

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ МОБІЛЬНОГО
ЗАСТОСУНКУ ДЛЯ ЧАТУВАННЯ З АНОНІМНИМИ
КОРИСТУВАЧАМИ СПІЛЬНОТ
SOFTWARE IMPLEMENTATION OF A MOBILE
APPLICATION FOR CHATTING WITH ANONYMOUS
GROUP MEMBERS

Виконав: студент(ка) 4 курсу, групи КНТ-3117
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

Старікова А.М.

(прізвище та ініціали)

Керівник Гофман Є.О.

(прізвище та ініціали)

Рецензент Скрупський С.Ю.

(прізвище та ініціали)

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

Інститут, факультет ІРЕ, ФКНТ
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.
С.О. Субботін
“ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Старіковій Альоні Михайлівні

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація мобільного застосунку для чатування з анонімними користувачами спільнот. Software Implementation of a Mobile Application for Chatting with Anonymous Group Members

керівник проєкту (роботи) Гофман Євген Олександрович, к.т.н.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) травень 2021 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. Розробка архітектури програми. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Гофман Є.О., доцент		
Нормоконтролер	Липовець М.В., асистент		

7. Дата видачі завдання “1” лютого 2021 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 та дослідження предметної області	2-5 тижні	Розділи 1, 2
3	Розробка архітектури програми	6 тиждень	Розділ 2
4	Розробка програми	7-9 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	10 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	11 тиждень	Додатки
7	Захист роботи	12 тиждень	

Студент(ка)

(підпис)

Старікова А.М.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Гофман Є.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 95 с., 52 рис., 3 табл., 4 дод., 13 джерел.

МОБІЛЬНИЙ ЗАСТОСУНОК, АНОНІМНІСТЬ, ЧАТУВАННЯ, ВІРТУАЛЬНЕ СПІЛКУВАННЯ, РОЗРОБКА, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ.

Об'єкт дослідження – процес анонімного чатування.

Предмет дослідження – методи організації анонімного чатування.

Метою роботи є розроблення мобільного застосунку, що реалізує можливість користувачів спілкуватися анонімно.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування Java, середовище розробки Android Studio, система керування базою даних MySQL.

Результати. Розроблено програмне забезпечення у вигляді мобільного застосунку для анонімного чатування.

Наукова новизна роботи полягає в розробці мобільного застосунку для організації анонімного чатування користувачів серед спільнот.

Висновки. Розроблено мобільний застосунок для організації анонімного чатування.

Галузь використання – користувачі мобільних пристроїв під управлінням операційної системи Android.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 95 pages, 52 figures, 3 tables, 4 appendixes, 13 sources.

MOBILE APPLICATION, ANONYMITY, CHAT, VIRTUAL COMMUNICATION, DEVELOPMENT, SOFTWARE.

The object of research is the process of anonymous guarding.

The subject of the research is methods of organizing anonymous guarding.

The aim of this work is to develop a mobile application that allows users to communicate anonymously.

Materials, methods, and technical tools: Object-Oriented Programming, Java programming language, Android Studio development environment, MySQL database management system.

Results. Software has been developed in the form of a mobile application for anonymous chat.

The scientific novelty of the work is the development of a mobile application for organizing anonymous chat of users among communities.

Conclusions. A mobile application for organizing anonymous surveillance has been developed.

Field of usage: users of mobile devices running the Android operating system.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ	9
1 Аналіз предметної області.....	10
1.1 Проблемна область віртуальне спілкування. Переваги та недоліки віртуального спілкування.....	10
1.1.1 Чатування як форма віртуального спілкування.....	12
1.2 Огляд існуючих рішень	13
1.2.1 Мобільний застосунок «Omegle»	14
1.2.2 Мобільний застосунок «Yik Yak»	15
1.2.3 Мобільний застосунок «Whisper»	15
1.2.4 Порівняння існуючих аналогів	16
1.3 Постановка завдання.....	18
1.3.1 Функціональність системи.....	18
1.3.2 Вимоги до інтерфейсу	19
1.3.3 Вимоги до даних	19
1.3.4 Не функціональні вимоги до продукту.....	20
1.3 Висновки за розділом 1	20
2 Розробка архітектури програми.....	22
2.1 Вибір середовища проєктування	22
2.1.1 Вибір мови програмування	22
2.1.2 Вибір середовища розробки.....	24
2.1.3 Порівняння середовищ розробки	29
2.2 Загальна структура додатку	30
2.3 Висновки за розділом 2	32
3 Основні рішення щодо реалізації компонентів системи.....	33
3.1 Проєктування моделей системи	33
3.2 Фізичне проєктування.....	40
3.3 Проєктування інтерфейсу.....	43

3.4 Висновки за розділом 3	48
4 Експлуатація, тестування та експериментальне дослідження програми	49
4.1 Опис застосування	49
4.2 Умови виконання програми	50
4.3 Інструкція по експлуатації програми	50
4.3.1 Характеристики програми	50
4.3.2 Перелік файлів, у яких знаходиться програма	52
4.3.3 Звернення до програми	52
4.3.4 Вхідні й вихідні дані	53
4.4 Методика та результати тестування	53
4.4.1 Тестування функціоналу	53
4.4.2 Тестування за допомогою N-Stalker	56
Висновки	58
Перелік джерел посилання	60
Додаток А Технічне завдання	62
Додаток Б Опис програми	68
Додаток В Текст програми	74
Додаток Д Слайди презентації	88

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

A – кроки альтернативного сценарію виконання;

B – кроки базового сценарію виконання;

GUI – Graphical user interface;

MVC – Model-view-controller;

UML – Unified Modeling Language;

БД – база даних;

ОС – операційна система

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер;

СКБД – система керування базами даних;

ТЗ – технічне завдання.

ВСТУП

Кількість мобільних пристроїв постійно зростає і також зростає кількість мобільних додатків. Android – найпоширеніша мобільна операційна система (ОС). Також більшість мобільних пристроїв мають відносно постійне інтернет-з'єднання. Тому при реалізації невеликого за функціоналом, але частого у використанні додатку доречно вибирати саме мобільні платформи.

Веб-застосунок – клієнт-серверний додаток, в якому клієнтом виступає браузер або інша клієнтська програма, а сервером – веб-сервер. Логіка веб-застосунків розподілена між сервером і клієнтом, зберігання даних здійснюється, переважно, на сервері, обмін інформацією відбувається по мережі.

Android-застосунок – клієнтський додаток, який працює на пристроях з ОС Android.

Зв'язка клієнтського додатку та веб-додатку – поширена та зручна практика не тільки для мобільних, а й для інших видів додатків.

На сьогоднішній день месенджери – за стосунки для спілкування у форматі миттєвих повідомлень – з кожним днем стають все більш поширеними, проте вони вимагають від користувача попередньої реєстрації: за номером телефону, електронною поштою, тощо. Більш того при спілкуванні в більшості випадків користувачі мають доступ до таких даних один одного. А маючи просто, наприклад, номер телефону можна знайти профіль людини в більшості месенджерів.

Проте залишаються актуальним розробка таких за стосунків, які б дозволили б обмінюватися повідомленнями анонімно за попередньою домовленістю.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Проблема область віртуальне спілкування. Переваги та недоліки віртуального спілкування

Віртуальне спілкування – невід’ємна складова сучасних соціальних процесів людства та суспільства. Для багатьох комп’ютер та смартфон – не просто різновид техніки, а скоріше співрозмовник. Саме по собі віртуальне спілкування має ряд позитивних аспектів: можливість обмінюватися інформацією з людьми, які знаходяться за тисячі кілометрів від нас, пошук однодумців, спосіб знайти віддалену роботу, стати фінансово незалежним. Однак, віртуальне спілкування також може стати причиною появи деяких проблем [1].

Проблема ворожого ставлення до «чужинців», які не належать до якихось спільнот. Люди, які проводять багато часу на певних форумах, в чатах, поступово один з одним знайомляться, об’єднуються в спільноти, з’являються негласні правила. Коли непосвячений новачок потрапляє на такий форум і намагається почати спілкування, він може бути грубо висміяний, ображений постійними «мешканцями» форуму. Дрібні помилки, промахи непосвяченого роздуваються до вселенських масштабів. Такий новачок або постарається заслужити прихильність старожили і сам буде надалі грубо ставитися до нових користувачів, або назавжди покине цей форум [2].

Проблема заміщення. Взаємодія через Інтернет залишає тільки саме спілкування як діяльність, хоча в житті воно частіше є елементом іншої діяльності (праці, навчання, гри), а вчинки і дії цінуються більше, ніж слова.

Проблема випадкового повідомлення. Випадкова відправка листа не тому адресату, скріншоти, записи та інші «компромати» можуть дуже негативно відобразитися на самооцінці людини [1]-[2].

З іншого боку віртуальне спілкування має ряд переваг. Серед таких слід виділити не просто спілкування на відстані або можливість спілкування для людей із особливими потребами, а саме психологічні переваги.

Віртуальне спілкування, як спосіб самовираження. Безумовно, усе інтернет-поле для самовираження. Чим і користуються блогери і відеоблогери, редактори журналів, автори сторінок і груп в соціальних мережах. Все це-форми спілкування в мережі. Можливість самовираження сама по собі прекрасна. Питання в тому, як саме людина вирішить самовиражатися, що він хоче донести людям і що бере сам. Головне, щоб не виникло «самотності в мережі»: коли і глядачів багато, і шанувальники є, і друзі віртуальні, і коментарів вистачає, а щастя в житті немає, і за допомогою звернутися нема до кого [1]-[2].

Можливість «нової» самоідентифікації та ідентифікації. У реальному житті кожен з нас відіграє певну роль, яка накладає відбиток на зовнішній вигляд, на спілкування, поведінку. При віртуальному спілкуванні фактор зовнішнього вигляду не має значення, адже можна написати про себе будь-яку інформацію. В онлайн-світі вік, стать, сексуальну орієнтацію, соціальний статус вибираємо ми самі. Наприклад, 45-річний чоловік може написати про себе, що він: 15-річний підліток та зареєструватися на форумі для обговорення улюбленої книги або фільму, що йому в звичайному житті «не за віком». Такий підхід може дозволити компенсувати певну неуспішність в реальному світі або відсутність кола близьких людей за інтересами. Крім того, при віртуальному спілкуванні в більшості випадків не знаєш, з ким спілкуєшся, тим самим відпадає потреба формальних взаємовідносин, що можуть ускладнювати соціальний контакт. У реальному світі існує модель спілкування між підлеглим і начальником, учителем і учнем, батьками і дітьми. У віртуальному світі ця грань стирається [1]-[2].

З іншого боку остання теза може містити і певні ризики. Адже повне де-ідентифікування може спричинити відсутність відповідальності за протизаконні дії. Дана проблема може ставати наслідком попередньої проблеми. Коли віртуальний співрозмовник не бачить тебе, знаходиться від тебе за тисячі кілометрів, відповідальність перед ним зникає. Можна пообіцяти що завгодно, взяти на виконання будь-яку роботу і не виконати,

адже обіцянка давалася клавіатурі і монітора, а не живій людині, та й у багатьох випадках покарати за невиконання чого-небудь в Інтернеті не так просто. За допомогою слів людина передає близько 7% інформації, звуку та інтонації – 38%, за передачу решти 55% інформації відповідає міміка, поза, жести, тобто невербальні комунікативні засоби. При віртуальному спілкуванні, щоб заповнити відсутні 93% переданої інформації, стали використовуватися в текстових повідомленнях смайлики, розділові знаки, скорочення. Такі засоби все ж таки не здатні передати весь спектр емоцій.

Саме тому певна анонімність повинна регулюватися та підтримуватися тільки після попередньої домовленості [1]-[2].

1.1.1 Чатування як форма віртуального спілкування

Чат – програмне забезпечення (ПЗ), як правило у вигляді веб- або мобільного застосунку для обміну повідомленнями по комп'ютерній мережі в режимі реального часу. А також ПЗ, що дозволяє організовувати таке спілкування. Характерною особливістю є комунікація саме в реальному часі або близька до цього, що відрізняє чат від форумів та інших «повільних» засобів [1]-[2].

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 або навіть SMS.

Існує кілька різновидів програмної реалізації чатів:

– HTTP або веб-чати. Такий чат виглядає як звичайна веб-сторінка, де можна переглянути історію повідомлень, тобто певну множину останніх фраз, що написані учасниками чату (в деяких випадках, модераторами). Сторінка чату автоматично оновлюється із заданою часовою періодичністю;

– чати, що використовують технологію Adobe Flash. Замість періодичного перезавантаження сторінки між клієнтом і сервером

відкривається сокет, що дозволяє моментально відправляти або отримувати повідомлення, витрачаючи менше трафіку;

- IRC, спеціалізований протокол для чатів.

- ПЗ-чати для спілкування в локальних мережах, які дуже часто надають можливість передачі файлів. Такі корпоративні чати можуть використовуватися окремими відділами або цілими компаніями, але виключно для ділового спілкування;

- чати, що працюють за схемою клієнт-сервер, це дозволяє використовувати їх в мережах зі складною конфігурацією, а також управляти клієнтськими застосунками;

- чати, що працюють в однорангових мережах. У них немає потреби в окремому сервері, вони часто використовують можливості технологій DHT і TCP Relay;

- чати, що використовують технологію Push. Замість періодичної відправки запитів серверу про нові повідомлення, використовуються вхідні повідомлення від сервера, що дозволяє відправляти і отримувати повідомлення, витрачаючи менше трафіку;

- повністю анонімні чати. У них співрозмовник не передбачає з ким спілкується і при кожному новому з'єднанні спілкується з новою людиною.

1.2 Огляд існуючих рішень

Хоча програми для обміну повідомленнями, такі як Telegram або Facebook Messenger, мають багато функцій, люди все ще продовжують шукати інші доступні варіанти.

Насправді, анонімні месенджери є ідеальним вибором для тих, хто має намір переїхати в нові місця. Крім того, вони можуть бути ідеальними для тих, у кого немає друзів поблизу. Якщо людина соромиться спілкуватися з незнайомцями і заводити нових друзів, він може скористатися анонімним застосунком чату. Він може почати розмову в анонімному режимі і проявити

себе, коли відчує себе впевнено. На сьогоднішній день існують веб- та мобільні застосунки, що надають можливість анонімного чатування, проте вони мають певні обмеження та недоліки.

1.2.1 Мобільний застосунок «Omegle»

Omegle сьогодні вважається одним з найпопулярніших застосунків для анонімного чату на ринку [3]. Цей інструмент був вперше запущений більше десяти років тому (у 2008 році). Omegle володіє надшвидкою функціональністю і дуже зручним для користувача інтерфейсом. Користувач може використовувати його для спілкування практично з будь-яким незнайомцем, який живе де завгодно. Можна використовувати його для чату зі збереженням анонімного профілю у випадковому режимі. Також можна вказати свої інтереси. Щоб вибрати інтереси, досить струсити телефон. Потім інтелектуальна система, інтегрована в Omegle, знайде схожі інтереси і дозволить вам зв'язатися з однодумцями. І як доповнення до цього, користувачі також можуть використовувати цю програму для анонімного чату та відеочату [3].

Приклад вікон застосунку наведено на рисунку 1.1.

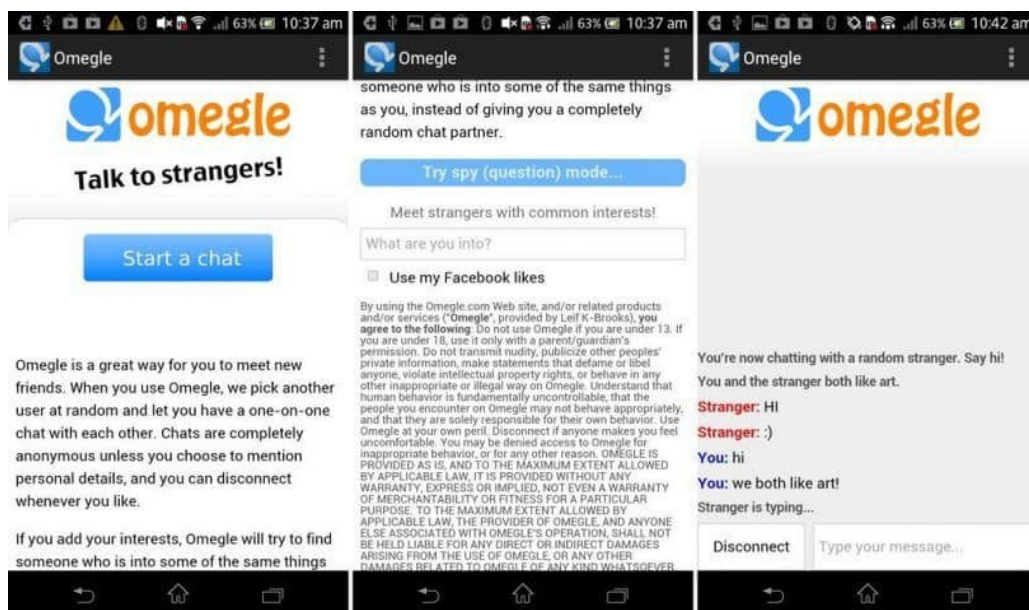


Рисунок 1.1 – Приклад вікон застосунку Omegle

1.2.2 Мобільний застосунок «Yik Yak»

Yik Yak – це ще одне популярне ПЗ, яке дозволяє спілкуватися з тими, хто має подібні інтереси [4]. Якщо ж спільних інтересів не має – користувач може просто знайти причину, щоб поговорити з ними. Якщо з обома сторонами все в порядку, користувач також може перейти на приватний канал. Ці приватні канали дозволяють наблизитися один до одного і продовжити бесіду дискретним способом. Це також дозволяє приєднатися до інших застосунків. Інтерфейс програми досить простий і зручний для користувачів. На платформі Yik Yak безліч користувачів, тому є більше шансів вибрати одного [4].

Приклад вікон застосунку наведено на рисунку 1.2.

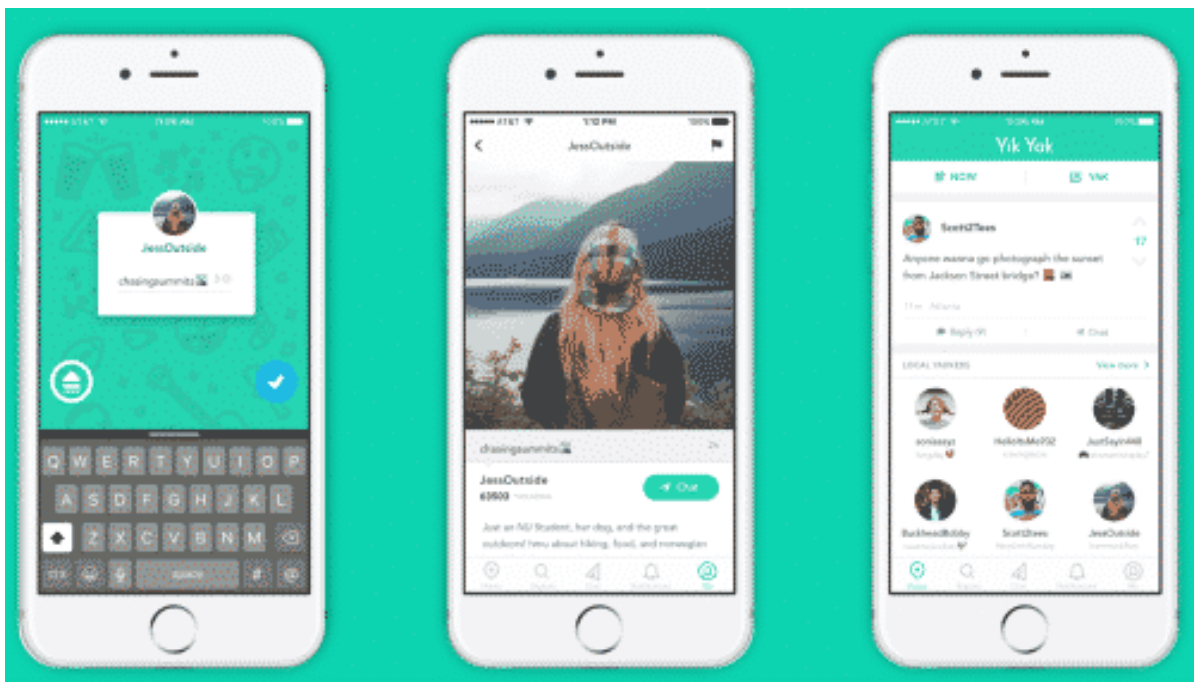


Рисунок 1.2 – Приклад вікон застосунку Yik Yak

1.2.3 Мобільний застосунок «Whisper»

Це ще один потужний варіант для тих, кому потрібно анонімно спілкуватися в чаті зі своїх смартфонів [5]. За останній час цей застосунок

набув значної популярності. Користувачка база цього застосунку зростає швидше. Це означає, що знайти друга через Whisper стає простіше. Однак у цього додатка є особливість в порівнянні з деякими чат-платформами. Це означає, що Whisper не сприймає контент відверто сексуального характеру. Замість цього він спонукає користувачів вести більш змістовні непристойні розмови. Використовуючи цю платформу, користувач може пояснити свої почуття та ідеї абсолютно незнайомим людям і розслабити свій розум. Принадність цього застосунку в тому, що він не змушує користувачів до того, щоб вони розкривали свою особистість. Крім того, це дозволяє користувачам знаходити нових друзів на основі їх інтересів, місця розташування тощо [5].

Приклад вікон застосунку наведено на рисунку 1.3.

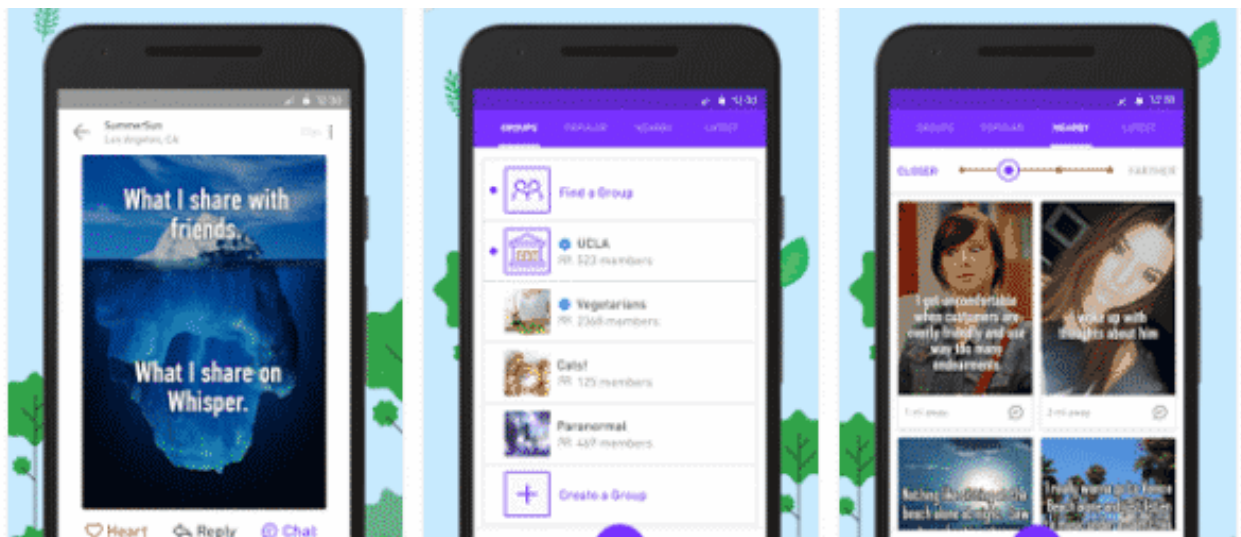


Рисунок 1.3 – Приклад вікон застосунку Whisper

1.2.4 Порівняння існуючих аналогів

Проведемо порівняння за певними критеріями, хід якого представимо у вигляді таблиці (табл. 1.1). Серед критеріїв виділимо наступні:

– анонімність на початку користування – тобто чи вимагає сервіс на початку якусь авторизацію за персональними даними (електронною поштою,

номером телефону, тощо) або ж просто доступна реєстрація за якимось внутрішнім ідентифікатором застосунку;

– вибір персональної інформації – чи потрібно на початку користування вказувати персональну інформацію (коло інтересів, цікаві теми, тощо) та чи можливо надалі контролювати розголошення цієї інформації;

– урегульованість тематик – чи відбувається контроль адміністрацією тематик чатування для забезпечення безпеки користувачів;

– адміністрування чатів – чи відбувається адміністрування та модерація чатів для забезпечення безпеки користувачів;

– платний функціонал – чи є певний функціонал, що може бути платним та вимагає додаткових коштів (наприклад, розширення привілеїв користувача або заборона демонстрації реклами).

Таблиця 1.1 – Порівняння існуючих аналогів

Критерій порівняння	Omegle	Yik Yak	Whisper	AnonSend
Анонімність на початку користування	Необхідно пройти реєстрацію	Необхідно пройти реєстрацію	Не вимагає реєстрації з персональними даними	Не вимагає реєстрації з персональними даними
Вибір персональної інформації	Так	Так	Так	Так
Урегульованість тематик	Так	Так	Так	Так
Адміністрування чатів	Ні	Ні	Частково	Так
Платний функціонал	Так	Так	Так	Ні

З результатів порівняння можна зробити висновок, що в більшості випадків сервіси не надають повної анонімності. Так під час бесіди користувачі можуть отримати певну особисту та персональну інформацію один одного. А розширення функціональності до відео-зв'язку в цьому

випадку не сприймається як позитивне оновлення. Більш того ряд аналогів не надає достатніх умов для підтримки безпеки користувачів саме під час чатування: відсутні фільтри спаму та жаргонізмів (нецензурних лексики та висловів). Саме тому, це може значно скоротити аудиторію таких за стосунків.

Саме тому, можна зробити висновок, що не дивлячись на велику кількість засобів для анонімного чатування актуальною задачею залишається розробка такого застосунку, що виправляв би недоліки аналогів та надавав повноцінний функціонал для проведення анонімного онлайн-спілкування.

1.3 Постановка завдання

За результатами аналізу проблемної області та дослідження існуючих аналогів для вирішення проблемних питань, сформулюємо постановку задачі.

Система потрібна для швидкого та анонімного спілкування з можливістю подальшої ідентифікації користувача.

Система дає можливість швидкого анонімного спілкування з іншими користувачами. При переході до чату дозволяє значно підвищити швидкість ідентифікації співбесідника за його Geek-кодом.

Розглянемо вимоги до розроблюваної системи за їх типом [6]-[7].

1.3.1 Функціональність системи

Функціональність системи має забезпечуватися за допомогою:

- реєстрація нового користувача;
- авторизація вже існуючого користувача;
- створення нового Geek-коду;
- збереження створеного коду у базі даних та управління ним;
- розшифрування коду, представленого у текстовому вигляді;

- надсилання повідомлення, яке буде доставлене випадковому користувачеві;
- отримання повідомлення від випадкового користувача;
- запрошення користувача до чату та обмін Geek-кодами;
- надсилання повідомлення до чату;
- отримання повідомлення з чату;
- збереження користувача до списку друзів (за Geek-кодом);
- управління списком друзів;
- фільтрація нецензурної лексики у повідомленнях;
- репорт адміністрації у разі спаму;
- управління повідомленнями;
- розгорнута довідка.

1.3.2 Вимоги до інтерфейсу

Для забезпечення ергономічності, зручності та зрозумілості інтерфейс розроблюваного мобільного застосунку повинен:

- бути зручним та інтуїтивно-зрозумілим;
- при першому користуванні продуктом користувач отримує підказки;
- пароль відображається зірочками.

1.3.3 Вимоги до даних

Вимоги до клієнтської частини:

- збереження свого Geek-коду та кодів друзів;
- збереження усіх повідомлень.

Вимоги до серверної частини:

- збереження ідентифікатору користувача та його Geek-коду;
- збереження інформації про перехід користувачів до чату.

1.3.4 Не функціональні вимоги до продукту

Мінімальні системні вимоги та програмне забезпечення, що необхідні для функціонування серверної частини мобільного застосунку:

- операційна система (ОС) – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- Web-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- пристрій з ОС Android 4.2 та вище;
- процесор - 300 MHz;
- обсяг оперативної пам'яті – не менше 256 Мб;
- підключення до мережі Internet.

1.3 Висновки за розділом 1

Виконано огляд проблеми віртуального спілкування з використанням анонімного чатування. Виділено основні шляхи для направи для розроблення нового ПЗ для анонімного чатування.

Тест – це система завдань специфічної форми зростаючої складності, що дозволяє оцінити структуру і виміряти рівень знань з певної навчальної дисципліни [9].

На сьогоднішній день користувачам пропонується ряд мобільних застосунків для анонімного тестування. Кожному з цих сервісів притаманні свої переваги та недоліки. Проаналізувавши сучасні сервіси, що надають можливості для віртуального спілкування з використанням анонімного

чатування можна зробити висновок, що більшість таких сервісів мають низку обмежень, не забезпечують безпеку своїх користувачів в повній мірі або ж взагалі не є анонімними як такі, оскільки вимагають від користувачів персональну інформацію, що потім може розповсюджуватися серед інших користувачів застосунку.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

В першому розділі був проведений аналіз проблемної області – віртуальне спілкування, а також розглянуті мобільні застосунки для анонімного чатування. На основі проведеного аналізу були сформовані основні вимоги до розроблюваної системи, були виявленні основні технічні та програмні особливості.

В результаті аналізу можна зробити висновок, що для розробки необхідно обрати мову програмування для реалізації функціональної частини, засоби для організації візуального оформлення застосунку, систему керування базами даних для розробки власної бази даних для централізованого збереження необхідної формації, веб-сервер для реалізації сумісної роботи раніше обраних технологій.

2.1 Вибір середовища проєктування

2.1.1 Вибір мови програмування

В даний час Java є найпопулярнішою мовою програмування для мобільної розробки на Android [8]-[9]. Корпорація Google активно просуває Kotlin у якості мови програмування на заміну Java. Також додатки під Android пишуть і на інших мовах.

Існують мови програмування, які використовуються для розробки Android [8]-[9]:

- Java: ця мова є офіційною мовою для розробки під ОС Android і підтримується Android Studio. Код Java виконується віртуальною машиною Java – Java Virtual Machine, яка працює на пристроях Android і інтерпретує код;

- Kotlin: останній введена мова Android; він схожий на Java, але в багатьох відносинах трохи легше;

– C/C++: Android Studio також підтримує цю мову з використанням Java NDK, що може бути зручно для ігор;

– C#: ця мова кілька зручніше для початківців на відміну від C або C++. Він підтримується деякими дуже зручними інструментами, наприклад Unity та Xamarin, які відмінно підходять для розробки ігор і крос-платформних додатків.

Проведемо порівняння мов програмування для реалізації мобільного застосунку, результати порівняння представимо в табличній формі (табл. 2.1). У якості критеріїв будемо використовувати наступні:

- швидкість роботи – умовна швидкість роботи технології та мови програмування за оцінками експертів;
- плагіни та розширення – у загальному випадку програмні бібліотеки, модулі чи плагіни, що можуть спростити та скоротити час розробки;
- парадигми програмування – загальна можливість мови підтримувати парадигми програмування, що може скоротити час розробки та зробити код більш зрозумілим для подальшої підтримки;
- спільнота – критерій для визначення кількості активних розробників, що використовують мову програмування та кількості ресурсів присвячених мові програмування.

Таблиця 2.1 – Порівняння мов програмування

Критерій	Java	Kotlin	C/C++	C#
Швидкість роботи	+	±	±	–
Плагіни та розширення	++	±	+ (залежить від середовища розробки)	+
Парадигми програмування	±	+	+	±
Спільнота	++	+	±	±

За результатами порівняння можна зробити висновок, що не дивлячись на тривалий час Java залишається кращою альтернативою, адже має і розширену спільноту, і набір розширень, що спрощують розробку. Для написання нашого застосунку за обраною темою розробки цілком достатньо вибрати мову Java.

2.1.2 Вибір середовища розробки

Android Studio – інтегроване середовище розробки від Google, за допомогою якого розробникам стають доступні інструменти для створення застосунків на платформі ОС Android (рис. 2.1) [10]. Android Studio можна встановити на Windows, Mac і Linux. Android Studio створювалася на базі IntelliJ IDEA. У ній присутні макети для створення UI, з чого зазвичай починається робота над додатком. В Studio містяться інструменти для розробки рішень для смартфонів та планшетів, а також нові технологічні рішення для Android TV, Android Wear, Android Auto, Glass та додаткові контекстуальні модулі [10].

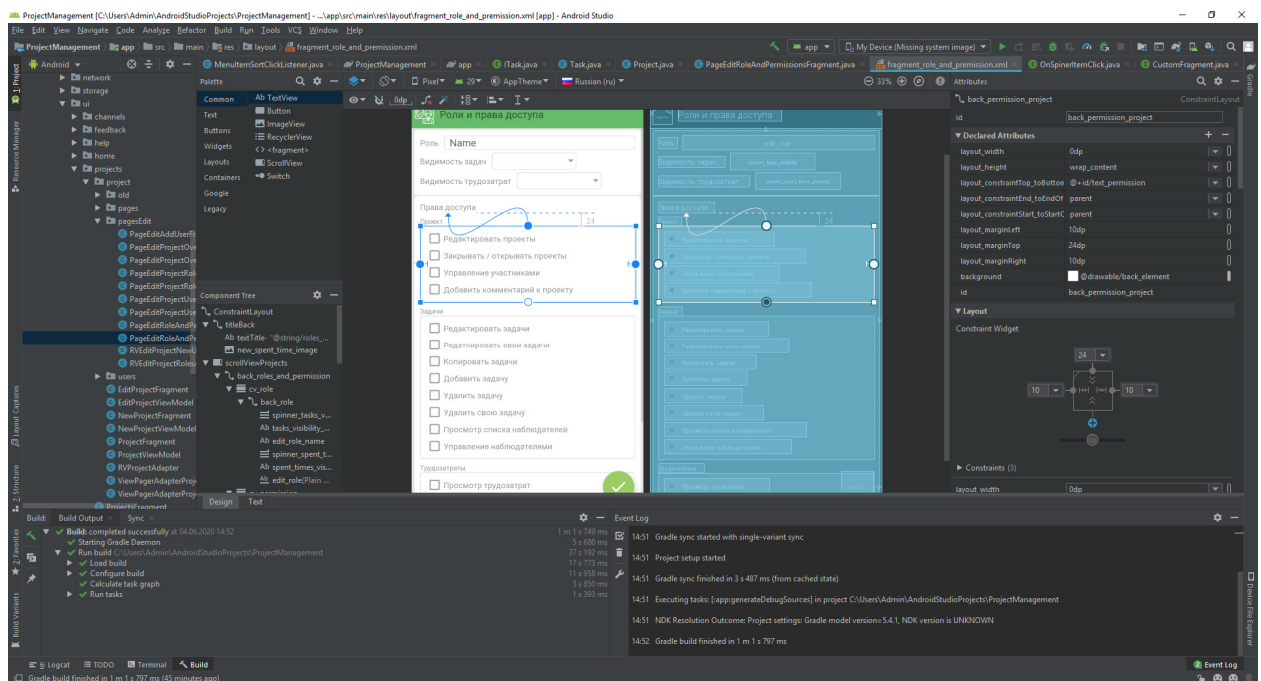


Рисунок 2.1 – Середовище розробки Android Studio

Середовище Android Studio призначене як для невеликих команд розробників мобільних додатків (навіть в кількості однієї людини), або ж великих міжнародних організацій з GIT або іншими подібними системами управління версіями.

Переваги:

- мови, які підтримуються. Середовище розробки підтримує роботу з декількома мовами програмування, до яких відносяться найпопулярніші - C/C++, Java і ще один молодий мову Kotlin;
- безкоштовність. Eclipse Java IDE поширюється безкоштовно;
- рефакторинг. Зручний і потужний рефакторинг вже готового коду;
- бібліотека. Досить велика бібліотека з готовими шаблонами і компонентами для розробки ПО;
- тестування. Великий набір засобів інструментів для тестування кожного елемента додатки;
- документація. Є багато посібників по використанню Android Studio для недосвідчених/початківців розробників;
- вбудований емулятор. Зручне і швидке тестування в вбудованому Android-емуляторі.

Недоліки:

- системна вимога. Для комфортної роботи в середовищі знадобиться потужний комп'ютер;
- неможливість написати серверні проєкти на мові Java для ПК, Android пристроїв.

Eclipse Java IDE – відкрита інтегрована середовище розробки на мові програмування Java. Розробники компанії регулярно покращували Eclipse, випускаючи оновлення, створюючи нові інструменти та впроваджуючи корисні для програмістів функції [8]-[9].

У цьому середовищі можна підключати додаткові модулі, що дозволяють розширити її функціонал. Наприклад, для розробки проєктів на

мовах C/C++ необхідно підключити модуль C/C++ Development Tools, а для написання програми на Java потрібно Java Development Tools.

Дане середовище досить зручна завдяки русифікованому інтерфейсу і документації, де можна знайти дуже багато корисної інформації.

Конфігурацію Eclipse розробник може формувати під свої потреби і на свій розсуд, роблячи середу максимально зручною для себе. У ній чудово реалізована функція групової розробки, що по достоїнству оцінять групи програмістів, які працюють над одним проектом.

Переваги:

- модулі. Eclipse можна гнучко налаштувати під себе завдяки простій розробці плагінів;

- мови, які підтримуються. Середовище розробки підтримує роботу з декількома мовами програмування, до яких відносяться найпопулярніші - C/C++, Java;

- безкоштовність. Eclipse Java IDE поширюється безкоштовно;

- кросплатформеність. У будь-якої програми, написаного на Java, за замовчуванням є ця перевага. Eclipse можна запустити на будь-якій платформі, на якій встановлена Java;

- системна вимога. Можна комфортно працювати в середовищі навіть на слабкому комп'ютері;

- workspace. В Eclipse можна відкрити кілька проектів в одному workspace, що значною мірою знижує з модульними проектами або частковим перенесенням коду з одного проекту в інший;

- документація. Eclipse - зрілий продукт, популярний у всьому світі, тому у нього є велика база документації, в якій з імовірністю 90% знайдеться відповідь на будь-яке питання;

- оформлення інтерфейсу.

Недоліки:

- не інтуїтивний інтерфейс;

- модулі. При наявності великої кількості несумісних плагінів IDE може впасти, і буде потрібна переустановка;
- налагодження. Функція налагодження не так досконала, як у інших платних IDE інструментів;
- проблеми з управлінням версіями, зокрема з Git;
- неповна інтеграція з Maven і Gradle.

Xamarin – це фреймворк з використанням мови C# для розробки кроссплатформених додатків (для iOS, Android, Windows Phone).

Xamarin дозволяє створювати одну єдинственную логіку додатка із застосуванням C# і .NET відразу для всіх трьох платформ - Android, iOS, Windows Mobile.

Основний інструмент, за допомогою якого компанія Xamarin пропонує нам розробляти програми для iOS, Android та Windows Phone, - це Xamarin, яка працює на Windows або Mac OS X. Також можна вести розробку завдяки розширенням в середовищі Microsoft Visual Studio (рис. 2.2).

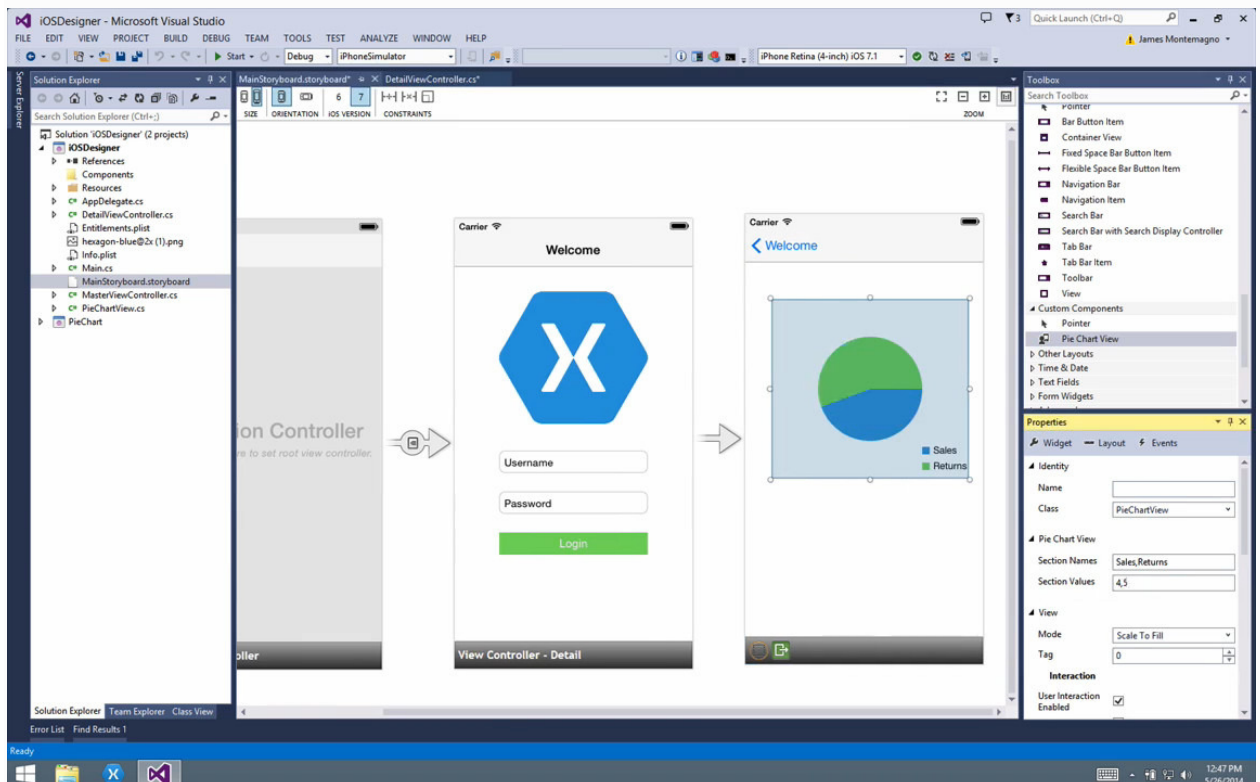


Рисунок 2.2 – Середовище розробки Visual Studio з плагіном Xamarin

Xamarin додатки містять у собі виконавчу середу Mono.

При запуску Xamarin додатки виконавче середовище Mono транслює скомпільований C# код в команди операційної системи (iOS або Android або Windows).

Функціонально платформа Xamarin представляє ряд субплатформ. Зокрема:

- Xamarin.Android - бібліотеки для створення додатків на ОС Android;
- Xamarin.Mac - бібліотеки для створення додатків на Mac OS X;
- Xamarin.iOS - бібліотеки для створення додатків для iOS;
- компілятор C #;
- .NET Framework;
- інструменти IDE (вбудовані в Visual Studio для Mac OS i Windows).

Переваги:

– кросплатформеність. В процесі розробки створюється єдиний код для всіх платформ;

– API. Xamarin надає прямий доступ до нативним API кожної платформи;

– документація. Для Xamarin доступна відмінно організована документація з практичними прикладами, фрагментами коду і покрокові інструкції;

– сумісність з пристроями. Якщо потрібно, щоб додаток отримувало дані про місцезнаходження користувача, показання гіроскопа, акселерометра або інших вбудованих датчиків, то Xamarin стане відмінним вибором;

– потужне тестування. Доступного середовища Test Cloud, що дозволяє емулювати більше 2000 пристроїв;

– оформлення інтерфейсу.

Недоліки:

– розмір додатків. Розмір додатків, написаних за допомогою Xamarin, зазвичай трохи перевищує обсяг нативних аналогів;

- не підходить для додатків з високопродуктивною графікою. Багатий користувацький інтерфейс або ж складні анімації з невеликою кількістю загального коду роблять Xamarin непридатним;
- складнощі з інтеграцією.

2.1.3 Порівняння середовищ розробки

У таблиці 2.2 наведено порівняння середовищ розробки Android додатків. Серед характеристик для порівняння виділімо:

- графічний інтерфейс користувача (GUI) – комплексна оцінка зрозумілості та інтуїтивності GUI середовища розробки для користувача;
- продуктивність – навантаження на ОС під час розробки;
- служба підтримки – комплексна оцінка для визначення рівня технічної підтримки середовища, його розробників та спільноти, що використовує середовище;
- підтримка української мови в GUI;
- підсвічування синтаксису;
- рефакторинг;
- автодоповнення коду;
- пошук по проєкту;
- налагодження;
- тестування додатків;
- мови, які підтримуються – визначає які мови підтримує середовище розробки, що в майбутньому дозволить розширювати функціонал при оновленні застосунку із використанням стороннього коду.

Для розробки мобільного застосунку для анонімного чатування було обране середовище розробки Android Studio, оскільки дане середовище більш зрозуміло і в ньому зручно працювати. Studio підтримує Java. У цьому середовищі також є повна можливість розробити інтерфейси під Android.

Таблиця 2.2 – Порівняння середовищ розробки

	Android Studio	Eclipse	Xamarin (Visual Studio)
GUI	++	++	++
Продуктивність	+	++	+
Служба підтримки	+	+	+
Українська мова	–	+	+
Підсвічування синтаксису	+	+	+
Рефакторинг	+	+	+
Автодоповнення коду	+	+	+
Пошук по проєкту	+	+	+
Налагодження	+	+	+
Тестування додатків	+	+	+
Мови, які підтримуються	Java Kotlin C++	Java C++	C#

2.2 Загальна структура додатку

Для побудови архітектурного каркасу додатку використовують схему проєктування Model-view-controller (MVC). Це схема використання декількох шаблонів проєктування, за допомогою яких модель даних програми, користувацький інтерфейс і взаємодія з користувачем розділені на три окремих компоненти.

Логіка відображення (front-end) – відповідає за збір даних від користувачів в різних формах та обробку їх у відповідності зі специфікацією.

Логіка управління даними (back-end) – обробляє дані, отримані від користувача для передачі на наступний рівень.

Бізнес-логіка – сукупність правил, принципів, залежностей поведінки об'єктів предметної області. Це реалізація предметної області в інформаційній системі.

Особливості розробки архітектури мобільного застосунку на основі предметної області:

- можливість більш докладного опису логіки взаємодії об'єктів предметної області;
- можливість гнучкої зміни логіки роботи додатку;
- відсутність надлишкового коду в кінцевій реалізації;
- поділ логіки відображення, логіки управління даними і бізнес-логіки.

Недоліки розробки архітектури мобільного застосунку на основі предметної області:

- велика трудомісткість при створенні проєкту;
- більш жорсткі вимоги до кваліфікації розробника.

На рисунку 2.3 показано, який вигляд має архітектура мобільного застосунку системи на основі предметної області, а на рисунку 2.4 – на основі використання фреймворків, але у даному випадку тільки розробка на основі предметної області може задовольнити потреби розробки.

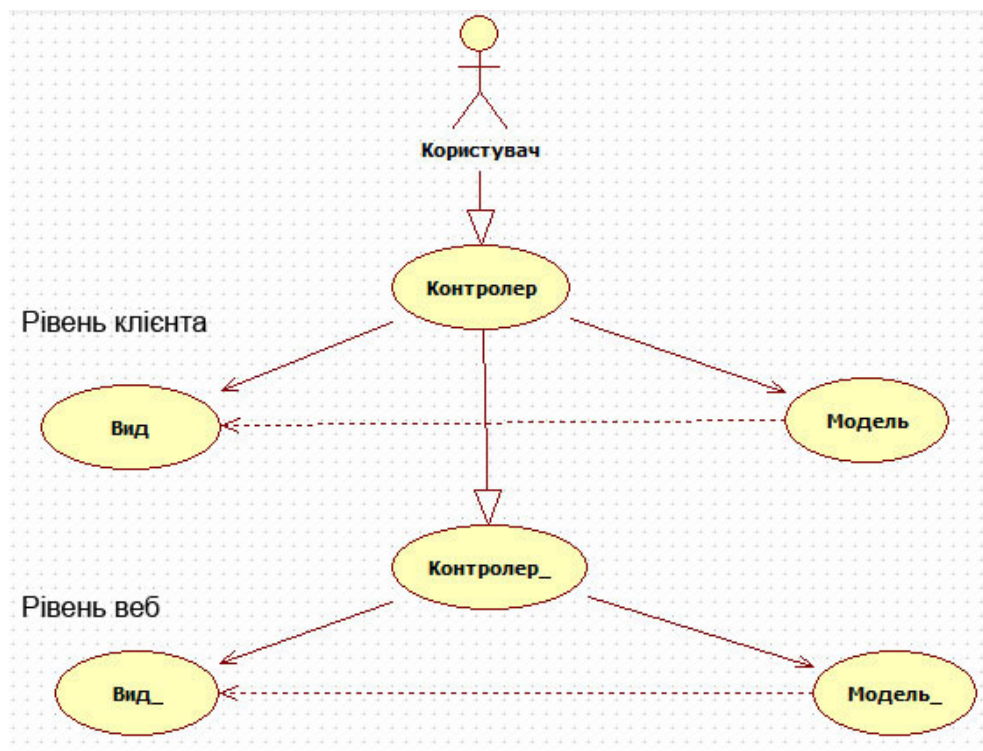


Рисунок 2.3 – Архітектура системи «AnonSend» на основі предметної області

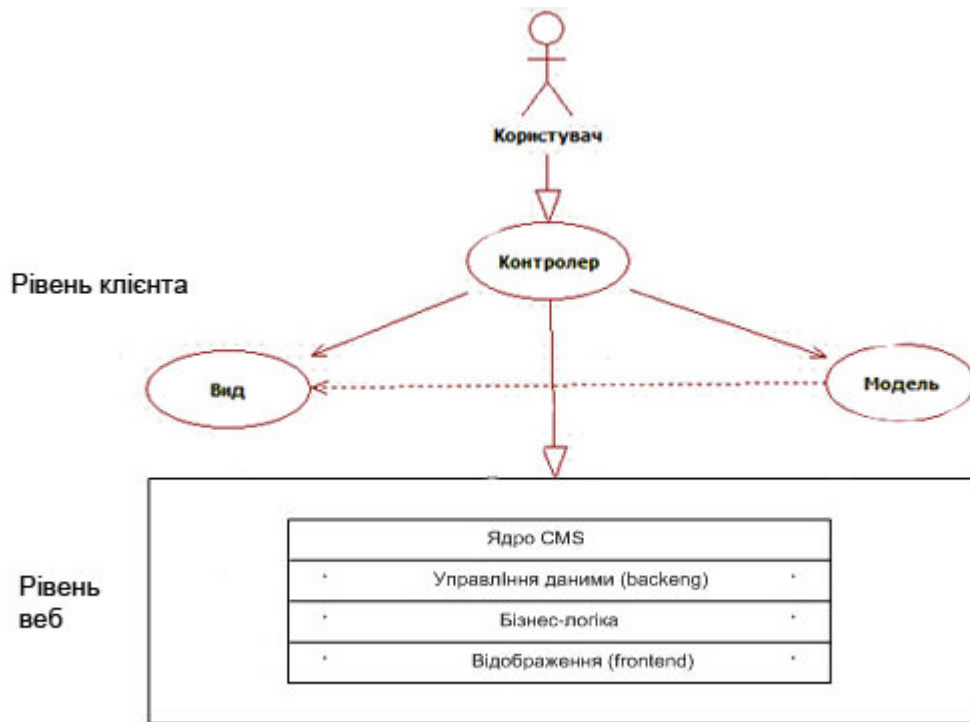


Рисунок 2.4 – Архітектура системи на основі використання фреймворків

2.3 Висновки за розділом 2

Для програмної реалізації мобільного застосунку для анонімного чатування було проаналізовано сучасні засоби для мобільної розробки на основі виявлених особливостей та архітектури застосунку, що розробляється в рамках даної дипломної кваліфікаційної роботи. Було виявлені необхідні засоби. Отже, для розробки практичної частини роботи обрано:

- мова програмування Java;
- середовище розробки Android Studio.

Проаналізувавши обрану клієнт-серверну архітектуру можна зробити висновок: застосування, що розроблюється є надійним, зручним у використанні, має відкриту структуру, що відкриває широкі можливості по адаптації та масштабуванню.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проєктування моделей системи

Unified Modeling Language (UML) – уніфікована мова моделювання, що використовується у парадигмі об’єктно-орієнтованого програмування та є невід’ємною частиною уніфікованого процесу розроблення ПЗ. UML заснована на діаграмах, що дають можливість представити систему у такому вигляді, щоб її можна було легко перевести у програмний код.

Діаграми класів – це різновид статичної структурної діаграми, що описують структуру системи, демонструють класи систем, їх атрибути, методи і залежності між класами [11]-[12].

Діаграма класів, розроблена для застосунку, що розробляється представлена на рисунку 3.1.

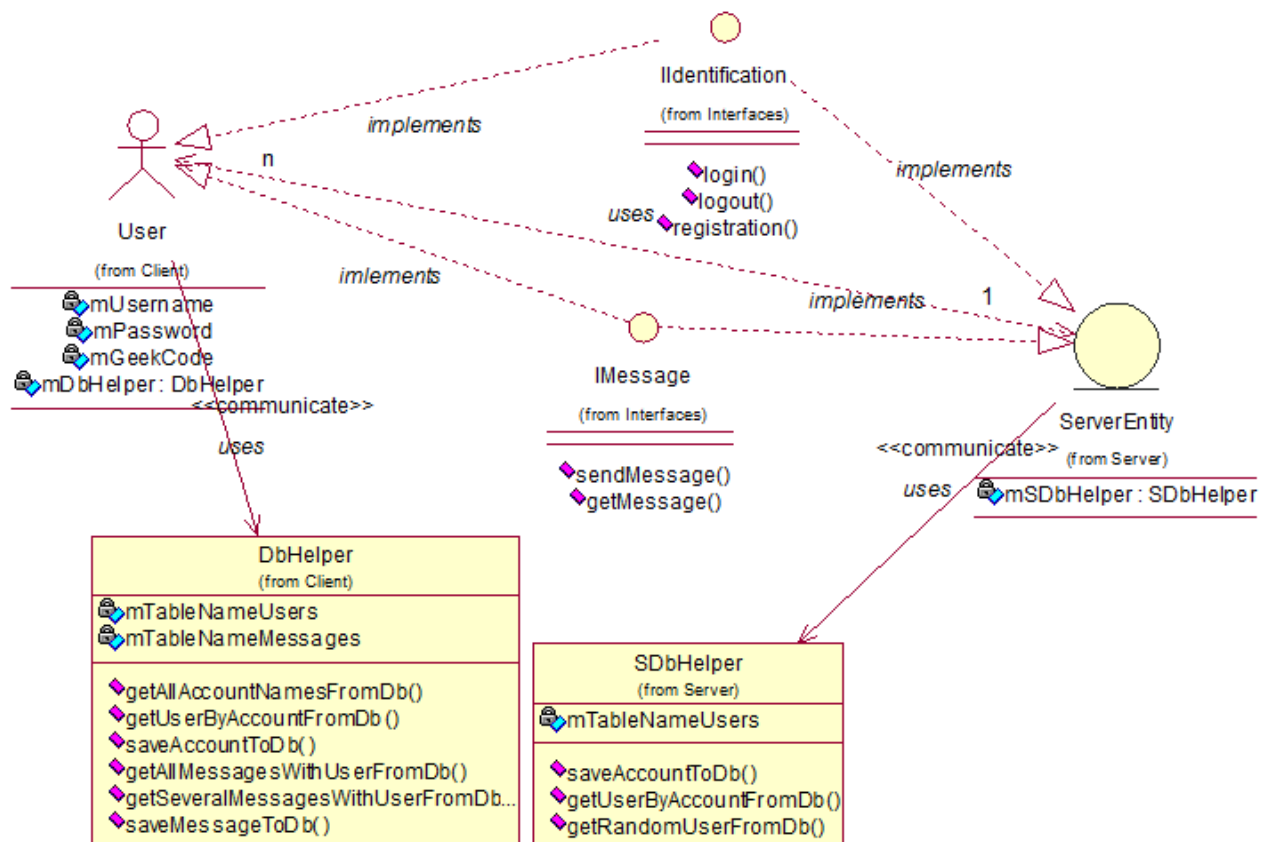


Рисунок 3.1 – Діаграма класів системи

Діаграми пакетів – це різновид структурної діаграми, основним вмістом якої є пакети і відношення між ними. Ці діаграми слугують, в першу чергу, для організації елементів в групі по яким-небудь ознакам з метою спрощення структури і організації роботи з моделлю системи [11]-[12].

Застосунок буде складатися з трьох пакетів, що представлені на рисунку 3.2.

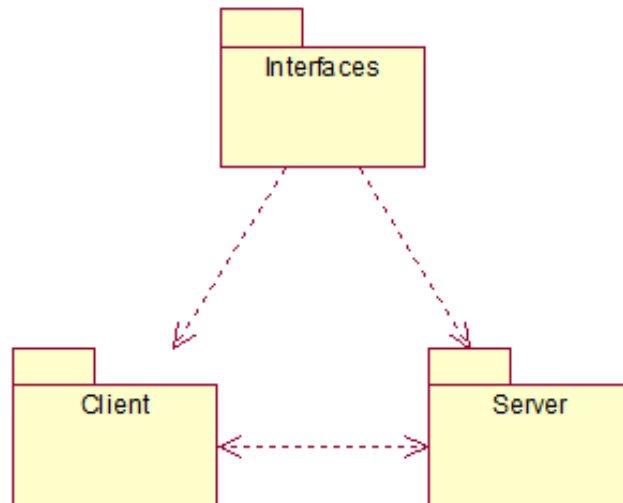


Рисунок 3.2 – Діаграма пакетів системи

Діаграма компонентів – це статична структурна діаграма, що показує розбиття програмної системи на структурні компоненти і зв'язки (залежності) між компонентами. В якості фізичних компонент можуть виступати файли, бібліотеки, модулі, виконавчі файли, пакети і т.п. [11]-[12].

На рисунку 3.3 показана схема компонентів системи.

Діаграми прецедентів застосовуються для моделювання вигляду системи з точки зору прецедентів (або варіантів використання). Даний вид діаграм використовується для моделювання динамічних аспектів системи, так само, як і діаграми станів та діяльності [11]-[12].

На рисунках 3.3-3.10 зображено діаграми прецедентів для системи. Для них приведено базові та альтернативні потоки виконання.

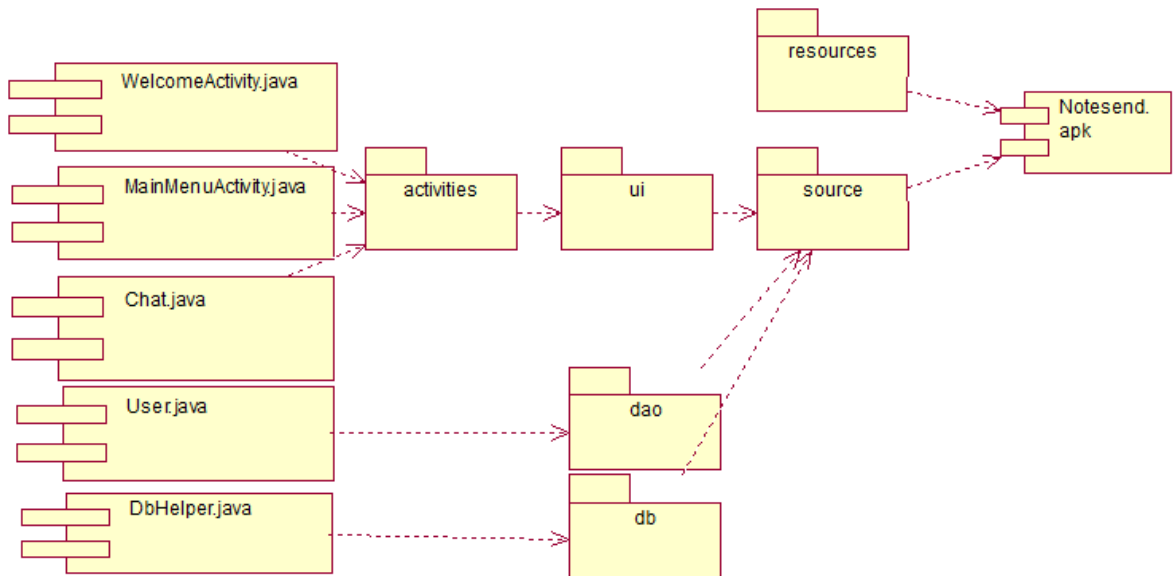


Рисунок 3.3 – Діаграма компонентів системи

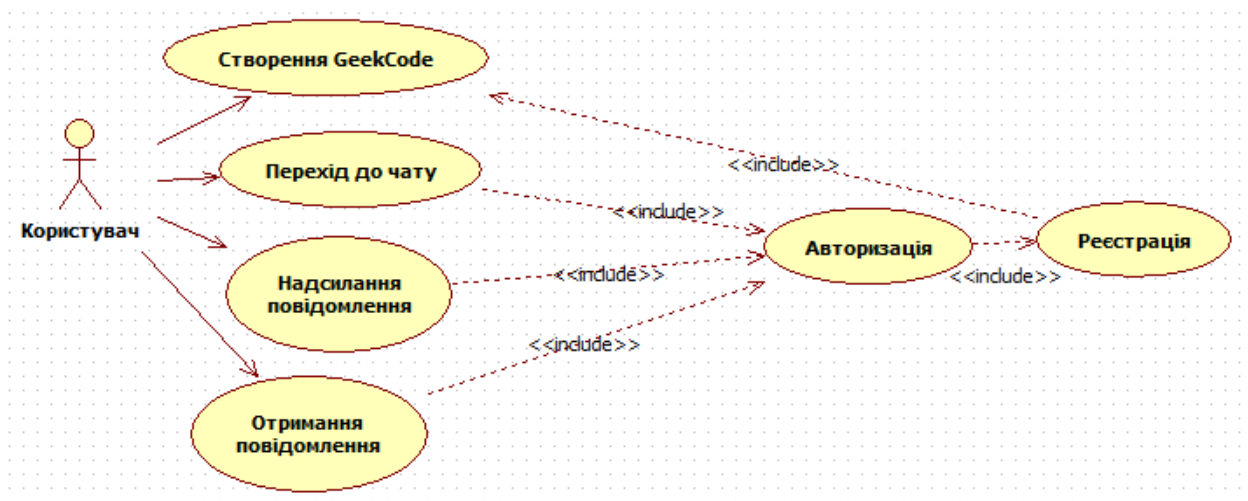


Рисунок 3.4 – Загальна діаграма прецедентів

Базовий сценарій виконання основними кроками (В).

В1 Користувач реєструється (авторизується).

В2 Користувач створює власний GeekCode.

В3 Користувач надсилає та отримує повідомлення.

В4 Користувач переходить до чату.

На рисунках 3.5-3.7 зображено діаграми прецедентів для системи.

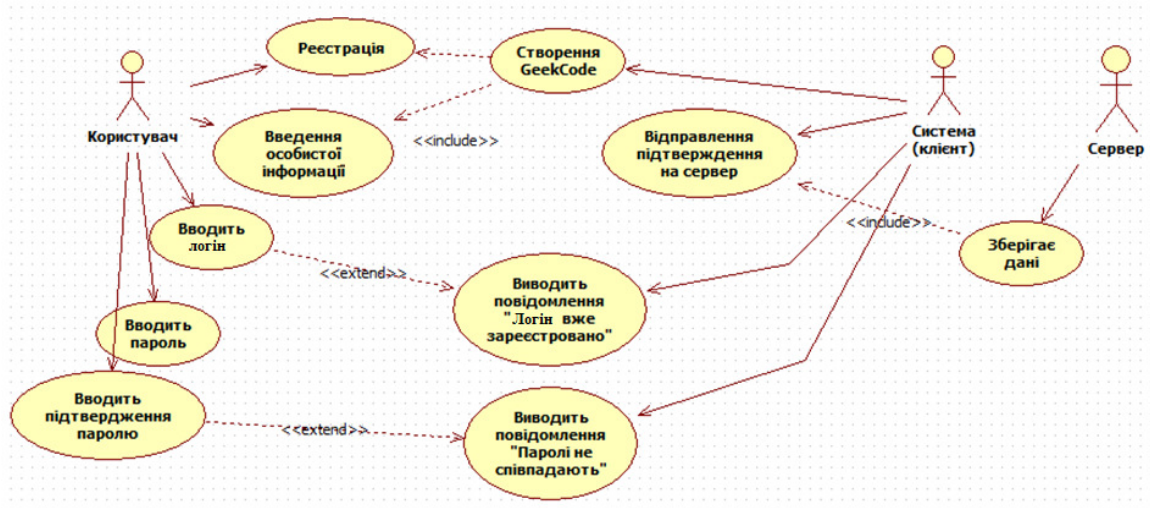


Рисунок 3.5 – Діаграма прецедентів «Реєстрація»

Базовий сценарій.

V1 користувач входить до меню реєстрації.

V2 Користувач вводить логін.

V3 Користувач вводить пароль.

V4 Користувач вводить підтвердження пароля.

V5 Користувач вводить особисту інформацію.

V6 Система створює GeekCode.

V7 Система відправляє інформацію на сервер.

V8 Сервер зберігає реєстраційні дані.

Альтернативні потоки (A).

Введений логін вже зареєстровано.

A1 Система виводить повідомлення «логін вже зареєстровано»

Повернення до пункту V2 базового потоку.

Введений пароль не співпадає з підтвердженим.

A1 Система виводить повідомлення «Паролі не співпадають».

Повернення до пункту V3 базового потоку.

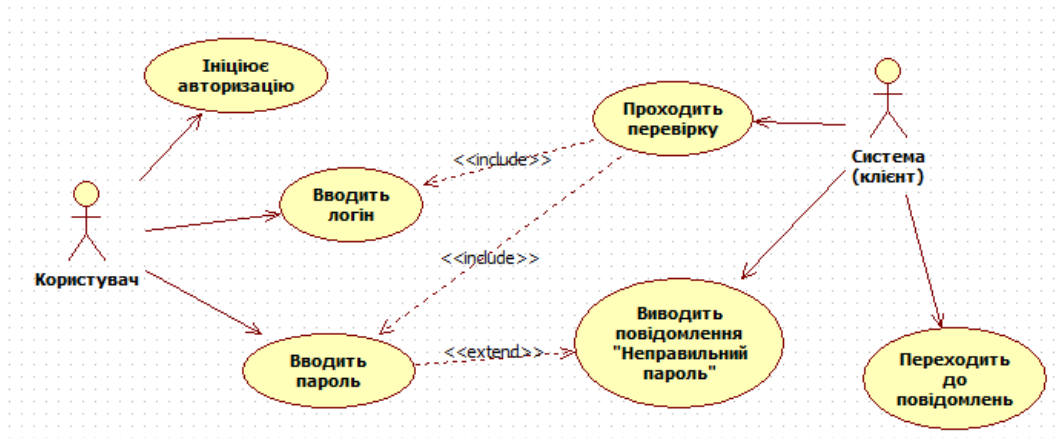


Рисунок 3.6 – Діаграма прецедентів «Авторизація»

Базовий сценарій.

V1 Користувач ініціює авторизацію.

V2 Користувач вводить логін.

V3 Користувач вводить пароль.

V4 Система проводить перевірку.

V5 Система виводить екран з повідомленнями.

Альтернативні потоки.

Введений пароль невірний.

A1 Система виводить повідомлення «Пароль введено не вірно» .

Повернення до пункту V3 базового потоку.

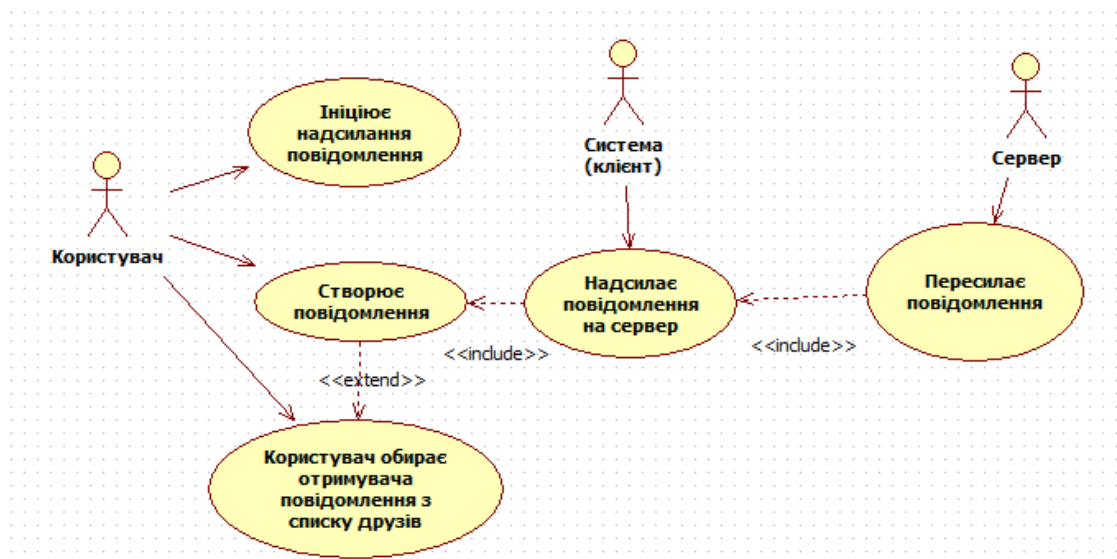


Рисунок 3.7 – Діаграма прецедентів «Надсилення повідомлення»

Базовий сценарій.

V1 Користувач ініціює надсилання повідомлення.

V2 Користувач створює повідомлення.

V3 Користувач може обрати отримувача повідомлення.

V4 Система надсилає повідомлення на сервер.

V5 Сервер пересилає повідомлення.

На рисунках 3.8-3.9 зображено діаграми прецедентів для системи.

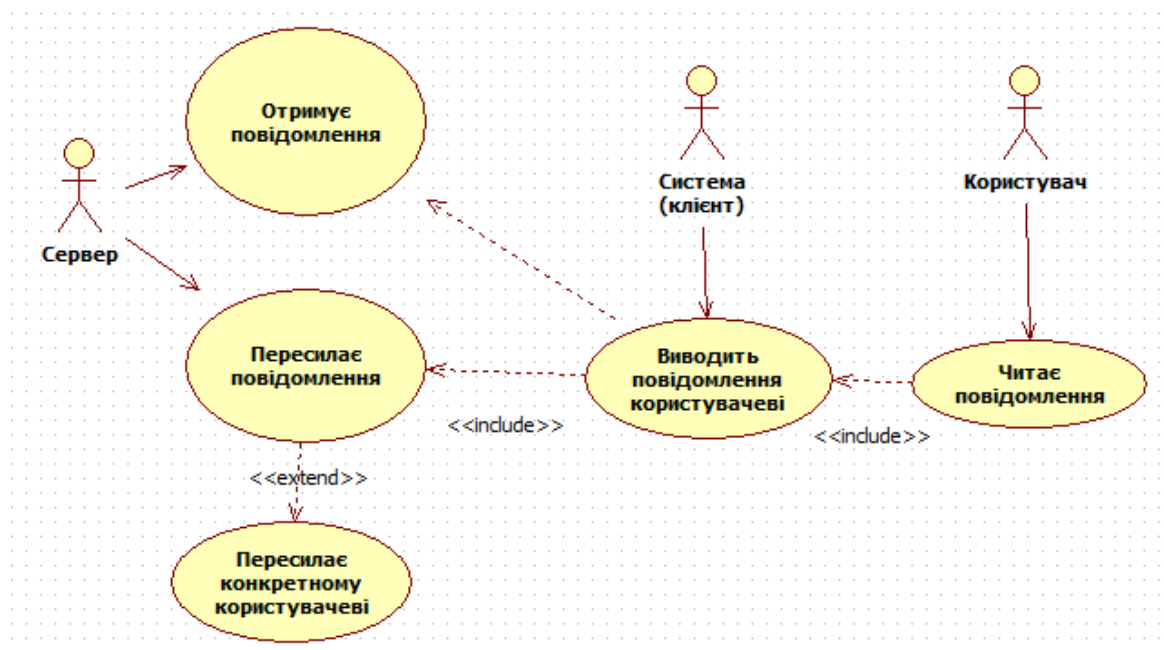


Рисунок 3.8 – Діаграма прецедентів «Отримання повідомлення»

Базовий сценарій.

V1 Сервер отримує повідомлення.

V2 Сервер пересилає повідомлення випадковому користувачеві або (якщо вказаний отримувач) – конкретному користувачеві.

V3 Система виводить повідомлення користувачеві.

V4 Користувач читає повідомлення.

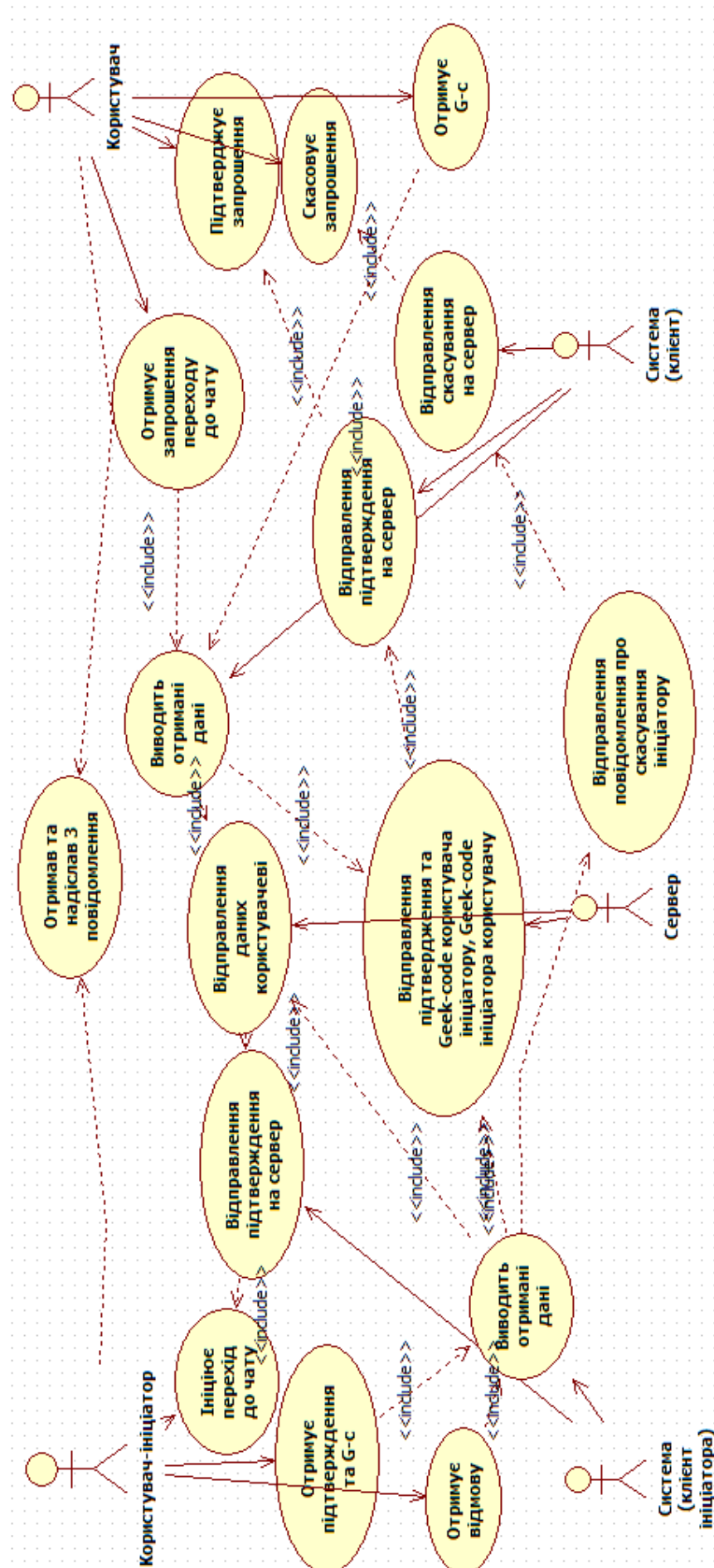


Рисунок 3.9 – Діаграма прецедентів «Перехід до чату»

Базовий сценарій.

B1 Користувач та користувач-ініціатор відправили та отримали по 3 повідомлення один від одного.

B1 Користувач-ініціатор ініціює перехід до чату.

B2 Система користувача-ініціатора відправляє інформацію на сервер.

B3 Сервер відправляє запрошення користувачеві.

B4 Система користувача виводить запрошення.

B5 Користувач отримує запрошення переходу до чату.

B6 Користувач підтверджує запрошення.

B7 Система користувача відправляє підтвердження на сервер.

B8 Сервер відправляє підтвердження та GeekCode користувача користувачу-ініціатору та GeekCode користувача ініціатора користувачу.

B9 Система користувача-ініціатора виводить отримані дані.

B10 Користувач-ініціатор отримує підтвердження та GeekCode користувача.

B11 Система користувача виводить отримані дані.

B12 Користувач отримує GeekCode користувача-ініціатора.

Альтернативні потоки.

Користувач скасовує запрошення.

A1 Система користувача відправляє скасування на сервер.

A2 Сервер відправляє повідомлення про скасування користувачу-ініціатора.

A3 Система користувача-ініціатора виводить отримані дані.

A4 Користувач-ініціатор отримує відмову.

3.2 Фізичне проєктування

Діаграми станів відображають всі можливі стани, в яких може знаходитись конкретній об'єкт, а також зміни стану об'єкту, які відбуваються в результаті впливу деяких подій на цей об'єкт [11]-[12].

На рисунку 3.10 зображено діаграму станів користувача у системі. На ній відображено переходи між станами користувача.

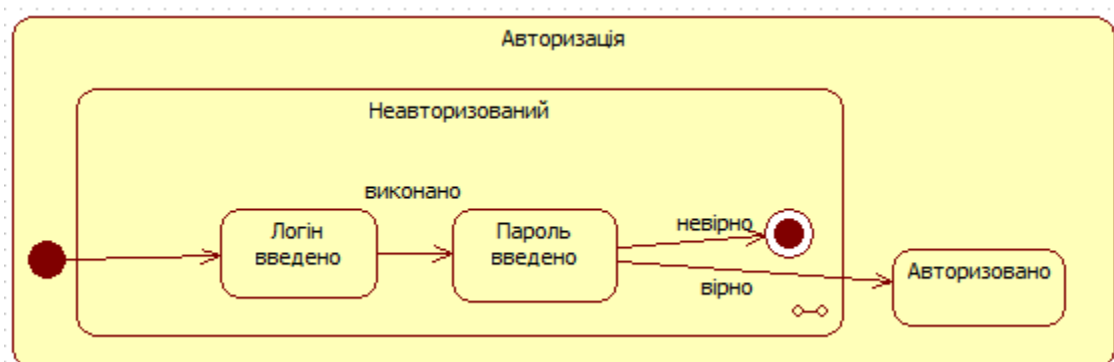


Рисунок 3.10 – Діаграма станів

Діаграма діяльності показує переходи між видами діяльності. Модель видів діяльності може подавати в графічній формі потік подій для прецеденту [11]-[12].

Діаграма діяльності переходу користувачів до чату показана на рис. 3.11.

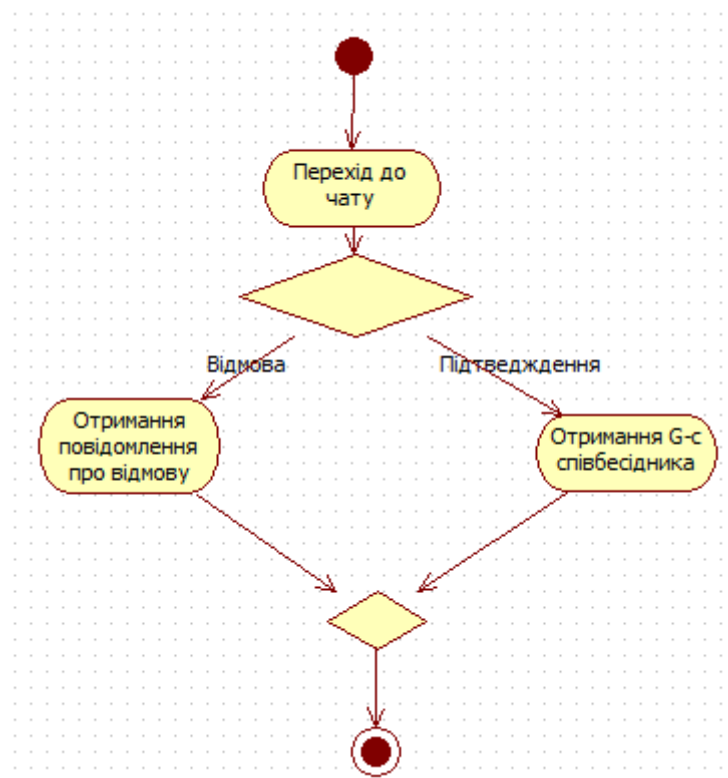


Рисунок 3.11 – Діаграма діяльності

Діаграма кооперації призначена для специфікації структурних аспектів взаємодії. Головна особливість діаграми кооперації полягає в можливості графічно уявити не тільки послідовність взаємодії, але і всі структурні відносини між об'єктами, що беруть участь в цій взаємодії [11]-[12].

На рисунках 3.12-3.16 зображено діаграми кооперації показано взаємодію користувача та системи коли виконуються операції: авторизація, реєстрація, надсилання повідомлення, отримання повідомлення, а також перехід до чату. Всі інші операції у системі здійснюються аналогічним чином.

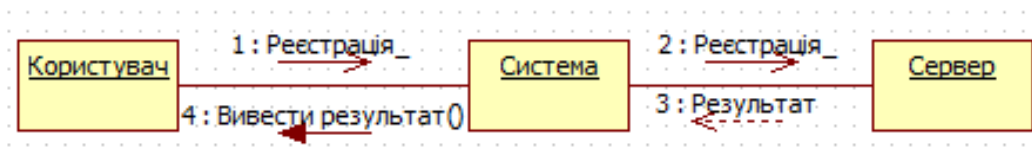


Рисунок 3.12 – Діаграма кооперації «Реєстрація»

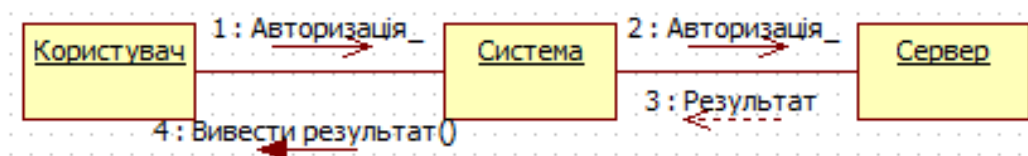


Рисунок 3.13 – Діаграма кооперації «Авторизація»

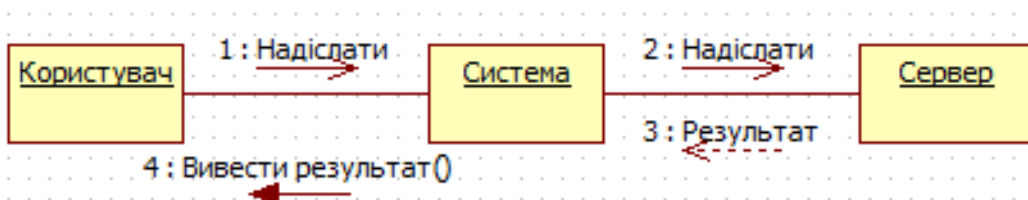


Рисунок 3.14 – Діаграма кооперації «Надсилання повідомлення»

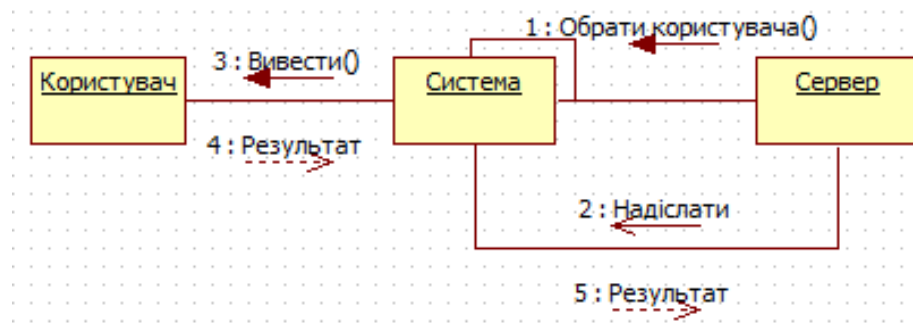


Рисунок 3.15 – Діаграма кооперації «Отримання повідомлення»

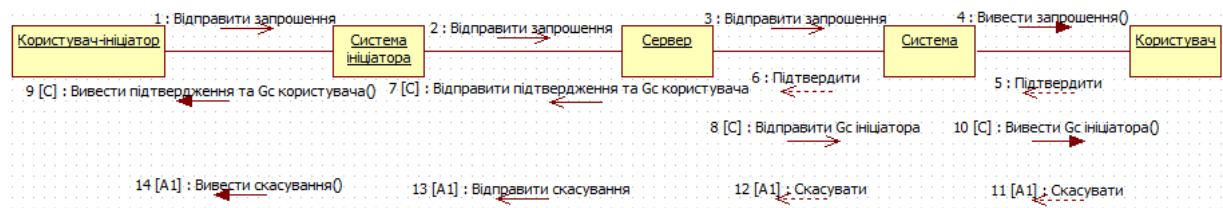


Рисунок 3.16 – Діаграма кооперації «Перехід до чату»

На рисунку 3.17 зображено в якій послідовності виконуються дії користувачів та системи під час переходу до чату.

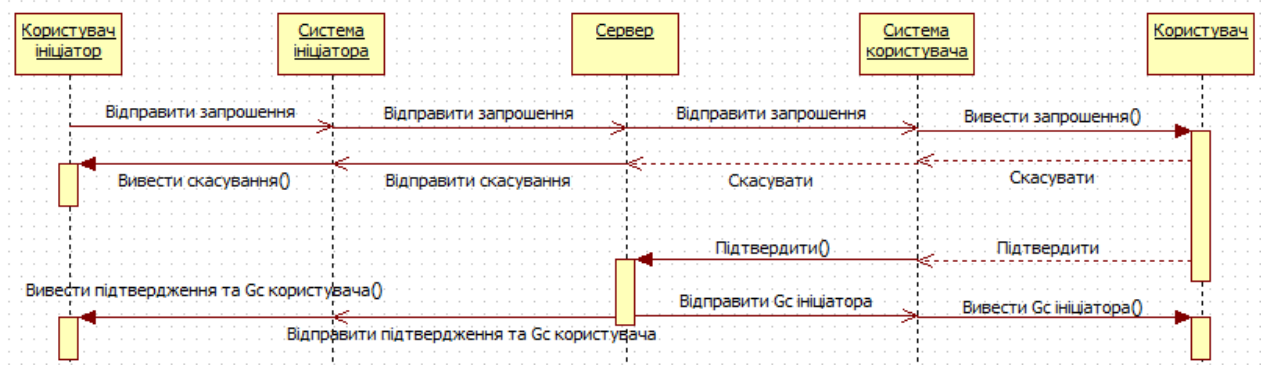


Рисунок 3.17 – Діаграма послідовності «Перехід до чату»

3.3 Проектування інтерфейсу

GUI – сукупність засобів і методів взаємодії між елементами системи. Залежно від контексту, поняття застосовує як до окремого елемента (інтерфейс елемента), так і до зв'язків елементів (інтерфейс сполучення елементів).

Інтерфейс користувача – сукупність засобів, за допомогою яких користувач спілкується з різними пристроями:

- інтерфейс командного рядка – інструкції програмою або пристроєм здійснюються шляхом введення з клавіатури текстових рядків;
- GUI – управління програмними функціями реалізовано графічними елементами екрану.

Багато кількісних та евристичних методів використовуються для аналізу та вивчення інтерфейсів. Кількісні методи часто можуть зводити спірні питання до простих обчислень. Ще однією, більш важливою, перевагою даних методів є те, що вони допомагають зрозуміти найважливіші аспекти взаємодії людини з машиною.

При запуску додатку початковим екраном є екран авторизації/реєстрації (рис. 3.18).

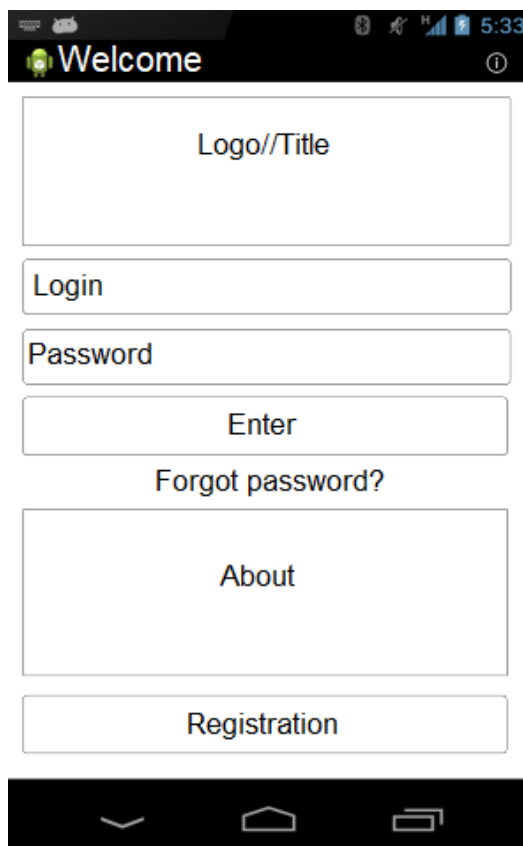


Рисунок 3.18 – Початковий екран авторизації/реєстрації

Натиснувши іконку додаткової інформації, користувач отримує довідку про систему. Тут та далі ця іконка буде використовуватись у якості довідки. Довідку можна переглянути для будь-якого екрану. Це стандартний діалог, тому він не зображений на рисунках.

Якщо користувач не зареєстрований він потрапляє на екран реєстрації (рис. 3.19).

The screenshot shows a mobile application interface for registration. At the top, there is a status bar with icons for signal, battery, and time (5:33). Below the status bar is a header bar with a back arrow, a robot icon, the text 'Registration', and an information icon. The main content area contains a box with 'Rules + Information', a 'Registration data' button, a 'User data' section with 'Login', 'Password1', and 'Password2' input fields, a checkbox labeled 'Know about rules', and a 'Finish of registration' button. At the bottom, there is a black navigation bar with three white icons: a chevron, a house, and a list.

Рисунок 3.19 – Початок реєстрації

Користувач може одразу завершити реєстрацію (GeekCode залишиться пустим) або створити картку (GeedCode) відповідаючи на запитання про його інтереси (рис. 3.20).

Оскільки це питання не відноситься до жодної категорії питань, то воно винесене перед категоріями, одночасно користувач знайомиться з інтерфейсом опитування та буде швидше реагувати на наступні запитання.

Жодне питання не являється обов'язковим, тому кожне питання користувач може пропустити.

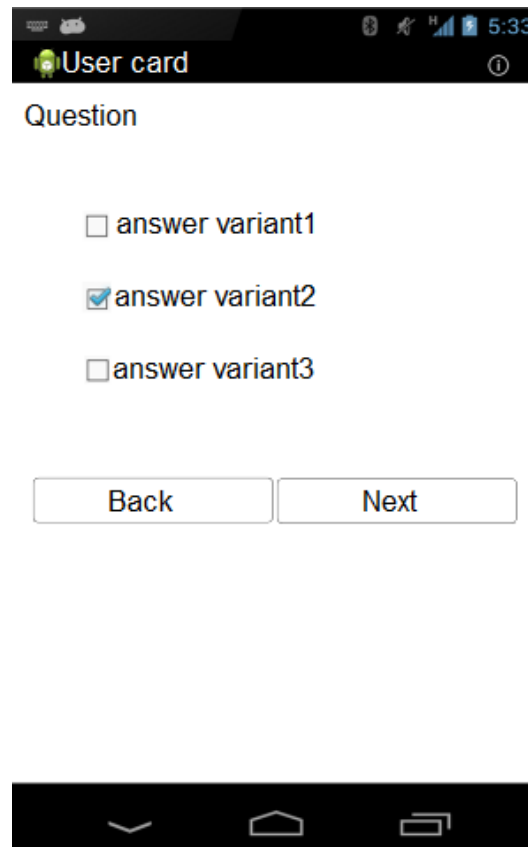


Рисунок 3.20 – Початок збору інформації про користувача під час створення картки

Кнопки не винесені до Action Bar, оскільки на пристроях з малою діагоналлю екрану до бару вміщується тільки 2 дії, щоб отримати доступ до інших користувач повинен розгорнути список дій, що не помістилися натисканням на додаткову кнопку на барі.

Коли користувач відповість на усі запитання – він повертається до змісту питань та система виводить стандартний діалог з повідомлення про завершення створення картки. Діалог неможливо відмінити натисканням на кнопку пристрою «Назад», тому єдиною можливою дією являється натискання на кнопку «Далі». Після натискання користувач потрапляє на екран відправлення анонімного повідомлення (рис. 3.21).

Користувачу не обов'язково відповідати на усі запитання. Він у будь-який момент може натиснути на кнопку у вигляді галки та завершити процес створення картки.

Після успішної реєстрації/авторизації користувач потрапляє до робочого середовища (рис. 3.21). У Action Bar представлені такі дії (зліва направо): відкрити свій профіль; перейти до створення нового повідомлення, яке буде доставлене випадковому користувачу; відкрити історію повідомлень (історія переписки з друзями зберігається постійно, а з анонімами – добу від часу останнього повідомлення); відкрити список друзів; переглянути довідку по екрану.



Background

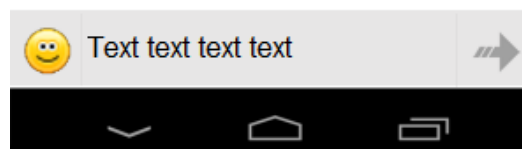


Рисунок 3.21 – Екран відправлення повідомлення випадковому користувачу

Користувач вводить повідомлення та відправляє його. Можна додати смайли.

З будь-якого екрану робочого середовища користувач може жестом зліва-направо викликати сайдбар (рис. 3.22).

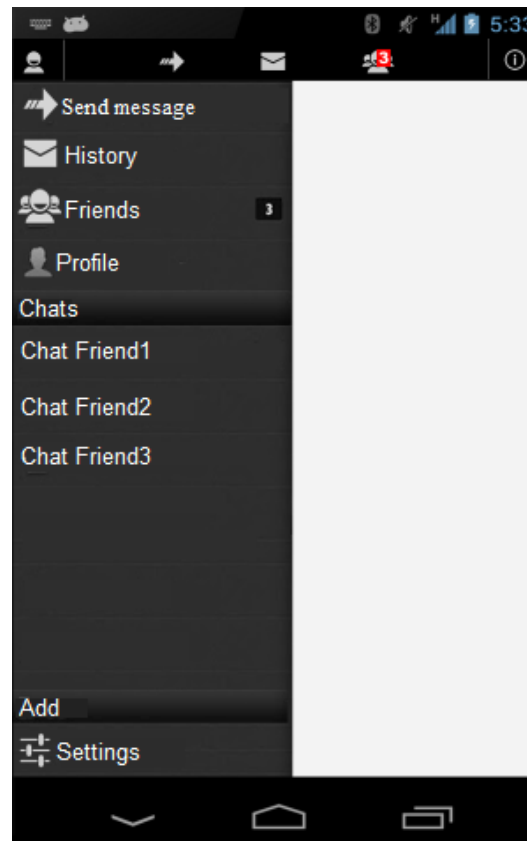


Рисунок 3.22 – Демонстрація сайд бару

3.4 Висновки за розділом 3

Запропоновано схему функціонування системи в цілому. Розроблена схема відповідає архітектурі клієнт-сервер і дозволяє виконувати всі поставлені перед системою функції, основними з яких є: анонімне онлайн чатування.

Крім того, розглянуті схеми функціонування окремо серверної та клієнтської частин розроблюваної системи.

Запропоновані способи функціонування як системи в цілому, так і частин окремо, дозволяє забезпечити логічно вірну і програмно раціональну роботу програмних засобів мобільного застосунку.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Опис застосування

Розроблений мобільний застосунок складається з серверної і клієнтської частини. Клієнтська частина за організацію користувацького інтерфейсу для проходження тестування, при використанні користувача з відповідними правами доста клієнтський інтерфейс дозволяє додавати нові тести та змінювати старі.

Серверна частина в свою чергу відповідає за перевірку користувача, реєстрацію користувача, відправлення запитань тесту, отримання і обробку відповідей, що підтверджує користувач, передачу миттєвих повідомлень при необхідності, організацію зручної статистики, зберігання статистики в базі даних, зберігання бази користувачів, які можуть проходити тести.

Програма призначена для спілкування людей. Користувачам, що використовують дану систему доступні такі можливості:

- реєстрація нового користувача;
- авторизація вже існуючого користувача;
- створення нового Geek-коду;
- збереження створеного коду у базі даних та управління ним;
- розшифрування коду, представленого у текстовому вигляді;
- надсилання повідомлення, яке буде доставлене випадковому користувачеві;
- отримання повідомлення від випадкового користувача;
- запрошення користувача до чату та обмін Geek-кодами;
- надсилання повідомлення до чату;
- отримання повідомлення з чату;
- збереження користувача до списку друзів (за Geek-кодом);
- управління списком друзів;
- фільтрація нецензурної лексики у повідомленнях;

- репорт адміністрації у разі спаму;
- управління повідомленнями;
- розгорнута довідка.

4.2 Умови виконання програми

Мінімальні системні вимоги та програмне забезпечення , що необхідні для функціонування серверної частини мобільного застосунку:

- ОС – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- Web-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- пристрій з ОС Android 4.2 та вище;
- процесор - 300 MHz;
- обсяг оперативної пам'яті – не менше 256 Мб;
- підключення до мережі Internet.

4.3 Інструкція по експлуатації програми

4.3.1 Характеристики програми

Логічну структуру програми визначає її функціональне призначення й можливості обраних програмних і інструментальних засобів проектування.

Логічну структуру системи можна представити у вигляді моделі, представленої на рисунку 4.1.

Web-застосунок – набір файлів програмного продукту, розміщених на Web- сервері, що надає хостинг.

Серверні сценарії – набір файлів, написаних мовою серверних сценаріїв PHP, призначених для створення сторінок, до яких може здійснюватись доступ за допомогою API.

Користувачська частина – Android-програма. Складається з програми та бази даних SQLite – разом становлять єдиний клієнтський програмний продукт.

База даних – серверна (MySQL) та клієнтська (SQLite). Серверна БД містить такі таблиці: Пристрій, Користувач, Повідомлення. Клієнтська БД містить такі таблиці: повідомлення.



Рисунок 4.1 – Структура системи AnonSend

4.3.2 Перелік файлів, у яких знаходиться програма

Веб-застосунок складається в загальному обсязі з 26 файлів, які умовно можна розділити на категорії:

- файли з розширення `php`, що необхідні для організації функціональної частини;
- файли з розширенням `html` та `css` для організації GUI частини;
- файл бази даних;
- графічні файли для створення зручного і приємного інтерфейсу.

4.3.3 Звернення до програми

Для розміщення Web-додатку необхідний Web-вузол із установленим на ньому програмним забезпеченням.

Використовуючи інструкції хостинг-провайдера, скопіювати розроблені PHP файли у зазначену директорію, не міняючи їхнього взаємного розташування.

На наданому для хостингу Web-вузла на MySQL-сервері необхідно створити базу даних.

Необхідно отримати логін і пароль користувача БД з правами адміністратора, дисковий простір для БД.

Використовуючи `phpMyAdmin` «Експорт», вибравши шлях до файлу дампу БД і вказавши кодування, перенести БД на новий сервер, або скопіювати папку з базою даних у директорію розміщення даних баз даних MySQL-сервера.

Для початку роботи необхідно встановити клієнтську програму на пристрій з `apk`-файлу та запустити її.

4.3.4 Вхідні й вихідні дані

Вхідними даними є інформація про: користувача (e-mail, пароль та результат анкетування, що покриває різні аспекти життя користувача: громадського, суспільного, професійного, домашнього та ін.), повідомлення (відправник та отримувач, текст). SHA1-код пароля, Geek-код (результат анкетування) та повідомлення зберігаються у текстовій формі в таблицях бази даних на сервері.

Вихідною інформацією є дані про користувача у вигляді Geek-коду, які клієнтська частина розшифровує в зручний для користувача вигляд та повідомлення користувачів.

4.4 Методика та результати тестування

Тестування мобільного застосунку виконаємо в декілька етапів. На початку протестуємо функціональність застосунку, а на другому етапі виконаємо тестування за допомогою спеціального ПЗ – N-Stalker.

4.4.1 Тестування функціоналу

Тестування функціональності розпочнемо із реєстрації нового користувача. На початку користувач переходить на вікно для авторизації або реєстрації, як показано на рисунку 4.2.

Після заповнення основних полей, користувач може в будь-який момент повернутися до їх редагування або встановлення фото профілю, як показано на рисунку 4.3.

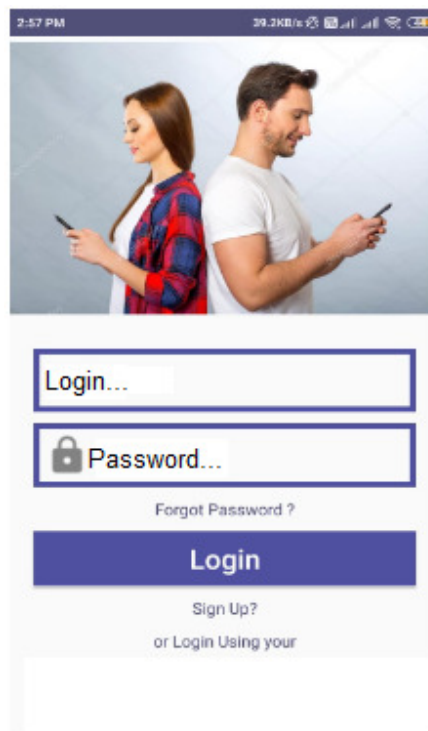


Рисунок 4.2 – Вікно авторизації

Після додавання декількох спів розмовників до друзів користувач в будь-який момент може переглянути їх список, як зображено на рисунку 4.4.



Рисунок 4.3 – Редагування інформації про користувача

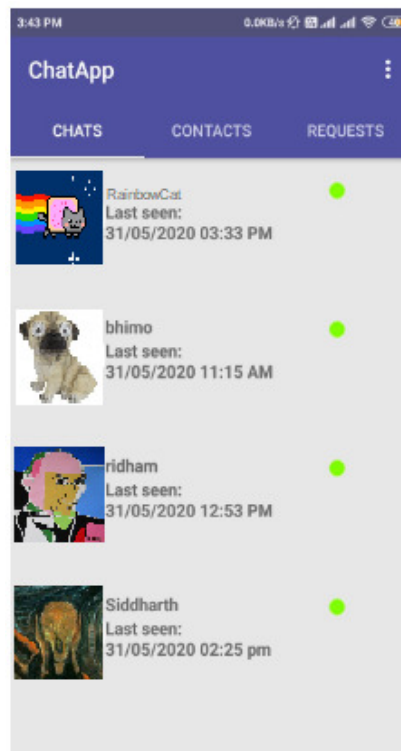


Рисунок 4.4 – Список друзів користувача

Приклад вікна із веденням діалогу між користувачами наведено на рисунку 4.5.

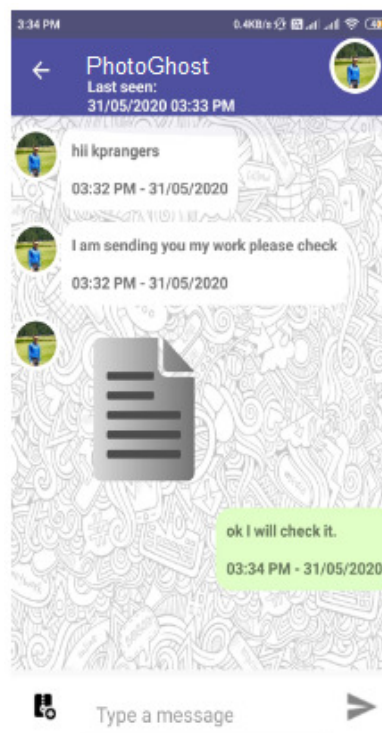


Рисунок 4.5 – Приклад ведення діалогу між користувачами

4.4.2 Тестування за допомогою N-Stalker

На рисунках 4.6 – 4.8 зображені результати тестування застосунку за допомогою інструменту N-Stalker [13].

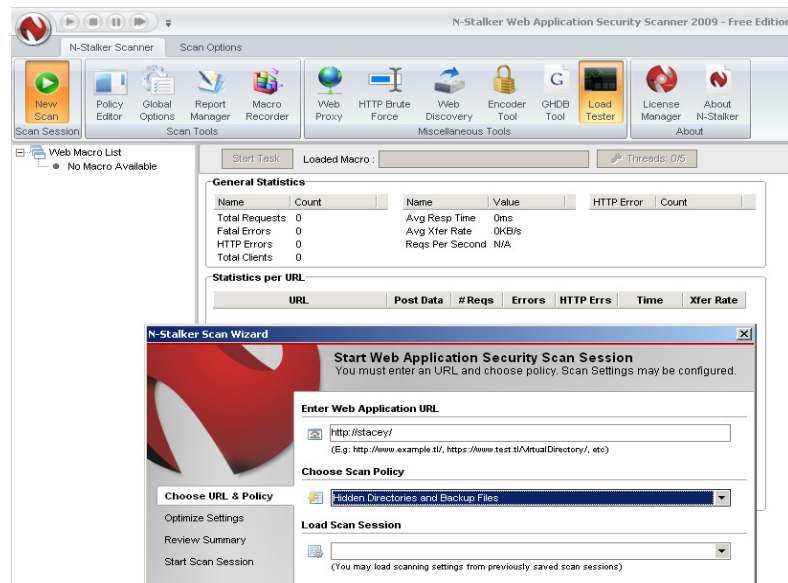


Рисунок 4.6 – Ввід веб-серверу та вибір тесту для тестування інструментом N-Stalker

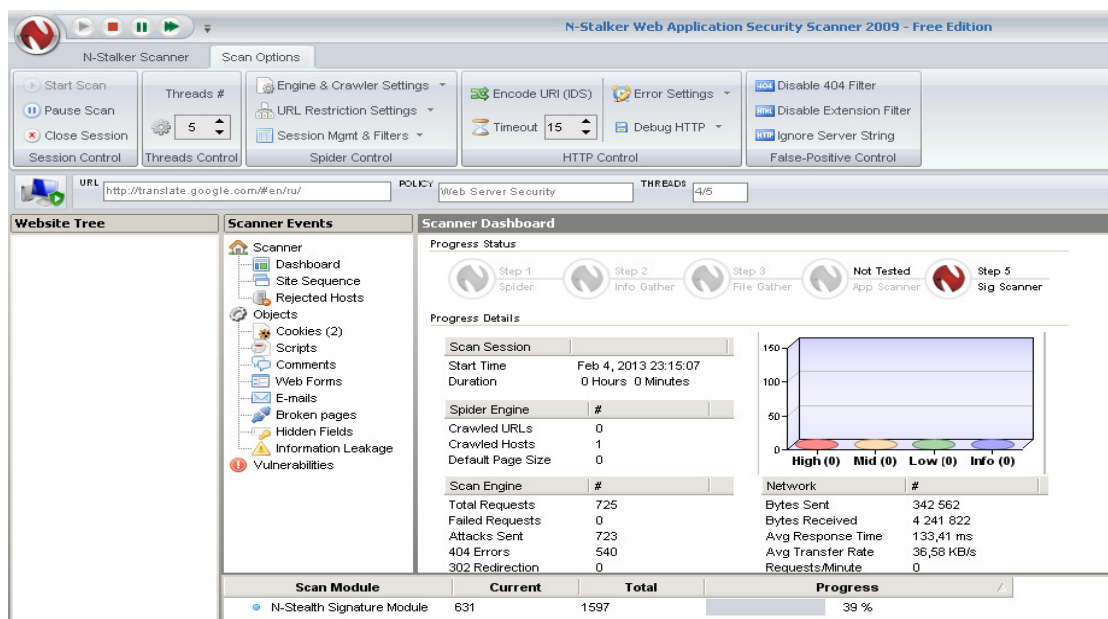


Рисунок 4.7 – Процес сканування модулів

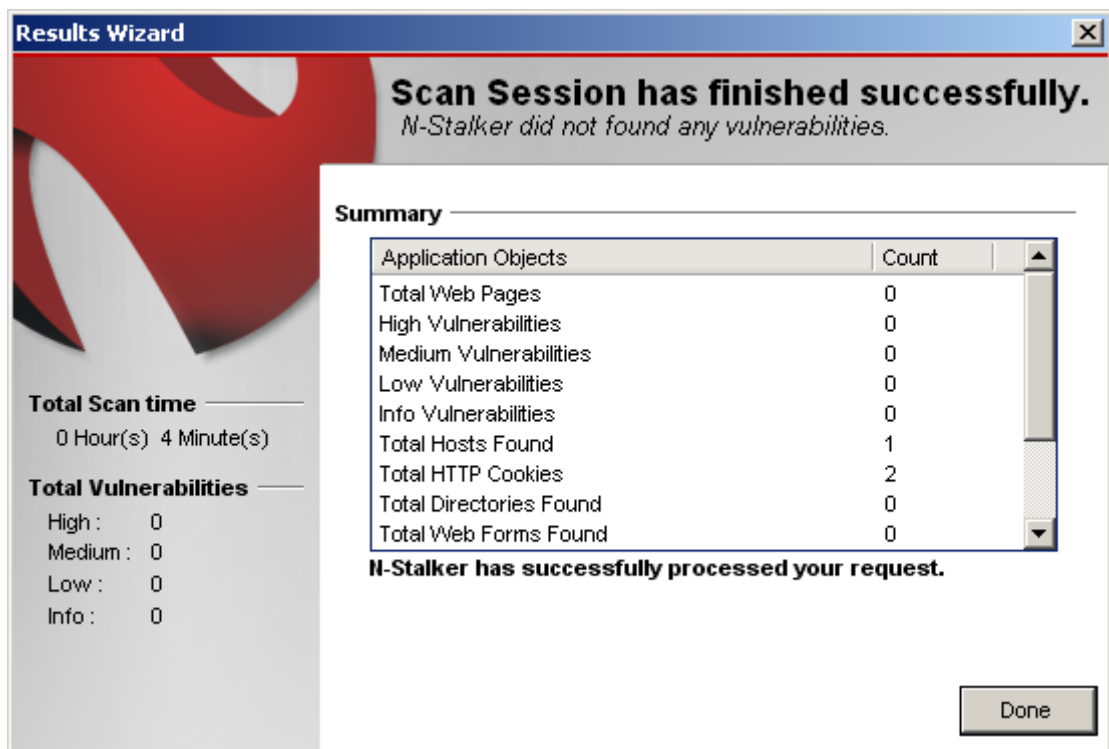


Рисунок 4.8 – Результати тестування

В результаті виконання завдання було отримано систему, що відповідає всім вимогам представленим в технічному завданні, зручний для використання та виконує всі необхідні функції, для спілкування людей, що мають спільні інтереси.

ВИСНОВКИ

У дипломній кваліфікаційній роботі бакалавра виконано аналіз проблеми віртуального спілкування засобами анонімного чатування. Для цього було виконано наступні завдання:

- розглянуто проблемну область;
- проаналізовано сучасні найбільш популярні аналоги для анонімного чатування;
- виконано проєктування мобільного застосунку;
- розроблено мобільний застосунок;
- розроблено сценарій та проведено тестування.

Для програмної реалізації мобільного застосунку для анонімного чатування було проаналізовано сучасні засоби для проєктування та розробки мобільних застосунків для смартфонів під управлінням ОС Android на основі виявлених особливостей та архітектури застосунку, що розробляється в рамках даної дипломної кваліфікаційної роботи. Було виявлені необхідні засоби. Отже, для розробки практичної частини була розроблена за допомогою: мова програмування Java, середовище розробки Android Studio; СКБД MySQL для проєктування та розробки бази даних з метою централізованого збереження інформації; веб-сервер XAMPP.

В процесі виконання роботи було пройдено ряд етапів, що дозволили швидко та якісно розробити необхідну систему:

- збір та аналіз вимог;
- розробка технічного завдання;
- проєктування системи з використанням UML діаграм;
- проєктування інтерфейсу мобільного застосунку;
- розгортання веб-серверу;
- аналіз якості та оцінка мобільного застосунку.

В результаті розробки дипломної кваліфікаційної роботи, було розроблено мобільний застосунок з наступними можливостями:

- створення індивідуального профілю;
- авторизація користувачів;
- обмін повідомленнями з випадковими користувачами;
- запрошення користувача до чату та обмін Geek-кодами (після обміну трьома повідомленнями);
- розшифрування Geek-коду співбесідника після переходу до чату;
- обмін повідомленнями у чаті;
- збереження користувача до списку друзів (за його Geek-кодом);
- управління списком друзів;
- фільтрація нецензурної лексики у повідомленнях;
- репорт адміністрації у разі спаму;
- розгорнута довідка.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам представленим в технічному завданні, зручна для використання та виконує всі необхідні функції, для спілкування людей, що мають спільні інтереси.

Усі завдання випускної дипломної роботи бакалавра повністю виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Малімон В. І. Віртуальне спілкування і самотність людини [Текст] / В. І. Малімон // Гуманітарний вісник Запорізької державної інженерної академії. – 2015. – Вип. 63. – С. 108-115.
2. Коханова О. П. Віртуальне спілкування як вид ділової комунікації [Текст] / О. П. Коханова // Науковий вісник Херсонського державного університету. Серія : Психологічні науки. – 2017. – Вип. 2(3). – С. 54-59.
3. Omegle: Talk to strangers! [Electronic resource]. – Access mode: <https://www.omegle.com/>
4. Yik Yak [Electronic resource]. – Access mode: https://play.google.com/store/apps/details?id=xdk.intel.blank.ad.version55&hl=en_US&gl=US
5. Whisper [Electronic resource]. – Access mode: https://play.google.com/store/apps/details?id=sh.whisper&hl=en_US&gl=US
6. Сав'юк Л. О. Розробка багатопотокових мобільних додатків інформаційно-навчального призначення [Текст] / Л. О. Сав'юк, Р. Б. Іванів // Вісник Національного технічного університету "ХПІ". Серія : Механіко-технологічні системи та комплекси. – 2016. – № 7. – С. 32-42.
7. Храбатин О. В. Вимоги до сучасного програмного забезпечення систем моніторингу навколишнього середовища об'єктів нафтогазової галузі [Текст] / О. В. Храбатин, Р. І. Храбатин, Г. В. Григорчук, Л. В. Саманів // Науковий вісник Івано-Франківського національного технічного університету нафти і газу. – 2011. – № 3. – С. 100-103.
8. Parfyonov Y. E. Using new features of Java SE 8 [Текст] / Y. E. Parfyonov, V. M. Fedorchenko // Системи обробки інформації. – 2015. – Вип. 4. – С. 45-48.
9. Ткаченко О.А. Розробка мобільних додатків під Android [Текст] : навч. посіб. / Ткаченко О. А., Ткаченко К. О., Чайковська О. А. ; Київ. нац.

ун-т культури і мистецтв. – Київ : КНУКіМ, 2017. – 274 с. : рис., табл. -
Бібліогр.: с. 272-274.

10. Android Studio [Electronic resource]. – Access mode:
<https://developer.android.com/studio>

11. Білоус В. Базові принципи проєктування мультимедійних освітніх
мобільних додатків [Текст] / В. Білоус // Освітологічний дискурс. – 2019. –
№ 3-4. – С. 175-184.

12. Скачков Д. А. Розробка практичних методів проєктування та
створення веб-додатків [Текст] / Д. А. Скачков // ScienceRise. – 2015. –
№ 9(2). – С. 22-26.

13. N-Stalker [Electronic resource]. – Access mode:
<https://www.nstalker.com/alliance/>

ДОДАТОК А
Технічне завдання

Вступ

Метою роботи є розробка системи «AnonSend». Система повинна забезпечувати можливість швидкого та анонімного обміну повідомленнями з випадковими користувачами та з можливістю переходу до чату.

A.1 Підстави для розробки

Розробка виконується на основі завдання на дипломну кваліфікаційну роботу бакалавра, яке було визначене за темою «Програмна реалізація мобільного застосунку для чатування з анонімними користувачами спільнот» і затверджене наказом № 103 від 30 березня 2021 р. за Національним університетом «Запорізька політехніка».

A.2 Призначення розробки

Завданням системи є об'єднання людей, які бажають анонімно спілкуватися чи знайти нових знайомих у мережі.

Користуватися системою матимуть змогу будь-які користувачі, які встановили програму та пройшли безкоштовну реєстрацію.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до нефункціональних характеристик

- система повинна працювати на платформі Android;
- мінімальна підтримувана версія ОС Android – 4.2.

А.3.2 Вимоги до функціональних характеристик

Програма має виконувати наступні функції:

- створення індивідуального профілю, що містить в собі інформацію про користувача (у вигляді Geek-коду);
- авторизація користувачів;
- обмін повідомленнями з випадковими користувачами;
- запрошення користувача до чату та обмін Geek-кодами (після обміну трьома повідомленнями);
- розшифрування Geek-коду співбесідника після переходу до чату;
- обмін повідомленнями у чаті;
- збереження користувача до списку друзів (за його Geek-кодом);
- управління списком друзів;
- фільтрація нецензурної лексики у повідомленнях;
- репорт адміністрації у разі спаму;
- розгорнута довідка.

А.3.3 Вимоги до надійності

Вся інформація про користувачів системи має зберігатися в окремій базі даних на сервері, з функцією періодичного автозберігання та резервного копіювання. Особиста інформація користувача також повинна зберігатись на його пристрої. Список друзів також повинен зберігатися і на пристрої і на сервері. Повідомлення зберігаються у базі даних на пристрої.

А.3.4 Умови експлуатації

Для розміщення серверної частини необхідний Web-вузол із установленим на ньому програмним забезпеченням:

- ОС – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- веб-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для встановлення клієнту пристрій повинен мати наступні характеристики:

- операційна система – Android 4.2 або вище.

A.3.5 Вимоги до складу та параметрів технічних засобів

Для експлуатації програмного комплексу необхідні такі програмно-технічні засоби:

- пристрій з ОС Android 4.2 та вище;
- процесор - 300 MHz;
- обсяг оперативної пам'яті – не менше 256 Мб;
- підключення до мережі Internet.

A.3.6 Вимоги до маркування та упакування

Програма може бути записана на компакт-диск. На упаковці повинна бути назва програми – «AnonSend».

A.3.7 Вимоги до транспортування та збереження

Вимоги до транспортування й зберігання аналогічні вимогам, пропонуваним до носія, на якому записана програма.

A.3.8 Вимоги до програмної документації

Програма повинна поставлятися з «Керівництвом адміністратора» та затвердженим додатком «AnonSend. Опис програми»

A.4 Стадії та етапи розробки

Можна виділити такі стадії та етапи розробки системи:

а) збирання та аналіз вимог. Проводиться збір вимог до програмного забезпечення, їх систематизація, документування, аналіз, виявлення протиріч, неповноти, рішення конфліктів в процесі розробки програмного забезпечення;

б) розробка технічного завдання (ТЗ). На основі раніше зібраних вимог формується вихідний документ (ТЗ) на проєктування програмного забезпечення. Описується всі вимоги до створюваного продукту;

в) проєктування архітектури системи. Дозволяє оцінити інформативність та компоновку елементів системи на початковому етапі розробки;

г) детальне проєктування архітектури;

д) розробка структури;

е) реалізація основних функцій серверної частини;

ж) розробка клієнтської частини. Початок інтеграції з серверною частиною;

з) остаточна інтеграція серверної та клієнтської частини;

и) тестування. Професійні тестувальники знаходять «баги» у системі та фіксують їх;

к) розміщення та відкритий доступ, технічна підтримка. Програма розміщується на Google Play;

л) оформлення документації.

A.5 Порядок контролю та приймання системи

Приймання системи здійснюється відповідно до складу та змісту робіт, наведених у Технічному завданні, і відповідно до умов контракту.

Звіт про проведену роботу направляється Замовнику на паперовому носії в 2-х екземплярах, підписаних належним чином, а також на електронному носії.

Вимоги до оформлення проміжних звітів і підсумкового звіту узгоджуються з Виконавцем при підписанні контракту.

Одночасно з підсумковим звітом про виконання робіт Виконавець надає Замовнику підписаний зі сторони Виконавця Акт здачі-приймання виконаної роботи.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Назва програмного продукту – AnonSend.

Програма була розроблена за допомогою засобів мобільного фреймворку Android.

Б.2 Функціональне призначення

Програмний продукт виконує такі основні функції:

- реєстрація користувача;
- авторизація користувача;
- обмін повідомленнями з випадковим користувачем;
- обмін повідомленнями з другом;
- збереження користувача до списку друзів;
- управління списком друзів.

Б.2.1 Функція реєстрації користувача

Функція реєстрації передбачає реєстрацію користувача за його e-mail'ом та паролем. Після реєстрації проводиться збір інформації про користувача за допомогою анкетування та створюється Geed-код користувача.

Б.2.2 Функція авторизації користувача

За допомогою авторизації користувач входить до системи. При виборі анонімного співрозмовника система враховує лише онлайн-користувачів.

Б.2.3 Функція обміну повідомленнями з випадковим користувачем

Функція обміну повідомлення з випадковим користувачем передбачає можливість доставки повідомлення випадковому онлайн-користувачу. Отримувач може відповісти на повідомлення, а може його проігнорувати.

Б.2.4 Функція обміну повідомленнями з другом

Функція обміну повідомлення з другом передбачає можливість обміну повідомленнями між двома користувачами у чаті. При цьому співбесідник бачить логін користувача та інформацію про нього.

Б.2.5 Функція збереження користувача до списку друзів

Функція передбачає можливість зберегти користувача до списку друзів, що дає можливість у будь-який момент відправити йому повідомлення або переглянути інформацію про нього. Співбесідника можна зберегти до списку друзів лише після його підтвердження. Запит можна відправити лише після того, як співбесідники обмінялися 3+ повідомленнями. Запит може бути ініційований будь-якою зі сторін.

Б.2.6 Функція управління списком друзів

Функція передбачає можливість видаляти користувачів зі списку друзів та перейменовувати друзів. При видаленні користувача зі списку друзів зв'язок з ним втрачається, історія видаляється.

Б.3 Опис логічної структури

Логічну структуру програми визначає її функціональне призначення й можливості обраних програмних і інструментальних засобів проектування.

Логічну структуру системи можна представити у вигляді моделі, представленої на рис. Б.1.



Рисунок Б.1 – Структура системи AnonSend

Web-додаток - набір файлів програмного продукту, розміщених на Web- сервері, що надає хостинг.

Серверні сценарії – набір файлів, написаних мовою серверних сценаріїв PHP, призначених для створення сторінок, до яких може здійснюватись доступ за допомогою API.

Користувачька частина – Android-програма. Складається з програми та бази даних SQLite – разом становлять єдиний клієнтський програмний продукт.

База даних – серверна (MySQL) та клієнтська (SQLite). Серверна БД містить такі таблиці: Пристрій, Користувач, Повідомлення. Клієнтська БД містить такі таблиці: Повідомлення.

Б.4 Використані технічні засоби

Для розміщення серверної частини необхідний Web-вузол із установленим на ньому програмним забезпеченням:

- ОС – будь-яка з різновидів операційної системи Unix (включаючи Mac OS X) або Windows;
- веб-сервер – HTTP-сервер Apache для Unix і Windows, сервер Microsoft Internet Information або сервер Netscape Enterprise;
- інтерпретатор PHP5;
- MySQL-сервер із правами створення та вилучення користувачів БД (або база SQLite з аналогічними правами).

Для експлуатації програмного комплексу необхідні такі програмно-технічні засоби:

- пристрій з ОС Android 4.2 та вище;
- процесор - 300 MHz;
- обсяг оперативної пам'яті – не менше 256 Мб;
- підключення до мережі Internet.

Б.5 Виклик та завантаження

Для розміщення Web-додатку необхідний Web-вузол із установленим на ньому програмним забезпеченням.

Використовуючи інструкції хостинг-провайдера, скопіювати розроблені PHP файли у зазначену директорію, не міняючи їхнього взаємного розташування.

На наданому для хостингу Web-вузла на MySQL-сервері необхідно створити базу даних.

Необхідно отримати логін і пароль користувача БД з правами адміністратора, дисковий простір для БД.

Використовуючи phpMyAdmin «Експорт», вибравши шлях до файлу дампу БД і вказавши кодування, перенести БД на новий сервер, або скопіювати папку з базою даних у директорію розміщення даних баз даних MySQL-сервера.

Для початку роботи необхідно встановити клієнтську програму на пристрій з Google Play та запустити її.

Б.6 Вхідні та вихідні дані

Вхідними даними є інформація про: користувача (e-mail, пароль та результат анкетування, що покриває різні аспекти життя користувача: громадського, суспільного, професійного, домашнього та ін.), повідомлення (відправник та отримувач, текст). E-mail, SHA1-код пароля, Geek-код (результат анкетування) та повідомлення зберігаються у текстовій формі в таблицях бази даних на сервері.

Вихідною інформацією є дані про користувача у вигляді Geek-коду, які клієнтська частина розшифровує в зручний для користувача вигляд та повідомлення користувачів.

ДОДАТОК В
Текст програми

```

package com.example.chatapp;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.iid.FirebaseInstanceId;

public class LoginActivity extends AppCompatActivity {

    private FirebaseAuth firebaseAuth;

    private Button btnLog,btnPhonLog;
    private EditText Ulog,Upass;
    private TextView btnSign,txtPass;
    private ProgressDialog progressDialog;
    private DatabaseReference userRef;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        firebaseAuth=FirebaseAuth.getInstance();
        userRef=
FirebaseDatabase.getInstance().getReference().child("Users");

        progressDialog=new ProgressDialog(this);

        btnLog=findViewById(R.id.login_button);
        btnPhonLog=findViewById(R.id.phone_login_button);
        Ulog=findViewById(R.id.login_login);
        Upass=findViewById(R.id.login_password);

```

```

        btnSign=findViewById(R.id.need_new_account_link);
        txtPass=findViewById(R.id.forget_password_link);

        btnSign.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                sendUserToRegisterActivity();
            }
        });

        btnPhonLog.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent phoneintent =new
Intent(LoginActivity.this,PhoneLoginActivity.class);
                startActivity(phoneintent);
            }
        });

        btnLog.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String login=Ulog.getText().toString();
                String password=Upass.getText().toString();

                if(TextUtils.isEmpty(login))
                {

                    Toast.makeText(LoginActivity.this,"Please enter
login...",Toast.LENGTH_SHORT).show();
                }
                else if(TextUtils.isEmpty(password))
                {

                    Toast.makeText(LoginActivity.this,"Please enter
password...",Toast.LENGTH_SHORT).show();
                }
                else
                {
                    progressDialog.setTitle("Log In");
                    progressDialog.setMessage("please
wait...");

                    progressDialog.setCanceledOnTouchOutside(true);
                    progressDialog.show();
                }
            }
        });

```

```

firebaseAuth.signInWithloginAndPassword(login,password).addOnCom
pleteListener(new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull
Task<AuthResult> task) {
        if(task.isSuccessful())
        {
            String
currentUserid=firebaseAuth.getCurrentUser().getUid();
            String          deviceToken=
FirebaseInstanceId.getInstance().getToken();

userRef.child(currentUserid).child("device_token").setValue(devi
ceToken)

.addOnCompleteListener(new OnCompleteListener<Void>() {
    @Override
    public void
onComplete(@NonNull Task<Void> task) {

        if(task.isSuccessful())

            {

                UtoMA();

                Toast.makeText(LoginActivity.this,"Logged          in
                Successfully...",Toast.LENGTH_SHORT).show();

            }

        }

    });

    }
    else
    {
        String
        errormessage=task.getException().toString();

        Toast.makeText(LoginActivity.this,"Error
        :"+errormessage,Toast.LENGTH_SHORT).show();
        }
        progressDialog.dismiss();
    }
    });
}
});
}
});
}
}

```

```

        private void UtoMA() {
            Intent mainIntent=new
Intent(LoginActivity.this,MainActivity.class);
            mainIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
            startActivity(mainIntent);
            finish();
        }

        private void sendUserToRegisterActivity() {
            Intent newuserIntent=new
Intent(LoginActivity.this,RegisterActivity.class);
            startActivity(newuserIntent);
        }
    }

    public class GroupsFragment extends Fragment {

        private View groupfragmenview;
        private ListView listView;
        private ArrayAdapter<String> arrayAdapter;
        private ArrayList<String> list_of_groups=new
ArrayList<>();

        private DatabaseReference databaseReference;
        public GroupsFragment() {
            // Required empty public constructor
        }

        @Override
        public View onCreateView(LayoutInflater inflater,
ViewGroup container,
                                Bundle savedInstanceState) {
            // Inflate the layout for this fragment

            groupfragmenview=inflater.inflate(R.layout.fragment_groups,
            container, false);

            listView=groupfragmenview.findViewById(R.id.list_view);
            arrayAdapter=new
ArrayAdapter<String>(getContext(),android.R.layout.simple_list_i
tem_1,list_of_groups);
            listView.setAdapter(arrayAdapter);

```

```

        databaseReference=
        FirebaseDatabase.getInstance().getReference().child("Groups");
        RetrieveAndDisplay();

        listView.setOnItemClickListener(new
        AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent,
            View view, int position, long id) {
                String
                currentgrouplogin=parent.getItemAtPosition(position).toString();

                Intent                                groupchatintent=new
                Intent(getContext(),GroupChatActivity.class);

                groupchatintent.putExtra("grouplogin",currentgrouplogin);
                startActivity(groupchatintent);
            }
        });

        return groupfragmenview;
    }

    private void RetrieveAndDisplay() {
        databaseReference.addValueEventListener(new
        ValueEventListener() {
            @Override
            public void onDataChange(@NonNull    dtSnpsht
            dtSnpsht) {
                Set<String> set=new HashSet<>();
                Iterator
                iterator=dtSnpsht.getChildren().iterator();
                while (iterator.hasNext())
                {

                set.add(((dtSnpsht) iterator.next()).getKey());
                }
                list_of_groups.clear();
                list_of_groups.addAll(set);
                arrayAdapter.notifyDataSetChanged();
            }

            @Override
            public void onCancelled(@NonNull DatabaseError
            databaseError) {

            }

        });
    }

```

```

    }

}

public class GroupChatActivity extends AppCompatActivity {

    private Toolbar tlbrM;
    private ImageButton send;
    private EditText Umsg;
    private ScrollView scrollView;
    private TextView displaytextmessage;

    private String
currentGrouplogin,currentlogin,currentUserID,currentDate,current
Time;

    private FirebaseAuth mauth;

    private DatabaseReference
RootRef,GrouploginRef,GroupMessageRefKey;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_group_chat);

currentGrouplogin=getIntent().getExtras().get("grouplogin").toSt
ring();

        mauth=FirebaseAuth.getInstance();
        currentUserID=mauth.getCurrentUser().getUid();
        RootRef=
FirebaseDatabase.getInstance().getReference().child("Users");

GrouploginRef=FirebaseDatabase.getInstance().getReference().chil
d("Groups").child(currentGrouplogin);

        tlbrM=findViewById(R.id.group_chat_bar_layout);
        setSupportActionBar(tlbrM);
        getSupportActionBar().setTitle(currentGrouplogin);

        send=findViewById(R.id.send_message_button);
        Umsg=findViewById(R.id.input_group_messages);
        scrollView=findViewById(R.id.my_scroll_view);

displaytextmessage=findViewById(R.id.group_chat_text_display);

        GetUserInfo();

```

```

        send.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v) {
                SaveMtoDB();
                Umsg.setText("");
                scrollView.fullScroll(ScrollView.FOCUS_DOWN);
            }
        });

        @Override
        protected void onStart() {
            super.onStart();

            GrouploginRef.addChildEventListener(new
            ChildEventListener() {
                @Override
                public void onChildAdded(@NonNull dtSnpsht
                dtSnpsht, @Nullable String s) {
                    if(dtSnpsht.exists())
                    {
                        DisplayMessages(dtSnpsht);
                    }
                }

                @Override
                public void onChildChanged(@NonNull dtSnpsht
                dtSnpsht, @Nullable String s) {
                    if(dtSnpsht.exists())
                    {
                        DisplayMessages(dtSnpsht);
                    }
                }

                @Override
                public void onChildRemoved(@NonNull dtSnpsht
                dtSnpsht) {

                }

                @Override
                public void onChildMoved(@NonNull dtSnpsht
                dtSnpsht, @Nullable String s) {

```

```

        }

        @Override
        public void onCancelled(@NonNull DatabaseError
databaseError) {

        }

    });
}

private void SaveMtoDB() {

    String Inptmsg=Umsg.getText().toString();
    String msgKey=GrouploginRef.push().getKey();
    if(TextUtils.isEmpty(Inptmsg))
    {
        Toast.makeText(this, "Please write a
message...", Toast.LENGTH_SHORT).show();
    }
    else
    {
        Calendar calendar =Calendar.getInstance();
        SimpleDateFormat simpleDateFormat=new
SimpleDateFormat("dd/MM/yyyy");

        currentDate=simpleDateFormat.format(calendar.getTime());

        SimpleDateFormat simpleDateFormat1=new
SimpleDateFormat("hh:mm a");

        currentTime=simpleDateFormat1.format(calendar.getTime());

        HashMap<String, Object> groupmsgKey= new
HashMap<>();
        GrouploginRef.updateChildren(groupmsgKey);

        GroupMessageRefKey=GrouploginRef.child(msgKey);

        HashMap<String, Object> messageInfoMap=new
HashMap<>();

        messageInfoMap.put("login", currentlogin);
        messageInfoMap.put("message", Inptmsg);
        messageInfoMap.put("date", currentDate);
        messageInfoMap.put("time", currentTime);
    }
}

```

```

GroupMessageRefKey.updateChildren(messageInfoMap);

    }
}

private void GetUserInfo() {

RootRef.child(currentUserID).addValueEventListener(new
 ValueEventListener() {
    @Override
    public void onDataChange(@NonNull dtSnpsht
dtSnpsht) {
        if(dtSnpsht.exists())
        {

currentlogin=dtSnpsht.child("login").getValue().toString();

        }
    }

    @Override
    public void onCancelled(@NonNull DatabaseError
databaseError) {

    }

});
}

private void DisplayMessages(dtSnpsht dtSnpsht) {
    Iterator
iterator=dtSnpsht.getChildren().iterator();

    while (iterator.hasNext())
    {
        String
chDat=(String) ((dtSnpsht) iterator.next()).getValue();
        String
chtxtmsg=(String) ((dtSnpsht) iterator.next()).getValue();
        String
chlogin=(String) ((dtSnpsht) iterator.next()).getValue();
        String
chTime=(String) ((dtSnpsht) iterator.next()).getValue();

        displaytextmessage.append(chlogin+"
: \n"+
chtxtmsg+"\n"+chTime+"
"+chDat+"\n\n");
    }
}

```

```

        scrollView.fullScroll(ScrollView.FOCUS_DOWN);
    }
}

public class FindFriendsActivity extends AppCompatActivity
{
    private Toolbar mtoolbar;
    private RecyclerView recyclerView;
    private DatabaseReference Userref;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_find_friends);

        Userref=
        FirebaseDatabase.getInstance().getReference().child("Users");

        recyclerView=findViewById(R.id.find_friends_recyclerlist);
        recyclerView.setLayoutManager(new
        LinearLayoutManager(this));

        mtoolbar=findViewById(R.id.find_friends_toolbar);
        setSupportActionBar(mtoolbar);

        getSupportActionBar().setDisplayHomeAsUpEnabled(true);

        getSupportActionBar().setDisplayShowHomeEnabled(true);
        getSupportActionBar().setTitle("Find friends");

    }

    @Override
    protected void onStart() {
        super.onStart();

        FirebaseRecyclerOptions<LstCont> options=new
        FirebaseRecyclerOptions.Builder<LstCont>()
            .setQuery(Userref,LstCont.class)
            .build();

        FirebaseRecyclerAdapter<LstCont,FindFriendsViewHolder> adapter=

```

```

        new        FirebaseRecyclerAdapter<LstCont,
FindFriendsViewHolder>(options) {
            @Override
            protected        void
onBindViewHolder(@NonNull FindFriendsViewHolder holder, final
int position, @NonNull LstCont model) {

holder.login.setText(model.getLogin());

holder.uStat.setText(model.getStatus());

Picasso.get().load(model.getImage()).placeholder(R.drawable.prof
ile_image).into(holder.profile);

holder.itemView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String
visit_user_id=getRef(position).getKey();
                Intent profileintent=new
Intent(FindFriendsActivity.this,ProfileActivity.class);
profileintent.putExtra("visit_user_id",visit_user_id);

startActivity(profileintent);
            }
        });
    }

    @NonNull
    @Override
    public FindFriendsViewHolder
onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

        View view=
LayoutInflater.from(parent.getContext()).inflate(R.layout.users_
display_layout,parent,false);
        FindFriendsViewHolder
viewHolder=new FindFriendsViewHolder(view);
        return viewHolder;
    }
};
recyclerView.setAdapter(adapter);
adapter.startListening();
}

```

```

        public static class FindFriendsViewHolder extends
RecyclerView.ViewHolder {
            TextView login,uStat;
            CircularImageView profile;
            public FindFriendsViewHolder(@NonNull View
itemView) {
                super(itemView);

login=itemView.findViewById(R.id.users_profile_login);
                uStat=itemView.findViewById(R.id.users_status);

profile=itemView.findViewById(R.id.users_profile_image);
            }
        }
    }

    public class ContactsFragment extends Fragment {

        private View view;
        private RecyclerView myrecyclerview;
        private DatabaseReference contactsRef,userRef;
        private FirebaseAuth mauth;
        private String currentUserId;
        public ContactsFragment() {
            // Required empty public constructor
        }

        @Override
        public View onCreateView(LayoutInflater inflater,
ViewGroup container,
                                Bundle savedInstanceState) {
            // Inflate the layout for this fragment
            view=inflater.inflate(R.layout.fragment_contacts,
container, false);

myrecyclerview=view.findViewById(R.id.contacts_recyclerview_list
);

            myrecyclerview.setLayoutManager(new
LinearLayoutManager(getContext()));

            mauth=FirebaseAuth.getInstance();
            currentUserId=mauth.getCurrentUser().getUid();

            contactsRef=
FirebaseDatabase.getInstance().getReference().child("Contacts").
child(currentUserId);

```

```

userRef=FirebaseDatabase.getInstance().getReference().child("Users");

        return view;
    }

    @Override
    public void onStart() {
        super.onStart();

        FirebaseRecyclerOptions<Contacts> options=new
        FirebaseRecyclerOptions.Builder<Contacts>()
            .setQuery(contactsRef,Contacts.class)
            .build();

```

ДОДАТОК Д
Слайди презентації

Міністерство освіти та науки України
Національний Університет «Запорізька політехніка»

**Програмна реалізація мобільного
застосунку для чатування із анонімними
користувачами спільнот**

Старікова Альона Михайлівна

ст. групи КНТ-3117

Гофман Євген Олександрович

к.т.н., доцент НУ «Запорізька політехніка»

Рисунок Д.1 – Слайд 1

Об'єкт, предмет та мета роботи

Об'єкт дослідження: процес анонімного чатування.

Предмет дослідження: методи організації анонімного чатування.

Мета роботи: розроблення мобільного застосунку, що реалізує можливість користувачів спілкуватися анонімно.

2

Рисунок Д.2 – Слайд 2

Постановка завдання

Розроблення мобільного застосунку для анонімного чатування.

Застосунок повинен виконувати такі функціональні вимоги:

- реєстрація користувача;
- авторизація користувача;
- обмін повідомленнями з випадковим користувачем;
- обмін повідомленнями з другом;
- збереження користувача до списку друзів;
- управління списком друзів.

3

Рисунок Д.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	Omegle	Yik Yak	Whisper	AnonSend
Анонімність на початку користування	Необхідно пройти реєстрацію	Необхідно пройти реєстрацію	Не вимагає реєстрації з персональними даними	Не вимагає реєстрації з персональними даними
Вибір персональної інформації	Так	Так	Так	Так
Урегульованість тематик	Так	Так	Так	Так
Адміністрування чатів	Ні	Ні	Частково	Так
Платний функціонал	Так	Так	Так	Ні

4

Рисунок Д.4 – Слайд 4

Порівняння мов програмування

Критерій	Java	Kotlin	C/C++	C#
Швидкість роботи	+	±	±	-
Плагіни та розширення	++	±	+ (залежить від середовища розробки)	+
Парадигми програмування	±	+	+	±
Спільнота	++	+	±	±

5

Рисунок Д.5 – Слайд 5

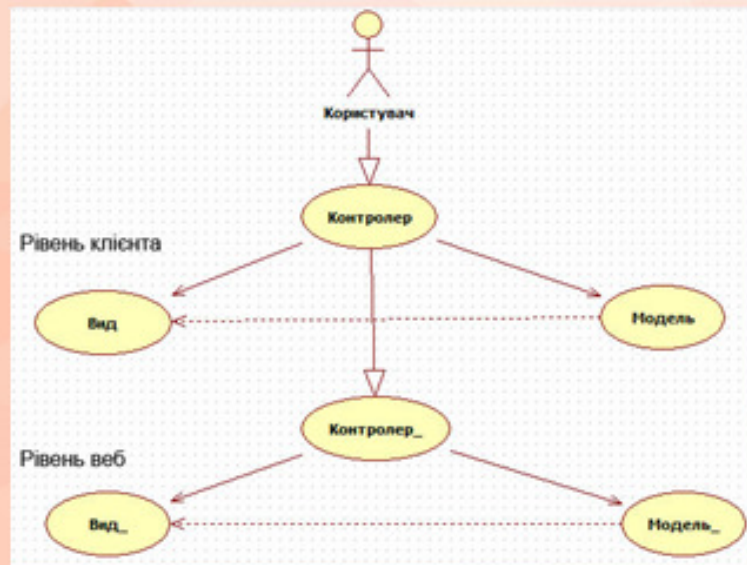
Порівняння середовищ розробки

	Android Studio	Eclipse	Xamarin (Visual Studio)
GUI	++	++	++
Продуктивність	+	++	+
Служба підтримки	+	+	+
Українська мова	-	+	+
Підсвічування синтаксису	+	+	+
Рефакторинг	+	+	+
Автодоповнення коду	+	+	+
Пошук по проекту	+	+	+
Налагодження	+	+	+
Тестування додатків	+	+	+
Мови, які підтримуються	Java Kotlin C++	Java C++	C#

6

Рисунок Д.6 – Слайд 6

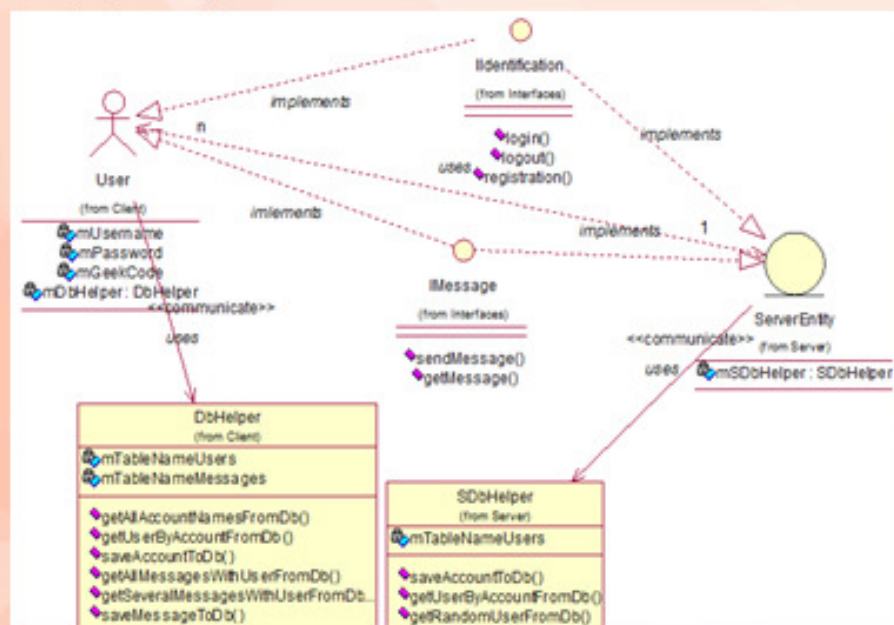
Архітектура системи «AnonSend» на основі предметної області



7

Рисунок Д.7 – Слайд 7

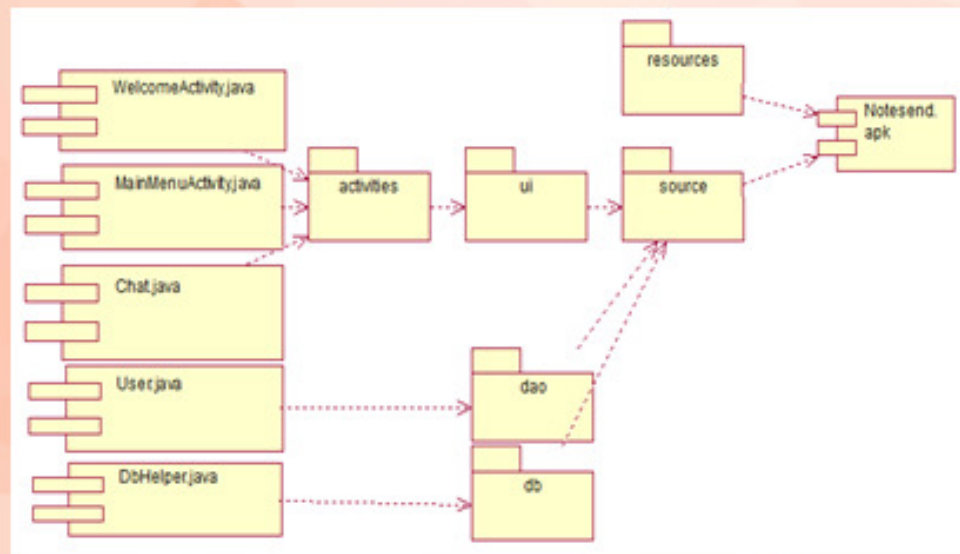
Діаграма класів системи



8

Рисунок Д.8 – Слайд 8

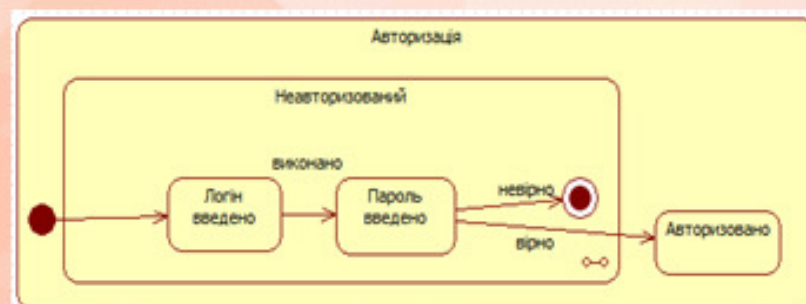
Діаграма компонентів системи



9

Рисунок Д.9 – Слайд 9

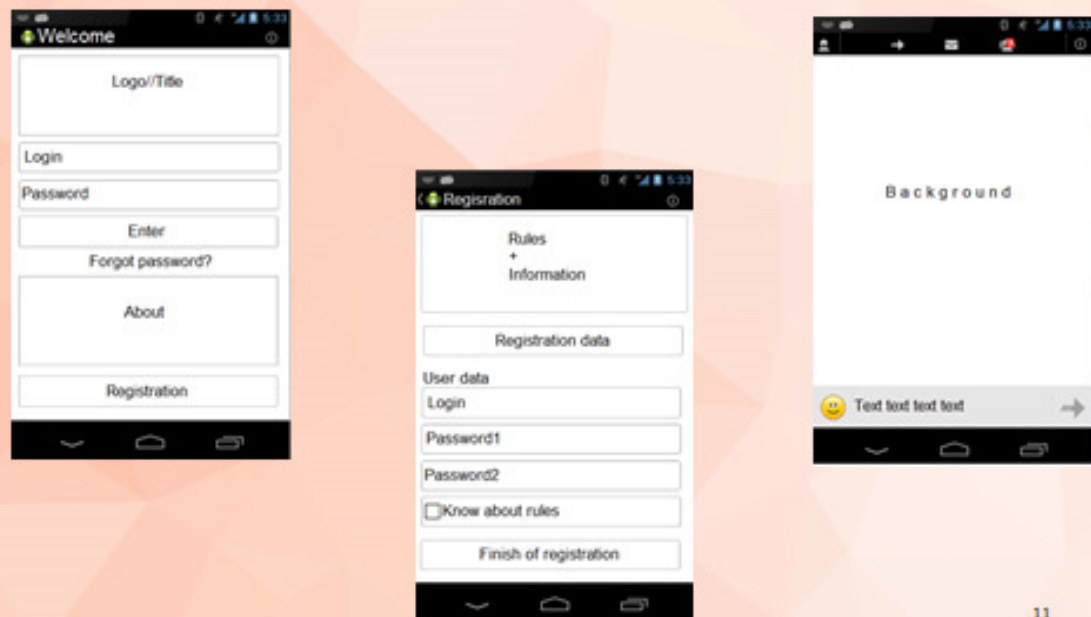
Діаграма станів процесу авторизації



10

Рисунок Д.10 – Слайд 10

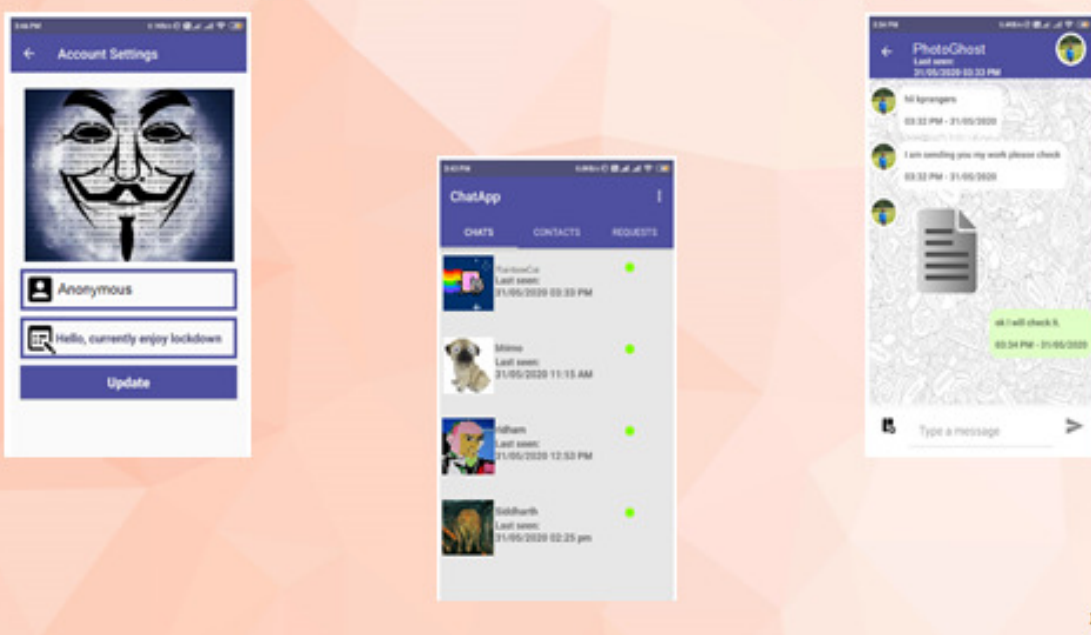
Проектування графічного інтерфейсу користувача



11

Рисунок Д.11 – Слайд 11

Приклади інтерфейсу розробленого застосунку



12

Рисунок Д.12 – Слайд 12

Висновки

У дипломній роботі було виконано наступні завдання:

- розглянуто проблемну область;
- проаналізовано сучасні найбільш популярні аналоги для анонімного чатування;
- виконано проектування мобільного застосунку;
- розроблено мобільний застосунок;
- розроблено сценарій та проведено тестування.

В результаті було розроблено мобільний застосунок з наступними можливостями:

- реєстрація користувача;
- авторизація користувача;
- обмін повідомленнями з випадковим користувачем;
- обмін повідомленнями з другом;
- збереження користувача до списку друзів;
- управління списком друзів.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам представленим в технічному завданні, зручна для використання та виконує всі необхідні функції, для спілкування людей, що мають спільні інтереси.

13

Рисунок Д.13 – Слайд 13

Дякую за увагу

14

Рисунок Д.14 – Слайд 14