows Server, Parallels Desktop для Mac Pro

- типи даних мають фіксований розмір (32-бітний int і 64-бітний long), що підвищує «мобільність» мови і спрощує програмування, так як ви завжди знаєте точно, з чим ви маєте справу;
- автоматичне «прибирання сміття» Це означає, що нам в більшості випадків не доведеться піклуватися про звільнення пам'яті. Вищезазначене середовище CLR саме викличе збирач сміття і очистить пам'ять;
- C# має багато схожого з іншими мовами програмування, завдяки чому полегшується перехід для програмістів. Мова C# часто визнають найбільш зрозумілим і придатним для новачків;
- за допомогою Xamarin на C# можна писати програми і додатки для таких операційних систем, як iOS, Android, MacOS і Linux.

Але є у C# і деякі недоліки:

- пріоритетна орієнтованість на платформу Windows;
- мова безкоштовна тільки для невеликих фірм, індивідуальних програмістів, стартапів і учнів. Великій компанії покупка ліцензійної версії цієї мови обійдеться в кругленьку суму.

Скільки часу потрібно на навчання C#?

C# не представляє складності для новачків, так як її порівняно легко вивчити. На просторах Інтернету можна знайти безліч курсів і онлайн-шкіл з навчання C#, що пропонують навчання тривалістю від 1 до 6 місяців. Взагалі, C# набагато простіше буде освоїти, якщо ви вже знаєте C, C++ або Java.

Затребуваність на ринку

C# на протязі довгого часу впевнено тримає позиції в рейтингу найбільш затребуваних на ринку мов програмування. Спочатку нею цікавилися тільки розробники під Windows, але потім C# навчилась працювати на Mac OS, Linux, iOS і Android. А після того як код платформи відкрили для всіх бажаючих, були зняті практично всі можливі обмеження в застосуванні C#. В результаті мова активно розвивається і застосовується все ширше. Її часто рекомендують до вивчення в якості однієї з базових для розробників будь-якого профілю.

Інструментарій C# дозволяє вирішувати широке коло завдань, мова дійсно дуже потужна та універсальна. На C# часто розробляють:

- веб-додатки,
- ігри,
- мобільні додатки для Android або iOS,
- програми під Windows.

Перелік можливостей розробки практично не має обмежень завдяки найширшому набору інструментів і засобів. Звичайно, все це можна реалізувати за допомогою інших мов. Але деякі з них вузькоспеціалізовані, а в деяких доведеться використовувати додаткові інструменти сторонніх розробників. У C# вирішити широке коло завдань можливо швидше, простіше і з меншими витратами часу та ресурсів.

Висновок до розділу 1

При розробці програмного продукту House Chat були враховані вимоги до повноцінного локального чату. Ці вимоги оптимально націлені на досягнення максимального задоволення потреб користувача.

Для роботи з програмою House Chat необхідна наявність локального з'єднання. Воно необхідне для цілеспрямованого використання програмного продукту так як при його відсутності програмний продукт стає марним.

Інтерфейс програми має сприяти зниженню кількості помилок, викликаних неправильними діями користувачів системи, також система повинна здійснювати контроль даних які вводить користувач

Надійність програми в цілому залежить від надійності операційної системи. Для забезпечення стійкого стану програмного засобу необхідно перевіряти коректність вхідних і вихідних даних. Категорично забороняється видаляти або виконувати будь-які дії з файлом програми.

РОЗДІЛ 2
ОСНОВНІ ЕТАПИ ПРОЕКТУВАННЯ ТА СТВОРЕННЯ МЕРЕЖЕВОГО ЧАТУ

2.1 Загальні відомості про програмний продукт

Програма має назву «House Chat», призначена як для домашнього використання, так і для використання на малому підприємстві. Розроблено для людей які не впевнено користуються ПК, програмний продукт не вимагає ніяких інструктажів з використання і зрозумілий на інтуїтивному рівні.

2.2 Призначення і область застосування

Основним завданням даного проекту є забезпечувати миттєвий зв'язок двох і декількох людей одночасно, шляхом передачі невеликої кількості інформації безпосередньо один одному. З цією метою було створено безліч різних видів чатів, у більшості з яких основний акцент робиться на гарно оформлений інтерфейс, з яскраво вираженим текстом. В цьому випадку чат перетворюється у велику і складну програму яка вимагає максимум можливостей від ПК.

Даний чат може застосовуватися як для домашнього використання, так і для використання на малих підприємствах.

2.3 Функціональне призначення

Мережевий чат – комплекс програмного забезпечення на базі ПК, призначений для спілкування користувачів по мережі в режимі реального часу.

House Chat дозволяє виконувати такі основні завдання:

– отримувати інформацію про користувачів які одночасно використовують програму;

– переглядати всю відправлену інформацію між користувачами з моменту входу до мережі (підключення до сервера);
– забезпечувати конфіденційний обмін інформацією;
– мати доступний і зрозумілий на інтуїтивному рівні інтерфейс.

Таким чином, «House Chat» дозволяє значно економити час в порівнянні з традиційними формами обміну інформації.

2.4 Опис логічної структури

На основі вище вказаного можна виділити основні етапи розробки програмного продукту:

1. Етап проектування інтерфейсу програми;
2. Етап написання програмного коду;
3. Етап доопрацювання, внесення змін;
4. Етап формування кінцевого продукту.

На першому етапі проектування створюється загальний інтерфейс програмного продукту і створюється його функціональна підтримка.

Етап завдання навчального матеріалу містить:

– створення певного макета програми;
– створення інтерфейсу управління;
– розробку оформлення.

В результаті проектування, програмний продукт повинен підтримувати такі функції:

– введення текстової інформації та її редагування до відправки;
– можливість швидко і зручно користуватись інтерфейсом.

Етап написання програмного коду:

– вибір і аналіз методів написання алгоритму програми;
– внесення виправлення і налаштування програми.

2.5 Структура проекту

Програмне рішення базується на клієнт-серверній архітектурі, в якій:

1. RSA – бібліотека для шифрування / розшифрування даних (рис.2.1);
2. Клієнт – WPF-додаток (MVVM);
3. Сервер – консольний додаток.

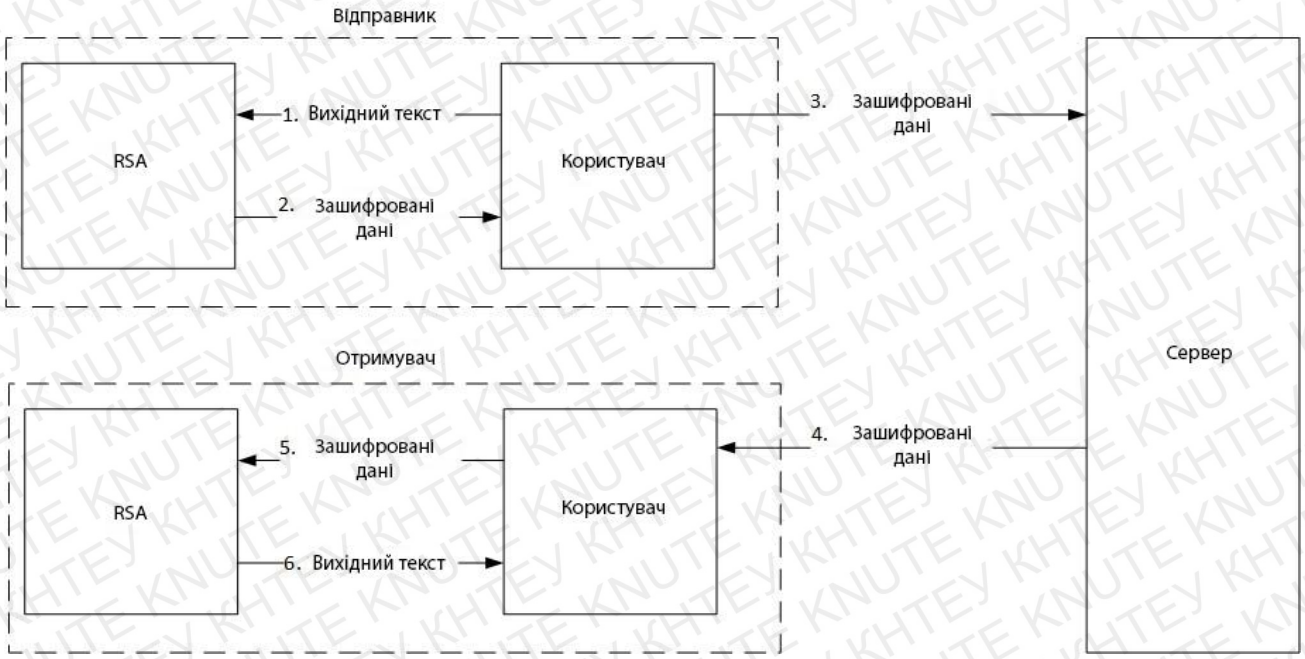

Рис.2.1 Модель роботи програмного комплексу

2.6 Технічні та програмні засоби які були використані

Програмне рішення, яке представляє собою клієнт-серверний додаток з можливістю обміну повідомленнями в зашифрованому вигляді, використовуючи RSA-шифрування з використанням об'єктно-орієнтованої мови програмування C#.
RSA – шифрування

RSA – алгоритм шифрування з відкритим ключем, який базується на факторизації простих чисел.

Серед безлічі абревіатур, пов'язаних з комп'ютерами, є окрема каста акронімів – це терміни, так чи інакше пов'язані з питаннями інформаційної безпеки.

Як у будь-якої абревіатури RSA має кілька варіантів розшифровок. RSA – це криптографічний алгоритм з відкритим ключем, назва якого походить від перших букв прізвищ його творців (Rivest, Shamir, Adleman). На сьогоднішній день RSA є

одним з найпопулярніших алгоритмів шифрування найрізноманітніших даних, крім того, його використовують для цифрового підпису різних документів.

Алгоритм шифрування – це математична процедура або набір кроків для кодування даних. RSA є найбільш широко використовуваний алгоритмом шифрування сьогодні.

RSA належить до асиметричних алгоритмів, у яких ключ шифрування не збігається з ключем дешифрування. Один з ключів доступний всім і називається відкритим ключем, інший зберігається тільки у його господаря і невідомий нікому. За допомогою одного ключа можна робити операції тільки в одну сторону. Якщо повідомлення зашифровано за допомогою одного ключа, то розшифрувати його можна тільки за допомогою іншого. Маючи один з ключів неможливо знайти інший ключ, якщо розрядність ключа висока.

В основі RSA лежить завдання факторизації відтворення двох простих великих чисел. Для шифрування використовується проста операція піднесення до степеня за модулем N. Для розшифрування необхідно обчислити функцію Ейлера від числа N, для цього необхідно знати розкладання числа n на прості множники (В цьому і полягає завдання факторизації). В RSA відкритий і закритий ключ складається з пари цілих чисел. Закритий ключ зберігається в секреті, а відкритий ключ повідомляється іншому учаснику, або десь публікується. Генерація проводиться за такою схемою

1. Вибираються два простих числа p і q (такі що p нерівно q).
2. Обчислюється модуль N = p * q.
3. Обчислюється значення функції Ейлера від модуля N: F (N) = (p-1) (q-1).
4. Вибирається число e, назване відкритою експонентою, число e повинно лежати в інтервалі 1 Обчислюється число d, назване секретною експонентою, таке, що d * e = 1 (modF (N)), тобто є мультиплікативно зворотнє до числа e по модулю F (N).

Загалом отримуємо:

Пара (e, N) – відкритий ключ.

Пара (d, N) – закритий ключ.

Мова C#

Для зручної та оптимізованої розробки додатку на об'єктно-орієнтованій мові програмування C# було використано середовище розробки Microsoft Visual Studio. Це зумовлено тим, що це середовище:

- Офіційне. Так як і мова, і середовище розробки створені в Microsoft, логічно припустити, що це середовище дуже функціональне. У деяких випадках без Visual Studio не обійтися – наприклад, при використанні технологій XAML (що використовується у даному проекті).
- Безкоштовне. Версії «Community edition» для рядового користувача буде достатньо. Тим більше, тепер можна підключати плагіни.
- Функціональне. У Visual Studio безліч якісних плагінів. З їх допомогою можна розширити функціональність програми.
- Підтримує платформи .NET. Visual Studio має широкі можливості по розробці додатків під Windows, в тому числі в .NET-сегменті.

2.7 Створення інтерфейсу користувача за допомогою XAML

Інтерфейс користувача – це сукупність засобів, за допомогою яких користувач спілкується з різними пристроями (комп’ютером або побутовою технікою) або іншим складним інструментарієм. Інтерфейс користувача – це такий різновид інтерфейсів, в якому з одного боку – людина, з іншого – машина. Інтерфейс користувача комп’ютерного додатку включає:

- засоби відображення інформації, відображувану інформацію, формати і коди;
- командні режими, мову «користувач–інтерфейс»;
- пристрої та технології введення – виведення;
- діалоги, взаємодію та транзакції між користувачем та комп’ютером, зворотній зв’язок з користувачем;
- підтримку прийняття рішень в конкретній предметній області;
- порядок використання програми і документації на неї.

Інтерфейс користувача часто розуміють лише як зовнішній вигляд програми. Однак насправді користувач сприймає через нього всю програму в цілому, тобто таке розуміння є надто вузьким.

В дійсності, інтерфейс користувача об’єднує в собі всі елементи і компоненти програми, які здатні впливати на взаємодію користувача з програмним забезпеченням (рис.2.2).

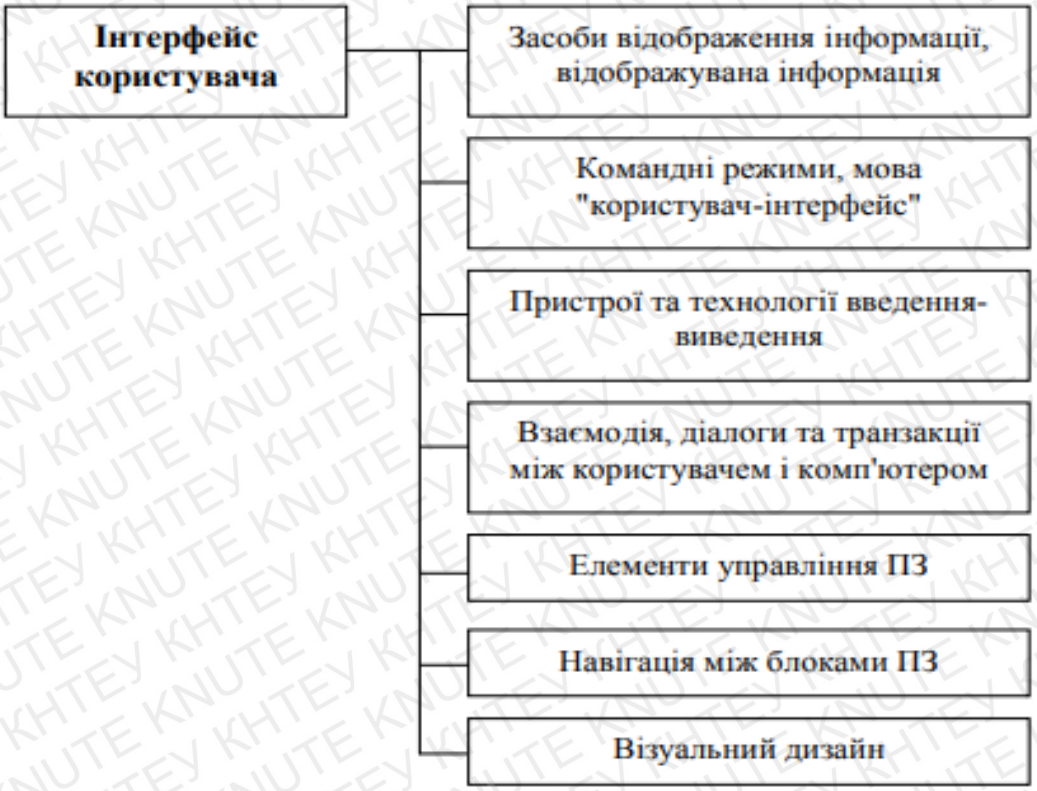

Рис.2.2 Складові інтерфейсу користувача

Процес проектування інтерфейсу користувача – це складний, нелінійний, недетермінований і неортогональний процес. Складність – звичайна властивість програмного забезпечення, але в ще більшому степені це стосується інтерфейсу – через численні фактори та невизначеності, які впливають на його розробку. Проектування інтерфейсу користувача – нелінійний процес, оскільки існування фіксованого, впорядкованого і прямолінійного шляху від початку до кінця зовсім не обов’язкове. Процес проектування відрізняється невизначеністю, оскільки не існує рівняння, за яким можна було б одержати однаковий результат при заданих

однакових початкових умовах, більше того, практично неможливо одержати ідентичний результат навіть під примусом. Інтерфейс користувача неортогональний в тому сенсі, що будь-який аспект проектного розв'язку може впливати на інші аспекти, причому результат цього впливу не завжди є приємним сюрпризом.

В даній роботі я застосував мову розмітки XAML. Мова використовується для ініціалізації об'єктів в технологіях на платформі .NET. Стосовно WPF дана мова використовується перш за все для створення призначеного для користувача інтерфейсу показовим шляхом. Хоча функціональність XAML тільки графічними інтерфейсами не обмежується: дана мова також використовується в технологіях WCF і WF, де вона ніяк не пов'язана з графічним інтерфейсом.

XAML – не є обов'язковою частиною програми, можна взагалі обійтися без неї, створивши всі елементи в Windows Forms на мові C# (рис.2.3). Однак використання XAML все-таки несе деякі переваги:

– можливість відокремити графічний інтерфейс від програмного коду, завдяки чому над різними частинами додатку можуть працювати різні фахівці: над інтерфейсом – дизайнери, а над програмним кодом – програмісти;
– компактність, зрозумілість, код на XAML відносно легко редагувати.

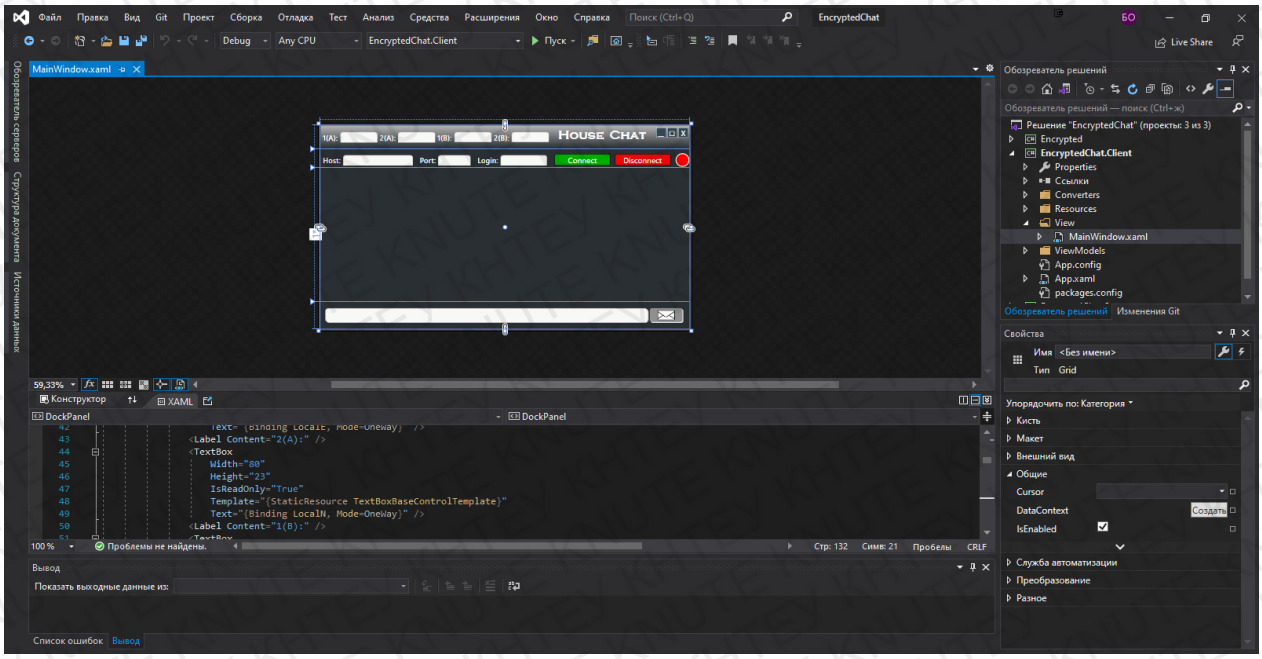

Рис.2.3 Створення дизайну програми

Висновок до розділу 2

Основне завдання даного проекту – це забезпечити миттєвий та безпечний зв'язок двох і декількох людей одночасно, шляхом передачі невеликої кількості інформації безпосередньо один одному. З цією метою створено безліч різних видів чатів, але більшості основний акцент зроблений на гарно оформлений інтерфейс, з яскраво вираженим текстом. В цьому випадку чат перетворюється у велику і складну програму яка вимагає максимум можливостей від користувача та ПК.

Даний чат може застосовуватися як для домашнього використання, так і для використання на малих підприємствах.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ПРОЕКТУ «HOUSE CHAT»

3.1 Загальні відомості про програму

Мережевий чат повинен включати в себе не тільки інформацію передану між користувачами, але і мати доступне управління. Програма повинна автоматизувати і скоротити витрату часу під час роботи.

Програма повинна виконуватися при наступних системних вимогах:

1. Процесор Intel Celeron J1800 (2.41 ГГц);
2. Оперативна пам'ять не менше 2Гб;
3. Графічна пам'ять не менше 2 Гб;
4. Операційна система Windows 7 і вище;
5. Вільне місце на жорсткому диску не менше 10 Мб;
6. Локальне з'єднання зі швидкістю 12 кбіт / сек і більше.

3.2 Характеристика програми

Програма має такі характеристики:

- Зрозумілий для першого використання інтерфейс.
- Миттєвий обмін конфіденційними повідомленнями.
- Найменше використання системних ресурсів.

3.3 Встановлення програми

Встановлювати програму не потрібно, так як для запуску необхідні самі виконувані файли та файли *.dll (рис.3.1);

• назва файлу для запуску сервера «Chat.Server.exe»;

• назва файлу для запуску клієнта «Chat.Client.exe»;

• для працездатності обох файлів необхідна наявність наступних файлів поруч:

1. Encrypted.dll;
2. Prism.dll;
3. Prism.xml;
4. System.Collections.Immutable.dll;
5. System.ComponentModel.Annotations.dll;
6. System.ValueTuple.dll;
7. System.ValueTuple

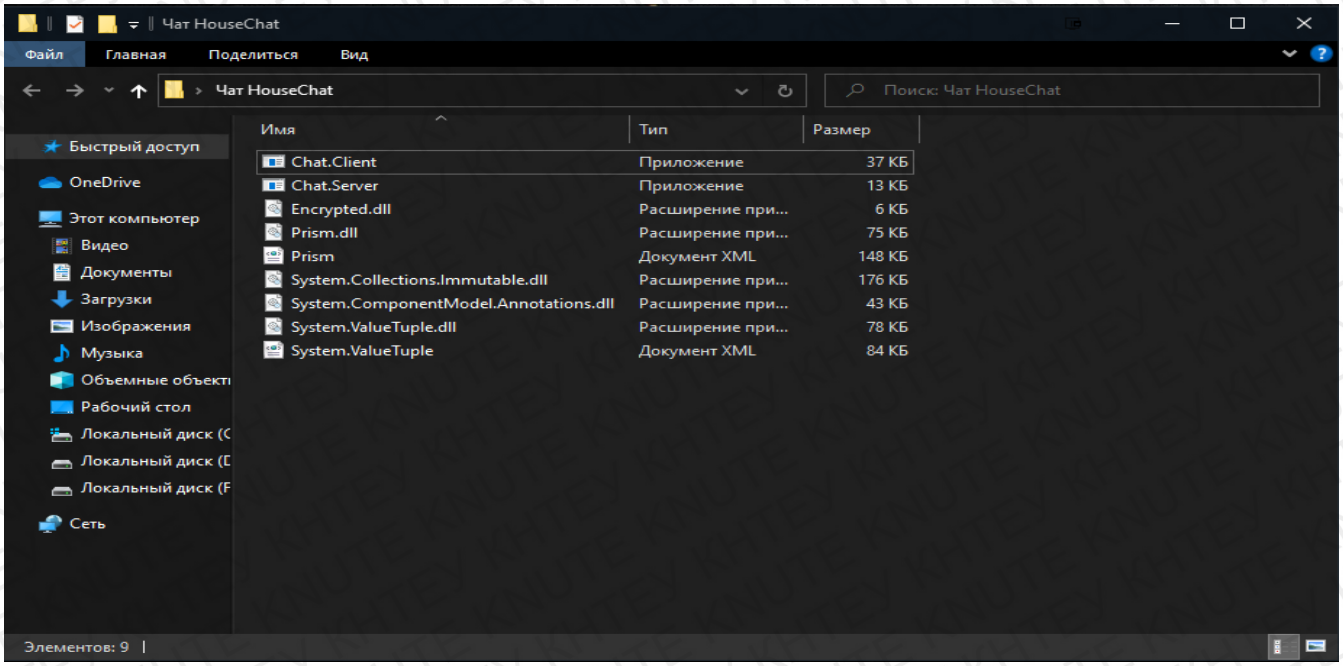

Рис.3.1 Файли роботи програми

3.4 Запуск програми

Для запуску сервера необхідно запустити файл «Chat.Server.exe» (рис.3.2);

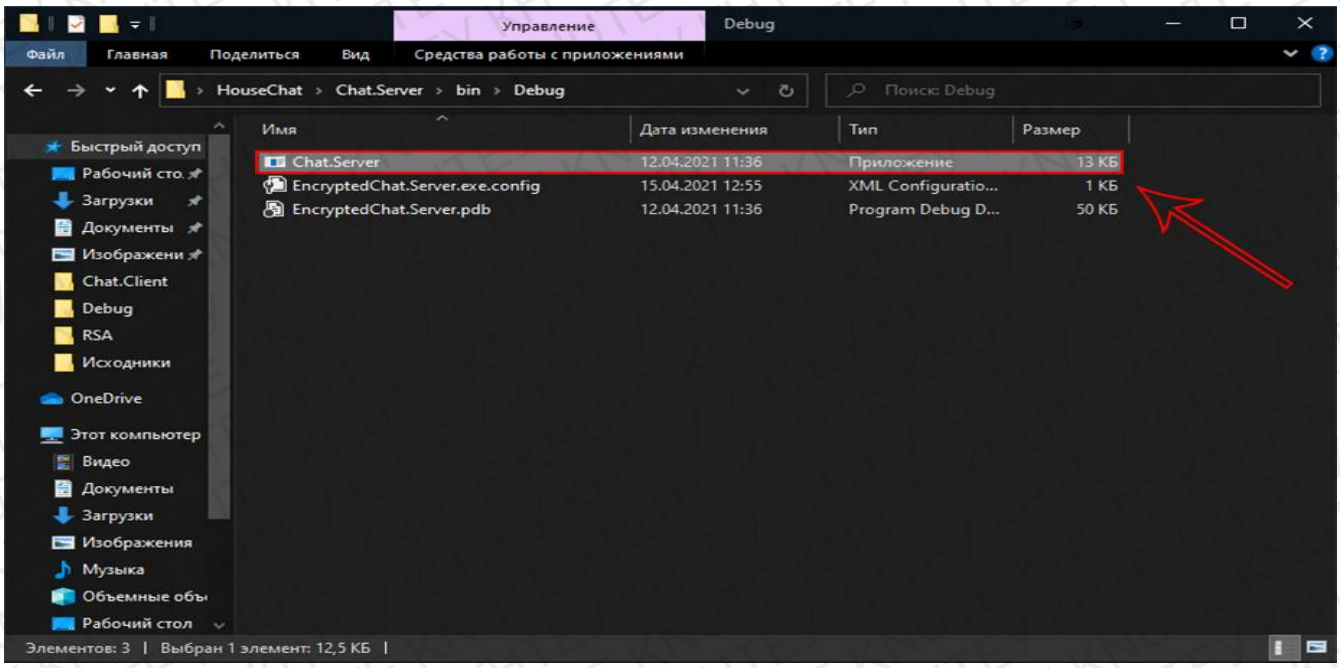

Рис.3.2 Файл «Chat.Server.exe»

Після чого з’явиться чорне вікно (консоль) з повідомленням, що сервер успішно запущений (рис.3.3).

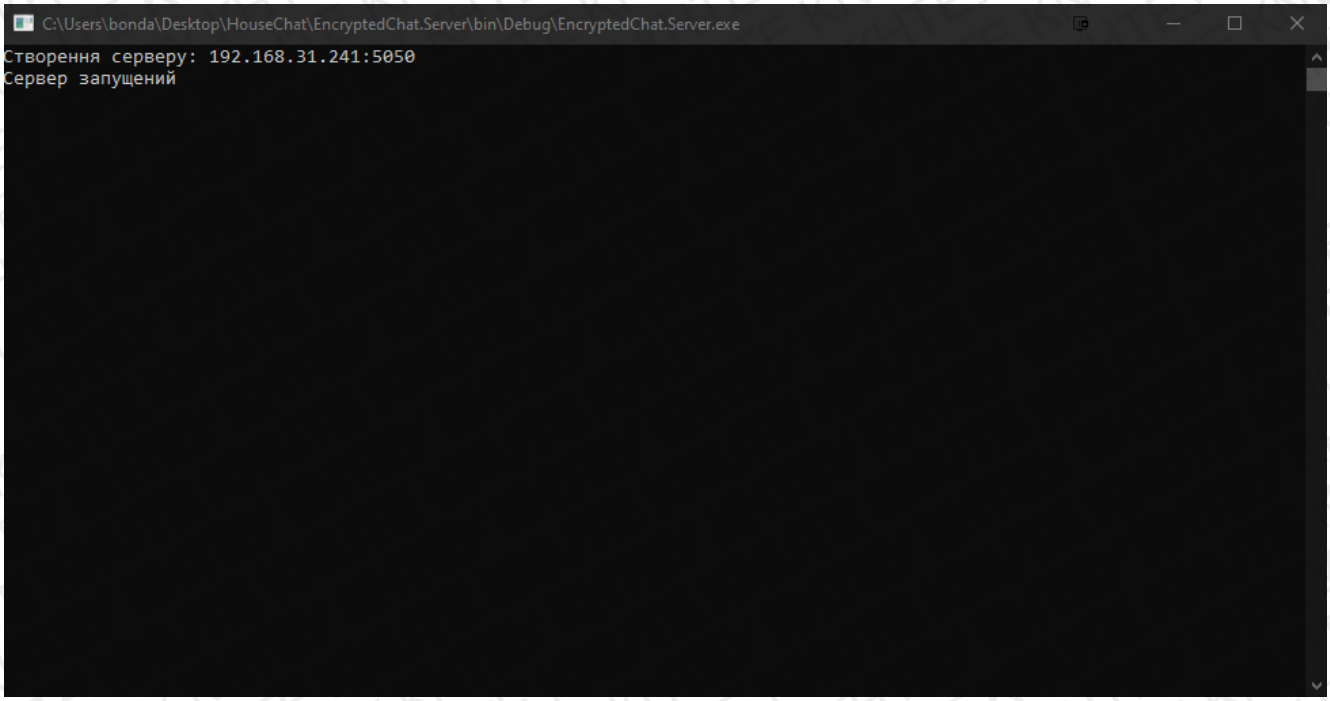

Рис.3.3 Успішно запущений сервер

Для запуску клієнта необхідно запустити файл «Chat.Client.exe» (рис.3.4);

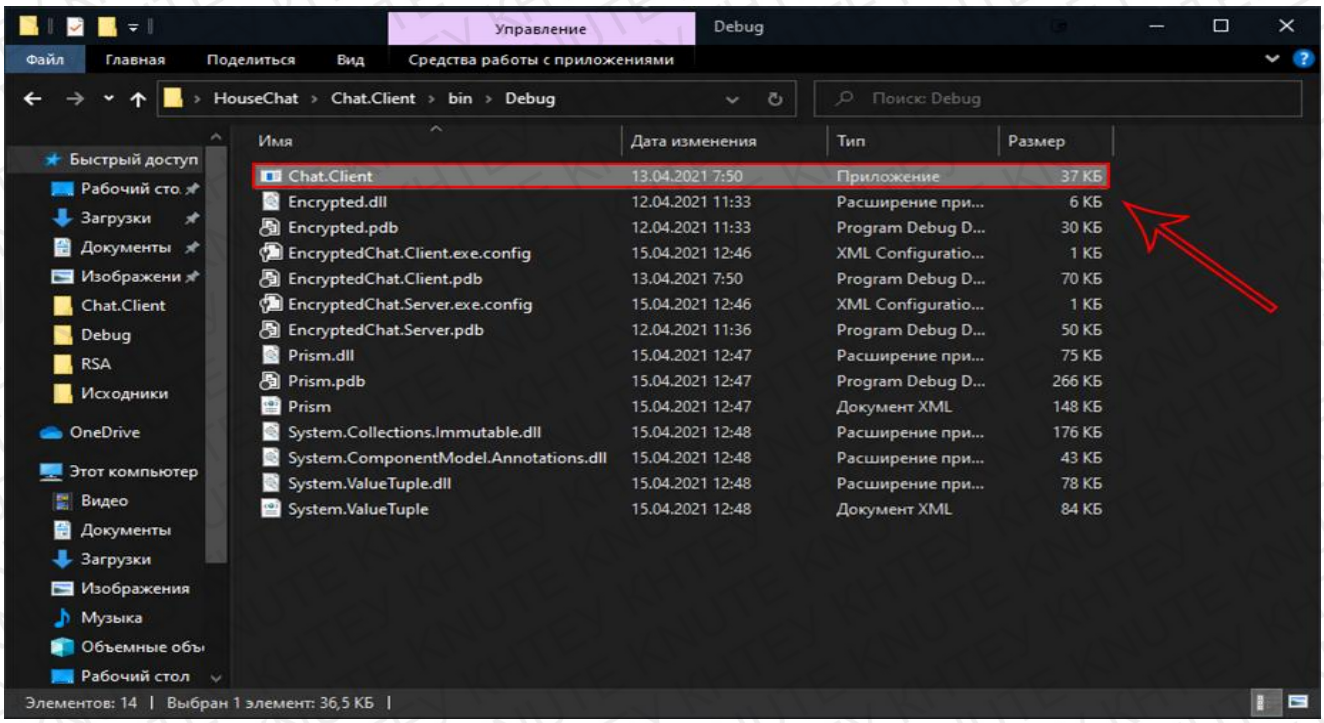

Рис.3.4 Файл «Chat.Client.exe»

Запустивши цей файл відкриється інтерфейс програми користувача (рис.3.5).

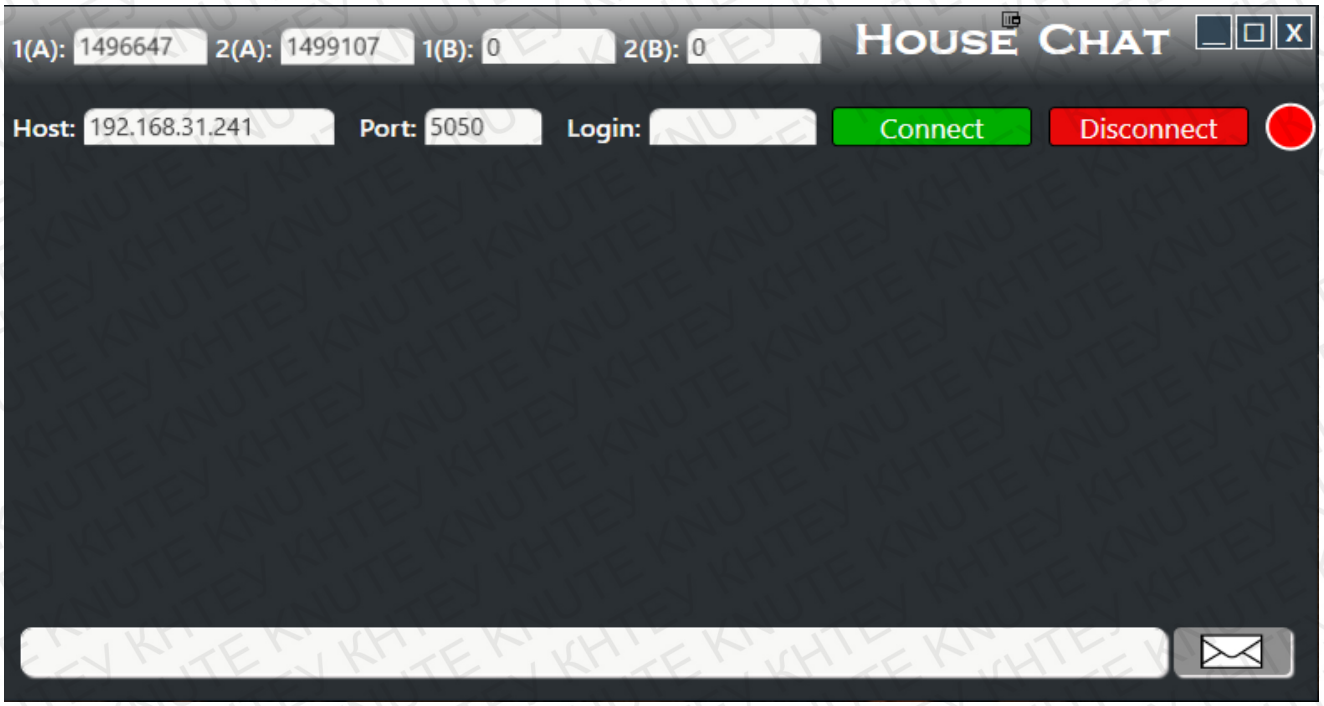

Рис.3.5 Інтерфейс програми користувача.

3.5 Клієнт «Chat.Client.exe»

Для клієнта в якості вихідних даних виступає відкритий ключ другого клієнта (1(A), 2(B)), унікальне ім'я клієнта в межах однієї сесії, IP адреси, на якій запущений сервер та порт.

3.6 Процес обміну ключовою інформацією

- ключова інформація генеруються під час запуску програми одноразово і втрачає актуальність після закриття клієнта;
- для забезпечення процесу обміну повідомленнями необхідно обмінятися відкритою парою ключів {1(A), 2(B)} вручну (процес не автоматизований) (рис.3.6).

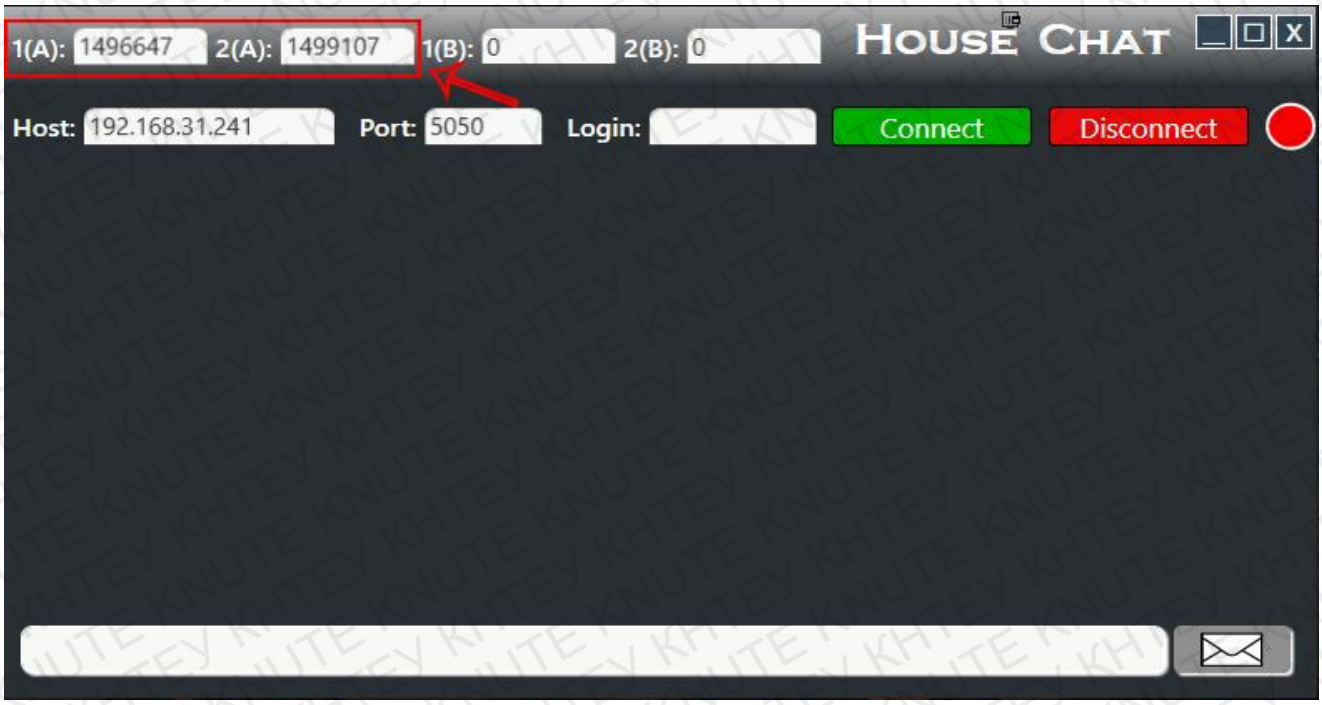

Рис.3.6 Ключі для обміну з іншим користувачем

3.7. Обмеження на використання програми

- якщо спробувати підключитися до сесії, в якій вже є користувач з таким іменем, то з'єднання не відбудеться;
- якщо клієнти обміняються некоректними ключами, то з'єднання розірветься;

– змінити ім’я користувача під яким ви підключились до чату можна лише після відключення від сесії.

3.8 Опис головного інтерфейсу програми та робота з нею

– Поля 1(A) та 2(A) які містять цифри. Ці цифри необхідно передати іншому користувачу для безпечного з’єднання.
– Поля 1(B) та 2(B) які мають значення 0. В ці поля необхідно ввести цифри, які повідомить вам інший користувач.
– В поле Host потрібно вказати IP адресу.
– В поле Port необхідно вказати порт.
– Поле Login відповідає за те, під яким іменем користувач приєднається до чату.
– Кнопка Connect відповідає за підключення до серверу.
– Кнопка Disconnect відповідає за відключення від серверу.
– Червоне коло (тригер) сповіщаєкористувача, що підключення до чату не відбулось. Якщо коло стає зеленим – підключення відбулось успішно.
– Пусте сіре місце, тут відображається переписка користувачів.
– Біле поле, використовується для написання повідомлення
– Кнопка з конвертом. За допомогою цієї кнопки користувач відправляє повідомлення також відправка можлива за допомогою клавіші Enter.

3.9 Результат роботи програми

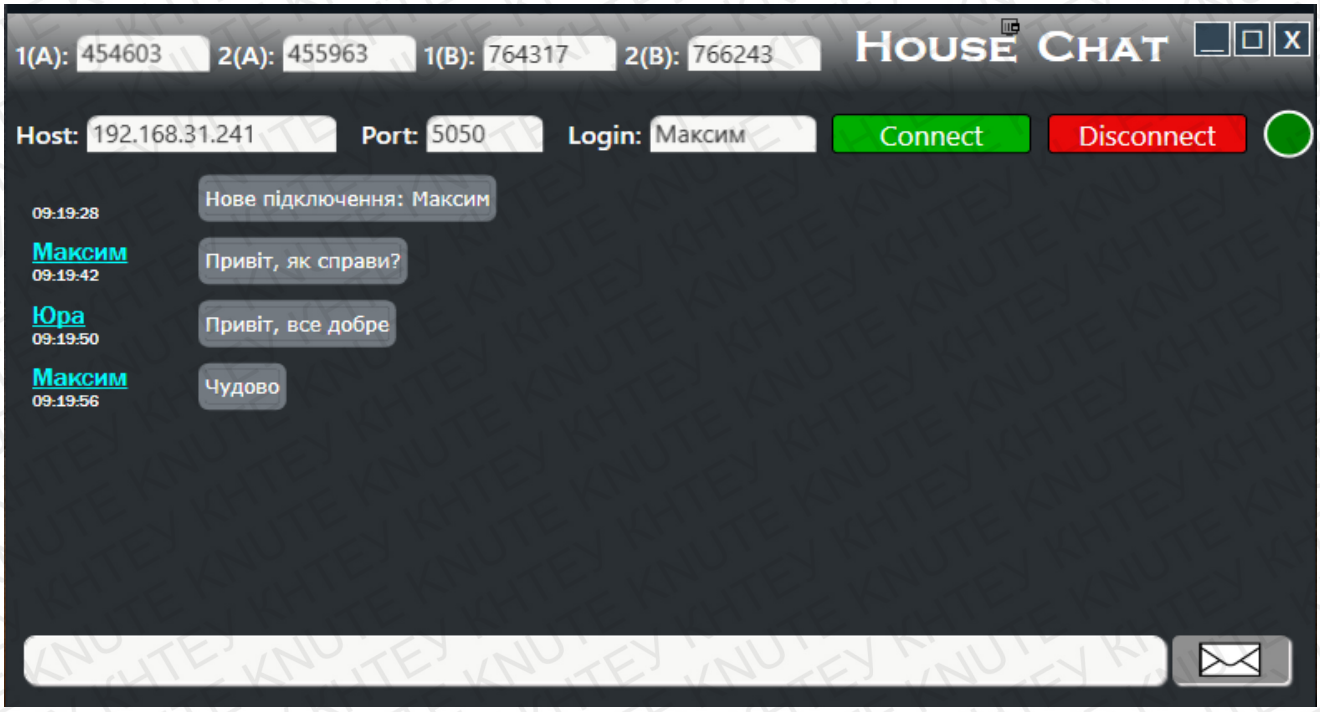

Рис.3.7 Перший користувач

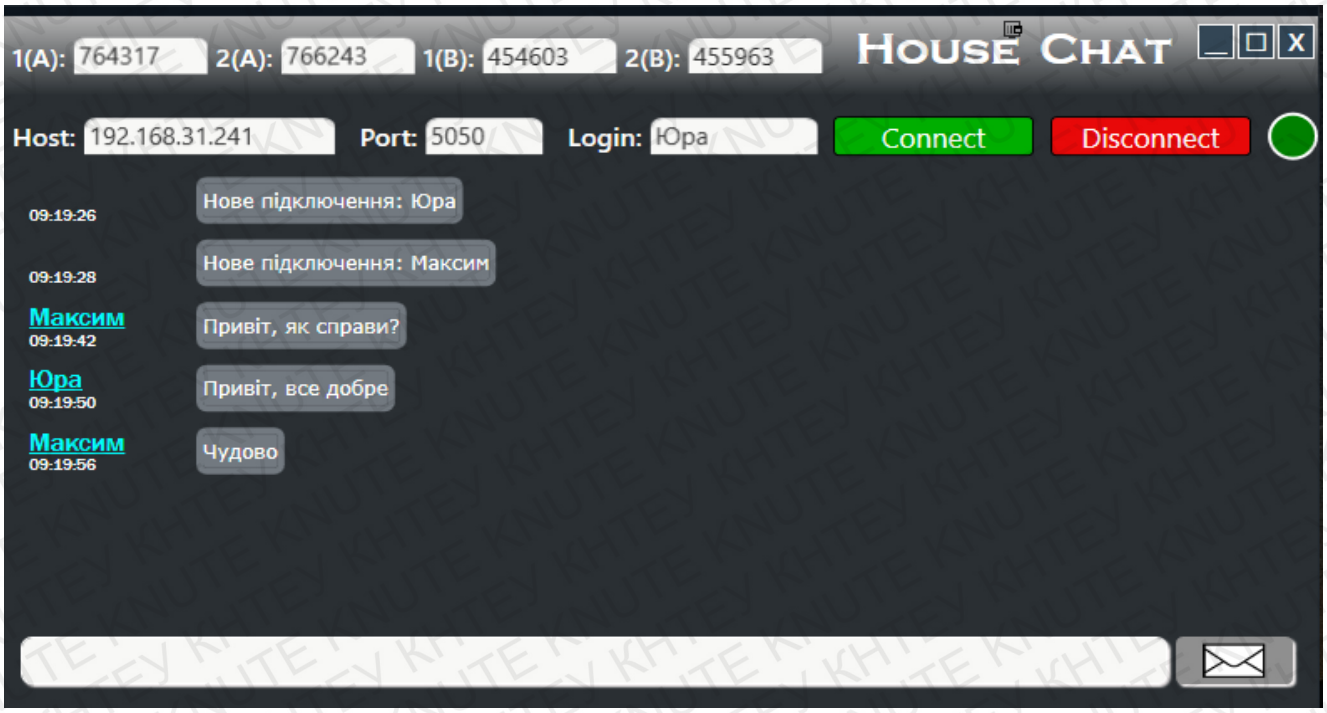

Рис.3.8 Другий користувач

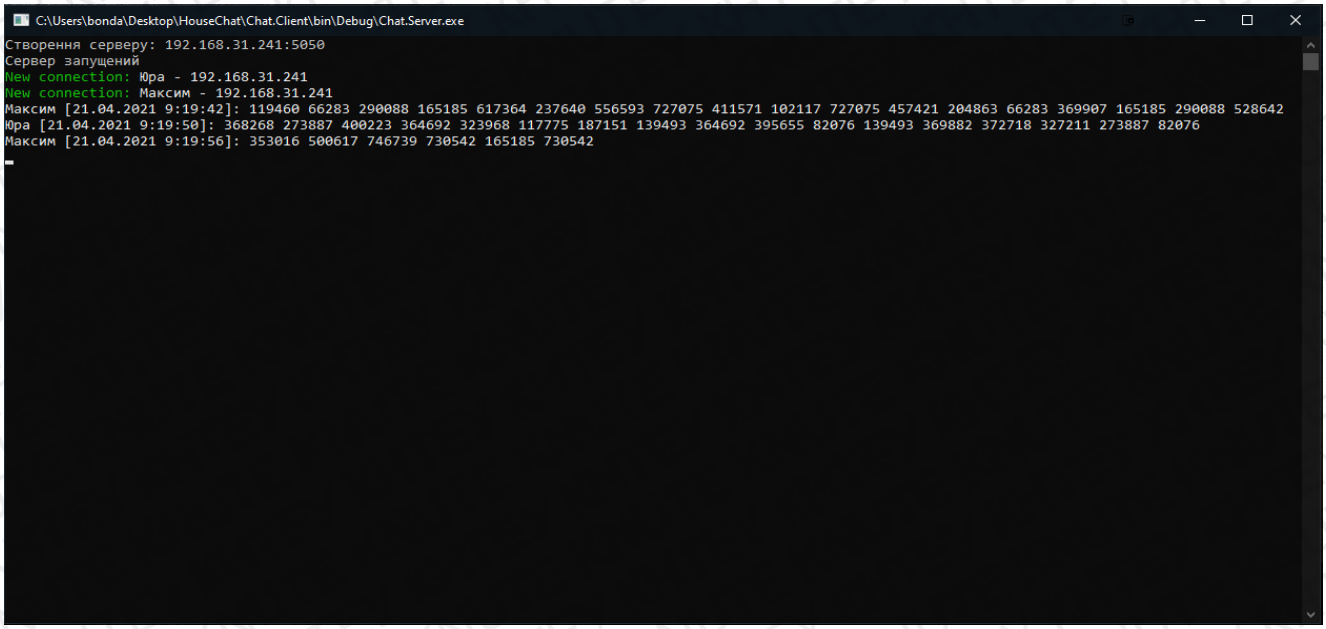

Рис.3.9 Лог серверу

3.10 Код програми «House Chat»

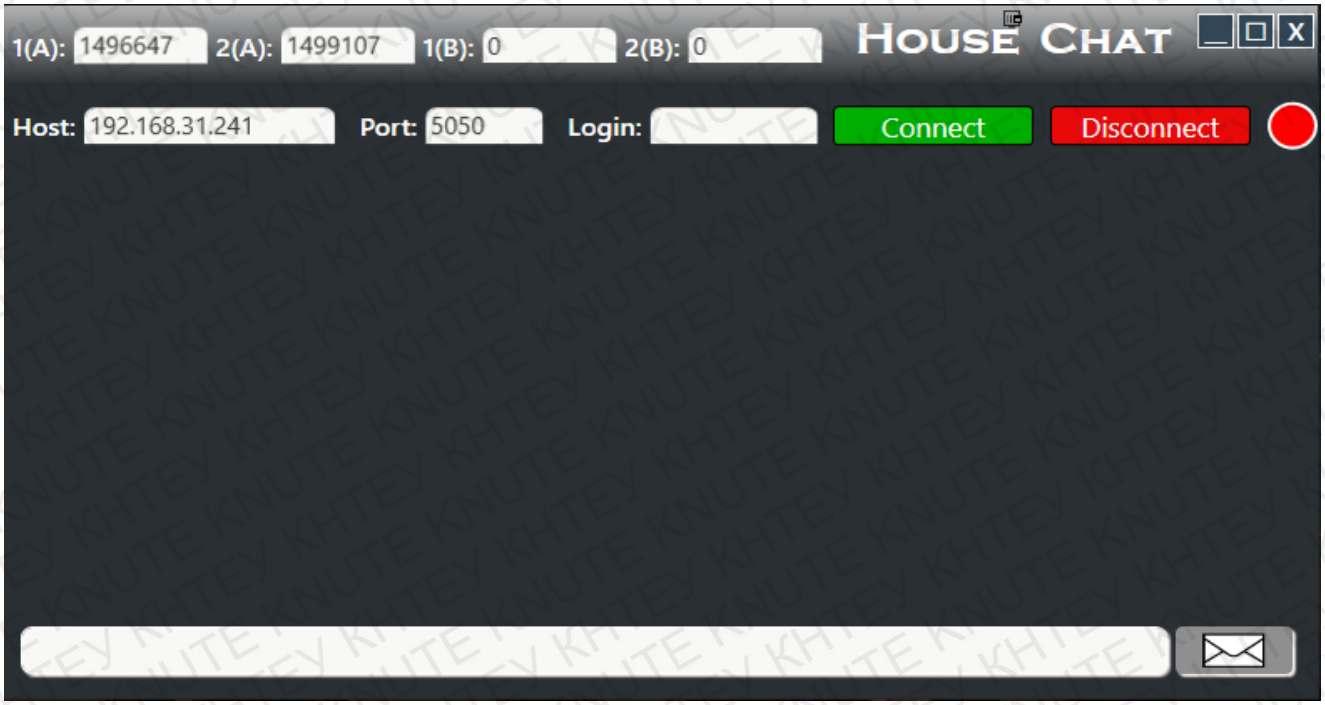

Рис.3.10 Програма «House Chat».

```
using Encrypted;
using EncryptedChat.Server.ClientModel;
using Prism.Commands;
using System;
```

```
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Net;
using System.Net.Sockets;
using System.Runtime.Serialization.Formatters.Binary;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Input;
using System.Windows.Threading;

namespace Chat.Client.ViewModels
{
public class MainViewModel : Design
//Створення нового ключа шифрування
private RSA _rsa = new RSA();
//Поля для введення ключів іншого користувача
public int Local1 => (int)_rsa.e;
public int Local2 => (int)_rsa.n;
//Перевірка введених ключів
public int Remote1 { get => remote1; set { remote1 = value; OnPropertyChanged(); } }
public int Remote2 { get => remote2; set { remote2 = value; OnPropertyChanged(); } }
//Поле для IP
public string Host { get; set; }
//Поле для порту
public int Port { get; set; }
//Поле для логіну (ім’я користувача)
public string Login { get; set; } = "";
//Поле для тексту
public string Text
{
get => _text; set
```

```
{
_text = value;
OnPropertyChanged();
}
//Кнопка відключення
private void Disconnect()
{
_tcpClient?.Close();
_canConnect = true;
CanDisconnect = false;
OnPropertyChanged();
}
//Кнопка підключення
private void Connect()
{
_tcpClient?.Close();
_tcpClient = new TcpClient();
//Перевірка на підключення до серверу
try
{
_tcpClient.Connect(Host, Port);
}
catch (Exception e)
{
MessageBox.Show(e.Message, "Помилка підключення до сервера",
MessageBoxButton.OK, MessageBoxImage.Information;
return;
}
//Перевірка на підключення до серверу іншого користувача
stream = _tcpClient.GetStream();
```

```
//Повідомлення про підключення іншого користувача
Messages.Add(new MessageItem() { Content = $"Нове підключення:
{conClient.Login}", SendTime = DateTime.Now };
};
//Повідомлення про відключення від чату
{
Messages.Add(new MessageItem() { Content = "З'єднання розірване...", SendTime =
DateTime.Now
}
//Відправка повідомлення
private void SendMessage()
{
try
{
//Шифрування цього повідомлення
var data = _rsa.Encrypt(Text, RemoteE, RemoteN);
//Виведення на екран повідомлення
Dispatcher.CurrentDispatcher.Invoke(() =>
{
Messages.Add(new MessageItem() { Login = _client.Login, Content = Text, SendTime
= DateTime.Now };
}

//Створення вікна програми, розмір та колір
<Window
x:Class="EncryptedChat.Client.View.MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
Title="MainWindow"
Width="800"
```

```
Height="450"
MinWidth="800"
MinHeight="200"
Background="#FF2A2F33"
WindowStyle="None">
//Поле 1(A)
</DockPanel.Background>
<StackPanel Orientation="Horizontal">
<Label Content="1(A):" />
<TextBox
Width="80"
Height="23"
IsReadOnly="True"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding LocalE, Mode=OneWay}" />
//Поле 2(A)
<Label Content="2(A):" />
<TextBox
Width="80"
Height="23"
IsReadOnly="True"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding LocalN, Mode=OneWay}" />
//Поле 1(B)
<Label Content="1(B):" />
<TextBox
Width="80"
Height="22"
IsReadOnly="{Binding CanDisconnect}"
Template="{StaticResource TextBoxBaseControlTemplate}"
```

```
Text="{Binding RemoteE, UpdateSourceTrigger=PropertyChanged}" />
//Поле 2(B)
<Label Content="2(B):" />
<TextBox
Width="80"
Height="22"
IsReadOnly="{Binding CanDisconnect}"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding RemoteN, UpdateSourceTrigger=PropertyChanged}" />
//Створення House
<Label Content="House" Width="110" Height="40" Margin="15,0,0,0"
VerticalAlignment="Top" FontSize="28" FontFamily="Copperplate Gothic Bold" />
//Створення Chat
<Label Content="Chat" Width="90" Height="40" VerticalAlignment="Top"
Margin="0,0,5,0" FontSize="28" Cursor="None" FontFamily="Copperplate Gothic
Bold" FontWeight="Bold" />
</StackPanel>
//Створення елементів управлінням вікном (згорнути, відкрити на весь екран,
закрити)
<WrapPanel
HorizontalAlignment="Right"
VerticalAlignment="Top"
Orientation="Horizontal">
<Button Click="MinimizeWindow" Content="___" />
<Button Click="MaximizeWindow" Content="☐" />
<Button Click="CloseWindow" Content="X" />
</WrapPanel>
<DockPanel Grid.Row="1" Margin="0,10,0,0">
<StackPanel Grid.Row="0" Orientation="Horizontal">
//Поле Host
```

```
<Label Content="Host:" />
<TextBox
Height="23"
MinWidth="150"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding Host}" />
</StackPanel>
<StackPanel Margin="10,0,0,0" Orientation="Horizontal">
//Поле Port
<Label Content="Port:" />
<TextBox
Width="70"
Height="23"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding Port}" />
</StackPanel>
<StackPanel Margin="10,0,0,0" Orientation="Horizontal">
//Поле Login
<Label Content="Login:" />
<TextBox
Width="100"
Height="23"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding Login, UpdateSourceTrigger=PropertyChanged}" />
</StackPanel>
<StackPanel
HorizontalAlignment="Right"
VerticalAlignment="Center"
Orientation="Horizontal">
//Створення кнопки Connect
```

```
<Button
Command="{Binding ConnectCommand}"
Content="Connect"
IsEnabled="{Binding Connected}"
Style="{StaticResource CommandButton}" Background="#FF00AE00" />
//Створення кнопки Disconnect
<Button
Command="{Binding DisconnectCommand}"
Content="Disconnect"
IsEnabled="{Binding CanDisconnect}"
Style="{StaticResource CommandButton}" Background="#FFE80909" />
//Створення тригеру який змінює колір при підключенні або відключення від
серверу
<Ellipse Width="30" Height="30">
<Ellipse.Style>
<Style TargetType="{x:Type Ellipse}">
<Setter Property="Fill" Value="Red" />
<Setter Property="Stroke" Value="White" />
<Setter Property="StrokeThickness" Value="2" />
<Style.Triggers>
<DataTrigger Binding="{Binding CanDisconnect}" Value="True">
<Setter Property="Fill" Value="Green" />
</DataTrigger>
</Style.Triggers>
</Style>
</Ellipse.Style>
</Ellipse>
</StackPanel>
<Grid Grid.Row="3" Margin="10,10,10,10">
<Grid.ColumnDefinitions>
```

```
<ColumnDefinition />
<ColumnDefinition Width="auto" />
</Grid.ColumnDefinitions>
//Створення поля для набору тексту
<TextBox
VerticalContentAlignment="Center"
Template="{StaticResource TextBoxBaseControlTemplate}"
Text="{Binding Text, UpdateSourceTrigger=PropertyChanged}" />
//Створення кнопки яка відправляє текст
<Button
Grid.Column="1" IsDefault="True"
Command="{Binding SendMessageCommand}"
IsEnabled="{Binding Text, Converter={StaticResource EmptyTextConverter}}"
Style="{StaticResource SendButton}" Background="#FF8F8F8F"
Click="Button_Click_1" />
</Grid>
```

Висновок до розділу 3

«House Chat» забезпечує безпечний та миттєвий зв'язок двох людей одночасно. Програма працює без встановлення на комп’ютер, що є великою перевагою для людей які не впевнено користуються ПК, програмний продукт не вимагає ніяких інструктажів з використання і зрозумілий на інтуїтивному рівні.

ВИСНОВКИ

Результатом даної випускної кваліфікаційної роботи було створення клієнт-серверного-мережевого чату.

В якості базової мови для проектування було обрано мову програмування C#, що дало змогу розробити повноцінний програмний продукт.

На ринку програмного забезпечення існує велика кількість подібних програмних продуктів, які так само допомагають вирішувати порушені мною проблеми. Однак подібні програми часто вимагають від користувача певних знань ПК, і споживають велику кількість ресурсів комп'ютера, тому власне і було вирішено написати дану програму.

Даний програмний продукт дає можливість, маючи мінімальні знання для роботи на комп'ютері вільно користуватися програмою.

Даний програмний продукт може бути використаний як в соціальній та і промисловій сфері.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Пройдаков Е. М. Англо-український тлумачний словник з обчислювальної техніки, інтернету і програмування./ Пройдаков Е. М. Теплицький Л. А. – К. : Видавничий дім «СофтПрес», 2006. – 824 с.
2. Касперски К. «Секреты поваров компьютерной кухни или ПК: решение проблем»,/ Касперски К. –Издано: 2003, BHV – 560 с.
3. Шпак З. Я. Програмування мовою C# /. Шпак З. Я. – Львів:Оріяна-Нова, 2006. – 432 с.
4. Дейтел Х. М. Как программировать на C# / Дейтел Х. М., Дейтел П.Дж.: 4-е издание. Пер. с англ. – М.:ООО «Бином – Пресс», 2009. – 912 с.
5. Шилдт Г. Полный справочник по C / Шилдт Г., 4-е издание.: Пер. с англ. – М.: издательский дом «Вильямс», 2010. – 704 с.
6. С.С. Байдачний .Net Framework 2.0. Секреты создания Windows приложений / С.С. Байдачний – М.СОЛОН-Пресс, 2006 – 520с.

Електронні ресурси

1. ***https://uk.wikipedia.org/wiki/%D0%A7%D0%B0%D1%82 – Чат Вікіпедія***
2. ***https://uk.wikipedia.org/wiki/TCP/IP – TCP/IP Вікіпедія***
3. ***https://uk.wikipedia.org/wiki/RSA – RSA Вікіпедія***
4. ***https://uk.wikipedia.org/wiki/C_Sharp – C Sharp Вікіпедія***
5. ***https://uk.wikipedia.org/wiki/XAML – XAML Вікіпедія***

					КНТЕУ 121 06-02.БР		
					Розробка клієнт-серверного мережевого чату	Стадія	Аркуш	Аркушів
						СД	41	41
Зм	Аркуш	№ докум.	Підпис	Дата				
Зав. каф.		Криворучко О.В.			Список використаних джерел	Факультет інформаційних технологій, 4 курс, 6 група		
Керівник		Харченко О.А.						
Гарант		Цензура М.О.						
Розроб.		Бондар В.О.						