

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

ДЛЯ ПІДТРИМКИ ТЕКСТОВОГО СПІЛКУВАННЯ

SOFTWARE TO SUPPORT

TEXT COMMUNICATION

Виконав: студент 4 курсу, групи КНТ-210

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

ПЕРЕТОКІН Д.В.

(ПРІЗВИЩЕ та ініціали)

Керівник КОЛПАКОВА Т.О.

(ПРІЗВИЩЕ та ініціали)

Рецензент СКРУПСЬКИЙ С.Ю.

(ПРІЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

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

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ПЕРЕТОКІНА Данила Владиславовича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для підтримки текстового спілкування. Software to Support Text Communication

керівник проєкту (роботи) к.т.н., доцент, КОЛПАКОВА Тетяна Олексіївна,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

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

2. Строк подання студентом проєкту (роботи) 03 червня 2024 року

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

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

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

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	КОЛПАКОВА Т.О., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання “ 24 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка)

_____ Данило ПЕРЕТОКІН
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

_____ Тетяна КОЛПАКОВА
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
86 с., 30 рис., 3 табл., 3 дод., 19 джерел.

ANDROID STUDIO, JAVA, МОВА ПРОГРАМУВАННЯ, ПРОГРАМНЕ
ЗАБЕЗПЕЧЕННЯ, ТЕКСТОВЕ ПОВІДОМЛЕННЯ, ТЕКСТОВЕ
СПІЛКУВАННЯ.

Об'єкт дослідження – процес розробки програмного забезпечення для комунікації між людьми.

Предмет дослідження – програмні засоби для підтримки процесу текстового спілкування.

Мета роботи – розробка програмного забезпечення для підтримки текстового спілкування.

Матеріали, методи та технічні засоби: мова програмування Java, середовище розробки Android Studio.

Результати. Розроблено програмне забезпечення для підтримки текстового спілкування за допомогою мови програмування Java та середовища розробки Android Studio.

Висновки. Виконано проєктування програмного забезпечення для підтримки процесу текстового спілкування. Розроблено програмне забезпечення для підтримки процесу текстового спілкування. Здійснено тестування розробленого програмного забезпечення для підтримки процесу текстового спілкування.

Галузь використання – програмні засоби для текстового спілкування.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 86 pages, 30 figures, 3 tables, 3 appendixes, 19 sources.

ANDROID STUDIO, JAVA, PROGRAMMING LANGUAGE, SOFTWARE, TEXT MESSAGING, TEXT COMMUNICATION.

The object of research is the process of developing software for communication between people.

The subject of the research is software for supporting the project management process.

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

Materials, methods and technical tools: Java programming language, Android Studio development environment.

Results. The software tools to support the text communication process using the Java programming language and the Android Studio development environment have been developed.

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

Scope of use – software for text communication.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	11
1.1 Поняття про текстове спілкування та програмні засоби для текстового спілкування.....	11
1.2 Аналіз програмного забезпечення для підтримки процесу текстового спілкування.....	25
1.3 Висновки за розділом 1	33
2 Матеріали і методи	35
2.1 Вибір мови програмування.....	35
2.2 Вибір середовища розробки для створення програмного забезпечення для підтримки процесу текстового спілкування	38
2.3 Висновки за розділом 2.....	40
3 Опис програми	41
3.1 Структура програмного забезпечення для підтримки процесу текстового спілкування.....	41
3.2 Функціонування програмного забезпечення для підтримки процесу текстового спілкування.....	43
3.3 Опис алгоритму реєстрації користувачів програмного забезпечення для підтримки процесу текстового спілкування	46
3.4 Проєктування інтерфейсу програмного забезпечення для підтримки процесу текстового спілкування	47
3.5 Висновки за розділом 3.....	49
4 Експлуатація, тестування та експериментальне дослідження програми	51
4.1 Призначення й умови застосування програми	51
4.2 Характеристики програми для підтримки процесу текстового спілкування.....	52
4.3 Інструкція по експлуатації програми	52

4.3.1	Звернення до програми.....	52
4.3.2	Вхідні й вихідні дані.....	52
4.3.3	Повідомлення.....	53
4.4	Виконання програмного забезпечення для підтримки процесу текстового спілкування.....	53
4.5	Тестування програмного забезпечення для підтримки процесу текстового спілкування.....	56
4.6	Висновки за розділом 4.....	57
	Висновки.....	58
	Перелік джерел посилання	61
	Додаток А Технічне завдання.....	63
	Додаток Б Фрагмент тексту програми	67
	Додаток В Слайди презентації	79

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API – Application Programming Interface;

RCS – Rich Communication Services;

MMS – Multimedia Messaging Service;

SMS – Short Message Service;

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Програмне забезпечення для текстового спілкування залишається надзвичайно актуальним і корисним в сучасному світі для особистих і професійних потреб [1]-[7].

Сучасне програмне забезпечення для текстового спілкування дозволяє користувачам легко та, як правило, безкоштовно спілкуватися з іншими людьми, в тому числі з різних частин світу. Такі програмні засоби забезпечують миттєвий доступ до інформації та можливість обміну думками в реальному часі [1].

Програмне забезпечення для підтримки текстового спілкування зазвичай надає багато корисних функцій, таких як можливість надсилання мультимедійних файлів, голосових повідомлень, відео дзвінків тощо, що полегшує спілкування між користувачами [1], [2].

Програмні засоби текстового спілкування дозволяють швидко передавати інформацію без необхідності телефонного дзвінка чи особистої зустрічі [3].

Програмне забезпечення для підтримки процесу текстового спілкування, як правило, є кросплатформним та може працює на різних апаратних та програмних платформах, включаючи комп'ютери, смартфони та планшети. Це надає користувачам гнучкість у виборі пристрою для спілкування [4].

Проте деякі програми для підтримки процесу текстового спілкування можуть мати проблеми з безпекою даних, зокрема, може бути вразливим до кібератак, витоку даних або несанкціонованого доступу до особистої інформації користувачів. Інші програмні засоби текстового спілкування можуть мати або зайві функції, непотрібні деяким користувачам, або, навпаки, мати недостатній функціонал, що залежить від потреб конкретних груп користувачів. Деякі програми для підтримки процесу текстового спілкування

можуть мати обмеження на кількість користувачів, які можуть брати участь у групових чатах або надсилати повідомлення одночасно [3]-[5].

Тому актуальною є розробка програмного забезпечення для підтримки процесу текстового спілкування. У дипломній кваліфікаційній роботі бакалавра розв'язується актуальне завдання розробки програмного забезпечення для підтримки текстового спілкування.

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

- виконати аналіз предметної області та програмних засобів для підтримки процесу текстового спілкування;
- здійснити проєктування програмного забезпечення для підтримки текстового спілкування;
- створити програмне забезпечення для підтримки текстового спілкування;
- виконати тестування розробленого програмного забезпечення для підтримки текстового спілкування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття про текстове спілкування та програмні засоби для текстового спілкування

Текстове спілкування або обмін текстовими повідомленнями полягає у відправленні літерно-цифрових текстів між різними портативними пристроями, такими як мобільні телефони, планшети або ноутбуки через бездротові або інші мережі, що забезпечуються операторами зв'язку [1].

Популярним та широко використовуваним методом обміну текстовими повідомленнями є використання мобільних телефонів та стільникового зв'язку, що використовується для різних цілей, від простих розмов між користувачами до інформаційних послуг, рекламних акцій, електронної комерції та інших цілей [1].

Текстові повідомлення стали альтернативою для багатьох користувачів для спілкування поряд з електронною поштою, миттєвими повідомленнями та соціальними мережами, оскільки вони мають широку підтримку від операторів, прості у використанні, доступні по всьому світу та широко використовуються в різних культурах [1].

Користувачі можуть надсилати текстові повідомлення через різні канали зв'язку або використовуючи спеціальні програмні сервіси. Вони стали особливо популярними, оскільки спочатку це було більш вигідно за виклик голосом, але з появою необмежених голосових планів, які включають і текстові повідомлення, вони залишаються популярними завдяки своїй зручності [1].

Обмін текстовими повідомленнями представляє собою спосіб спілкування в режимі реального часу через програмне забезпечення, яке дозволяє обмінюватися повідомленнями [2].

Багато зовнішніх програм мають додаткові можливості, такі як обмін файлами, текстовими повідомленнями, відео та голосові виклики, на додачу до основної функції обміну повідомленнями [2]. До найпопулярніших програм

для обміну повідомленнями можна віднести: Facebook Messenger, WhatsApp, Snapchat та Skype [2]. Терміни, такі як прямі повідомлення або чат, використовуються для опису аналогічних функцій обміну текстовими повідомленнями у соціальних медіа [2]. Хоча ці терміни більше пов'язані з соціальними мережами, як Instagram або Facebook, вони всі відносяться до обміну повідомленнями [2].

З іншого боку, текстові повідомлення – це короткі електронні повідомлення, які передаються між різними пристроями. Текстові повідомлення можуть використовувати стільникову мережу або інші види мереж, зокрема, мережу Інтернет, для передачі повідомлень, які можуть бути обмежені певною кількістю символів [2].

За допомогою програмних засобів обміну текстовими повідомленнями або чатування, текстові повідомлення можна використовувати для обміну інформацією, подібно до інших програм [2]. З появою розумних телефонів, таких як iPhone та Android, можна відправляти текстові повідомлення з мобільних пристроїв через Інтернет [2].

Однак, навіть при наявності розумних телефонів, існують основні відмінності між текстовими повідомленнями та миттєвим обміном повідомленнями, які роблять їх дуже різними [2]. Для миттєвого обміну повідомленнями як відправник, так і одержувач повинні використовувати одну програму. Користувач, як правило, не може надіслати повідомлення в одній програмі і доставити його комусь в іншій [2]. За допомогою текстових повідомлень можна надсилати та отримувати повідомлення за допомогою будь-якого мобільного телефону, незалежно від оператора мобільного зв'язку [2].

За допомогою миттєвих повідомлень можна надсилати повідомлення людям у будь-якій точці світу, якщо вони мають підключення до Інтернету [2]. У випадку текстових повідомлень, що надсилаються через засоби мобільного телефону, тарифний план визначатиме, чи зможе користувач надсилати повідомлення за кордон [2].

Миттєвий обмін повідомленнями часто можна здійснювати на комп'ютері, планшеті чи ноутбуці. Однак, якщо ви хочете використовувати служби обміну повідомленнями на мобільному телефоні, потрібен розумний телефон [2]. Надсилання текстових повідомлень вбудовано в кожен мобільний пристрій, тому не потрібно спілкуватися через розумний телефон, якщо потрібно надіслати звичайне повідомлення [2].

Спілкування за допомогою текстових повідомлень являє собою процес надсилання та отримання коротких письмових повідомлень за допомогою мобільного телефону або іншого технічного засоби. Їх також можна називати мобільними повідомленнями, повідомленнями поштою або послугою коротких повідомлень [3].

У порівнянні з миттєвими повідомленнями, текстові повідомлення зазвичай були довшими та містили більше речень. Ймовірно, це сталося через фактори вартості та тенденції розбивати розмови у миттєвих повідомленнях на короткі фрагменти. У текстових повідомленнях було багато скорочень, але їх кількість все одно залишалася обмеженою [3].

Відзначимо, що при розробці програмного забезпечення для текстового спілкування важливо забезпечити прогнозне надсилання текстових повідомлень, що являє собою функцію, яка передбачає автоматичне виведення повного слова після введення лише його перших літер [3].

Хоча ця функція зменшує кількість натискань клавіш, вона також може мати свої недоліки. Не всі користувачі використовують прогнозне надсилання текстових повідомлень. Деякі користувачі не використовують цю функцію, оскільки вважають, що це сповільнює їхню роботу, оскільки відповідні програмні модулі не завжди пропонують правильні слова, або ж додають нові слова повільно [3].

Хоча прогнозне надсилання текстових повідомлень може бути корисним для правопису, воно не завжди ефективно працює. Наприклад, введення певних фраз може призвести до виведення неправильних слів. Щодо проникнення умов обміну текстовими повідомленнями в письмову мову,

прогнозне надсилення текстових повідомлень стає все більш поширеним і вдосконалим, і хоча можна передбачити, що воно буде впливати на уявлення про стандарти мови, точний характер цього впливу важко передбачити [3].

Текстові повідомлення можуть вважатися актом надсилення коротких повідомлень за допомогою мобільних телефонів, зазвичай за допомогою служби коротких повідомлень [4].

Спілкування на основі текстових повідомлень було розроблено наприкінці минулого століття. Проте обмін текстовими повідомленнями не отримав поширення, доки не стало можливим надсилати повідомлення між різними мережами стільникового зв'язку [4].

Текстові повідомлення, що надсилають за допомогою спеціальних програмних засобів, використовуються активістами для оголошень про демонстрації на вулицях різних країн або для подання іншої інформації. Основні компанії бездротового зв'язку повідомляють, що користувачі тепер більше пишуть текстові повідомлення, ніж розмовляють по мобільному телефону [4].

Протягом останнього десятиліття спосіб використання текстових повідомлень зазнав значних змін як серед споживачів, так і серед брендів. Від початкового використання тексту для спілкування з родиною та друзями до сучасного моменту, коли споживачі регулярно отримують повідомлення від брендів, обмін повідомленнями став більш цінним [5].

Користувачі є завжди доступними для зв'язку, і цей рівень доступності відкриває великі можливості для компаній будь-якого розміру. Текстові повідомлення мають найвищі показники відкриття серед усіх форм спілкування, таких як електронна пошта та телефон. Для споживача дуже легко відкрити текстове повідомлення, натиснути посилання та перейти на сайт або відео [5]. Текстові повідомлення стали повсякденною формою спілкування, але раніше переважно використовувалися в особистому спілкуванні [5].

На наступному етапі розвитку текстового спілкування різні компанії почали взаємодіяти з клієнтами онлайн через свої сайти та електронну пошту, використовуючи, зокрема, текстові повідомлення. У даний час обмін текстовими повідомленнями став одним із найпоширеніших засобів спілкування, і цей тренд може продовжитися і далі. Сучасні технології та програмні засоби обміну повідомленнями, зміна споживацьких уподобань та підходу різних компаній та організацій до цих технологій визначають нові відносини між споживачем та бізнесом. Надсилання текстових повідомлень тепер використовується не лише для особистого спілкування. Замість цього споживачі починають довіряти компаніям та очікувати отримувати оновлення та інформацію через текст [5].

Споживачі та компанії все більше спираються на свої телефони і текстові повідомлення, від сповіщень про замовлення з вулиці до того, що більшість споживачів планують робити більше половини своїх покупок у цифровому форматі. Завдяки простоті, багато компаній вже впровадили технології та програмні засоби обміну текстовими повідомленнями, щоб спростити використання для своїх клієнтів і підтримувати зв'язок, оскільки особисті контакти залишаються досить обмеженими [5].

З боку виробничих та торгівельних компаній програмні засоби обміну текстовими повідомленнями не можуть використовуватися лише для надсилання масової інформації, яка розповсюджується на всіх платформах. Навіть якщо споживач підключився до текстових повідомлень, це не означає, що він залишиться в них. Компаній не можуть просто додати повідомлення до свого списку каналів, але мають підтримувати бажання користувачів залишатися зацікавленими та отримувати більше інформації. Тому персоналізація текстових повідомлень при спілкуванні організацій з клієнтами грає ключову роль. Споживачі очікують отримати персоналізовану та релевантну для них інформацію, наприклад, про новий продукт, який відповідає їхнім попереднім покупкам [5]. Тому відповідні програмні засоби повинні враховувати цю інформацію [5].

Повідомлення від користувача до клієнта стають більш розмовними. Двосторонні діалоги з використанням спеціалізованого програмного забезпечення стають нормою, особливо щодо підтримки клієнтів, для того щоб споживачі відчували, що у них є цікаві стосунки з брендом [5].

Крім того, для компаній недостатньо просто покладатися на текстове спілкування, їм також потрібно використовувати спеціалізоване програмне забезпечення у вигляді чат-програм та подібних каналів, оскільки вони створюють більш особисті стосунки зі споживачами. Програми для спілкування пропонують унікальні можливості, і компаній потрібно розвивати цей підвищений рівень багатоканального обміну повідомленнями у всіх аспектах свого бізнесу [5].

Беручи до уваги багаторічний досвід використання електронної пошти, споживачі також можуть шукати подібний рівень зручності і взаємодії в текстових повідомленнях, що сприяє створенню більш гармонійного контакту з клієнтами. Як і в електронній пошті, текстові повідомлення можуть містити новини, зображення, відео та посилання, і очікується, що такий рівень інформації та взаємодії буде переданий також у повідомленнях. Цей підхід також аналогічний іншим технологіям, таким як чат-програми, оскільки вони включають більш інтегровані та розширені функції, що підвищують сприйняття бренду компанії чи організації [5].

При розробленні програмного забезпечення для текстового спілкування важливо врахувати, що споживчі можуть очікувати, що обмін повідомленнями буде ще більш динамічним, оскільки відео, голос та високоякісні зображення стануть все більш важливими складовими багатогранного та стандартного досвіду обміну повідомленнями. Замовлення покупців можуть оновлюватися в режимі реального часу через повідомлення, при цьому замовлення можуть перевірятися або змінюватися, не переходячи за посиланням на окремий сайт. У порівнянні зі стандартним текстовим повідомленням, такий інтегрований та персоналізований вміст, створений з використанням спеціалізованого

програмного забезпечення, може забезпечити підвищений рівень залучення та ефективності витрат [5].

Незважаючи на те, що компанії продовжують впроваджувати інновації та нові програмні засоби у способи використання обміну повідомленнями, технічні команди не повинні забувати про наявність базових елементів системи обміну повідомленнями. Споживачі очікують безпечної та надійної доставки повідомлень від усіх компаній і можуть відмовитися від тих, які не відповідають їхнім очікуванням, тому стабільність швидкість доставки повідомлень є суттєвими факторами, які слід враховувати при розробці відповідного програмного забезпечення [5].

Програмні засоби для текстового спілкування шляхом обміну повідомленнями пропонує надзвичайно перспективні можливості, які компанії часто ігнорують. З еволюцією обміну повідомленнями від простого особистого використання до сьогоденного регулярного явища у відносинах між компанією і споживачем, важливо враховувати, що бренди повинні реалізовували широку, надійну та інноваційну стратегію обміну повідомленнями, щоб вони продовжували будувати та розвивати свої відносини як зі своїми поточними, так і з майбутніми клієнтами [5].

Обмін повідомленнями є типом приватної або групової текстової розмови, яка зазвичай відбувається на мобільних пристроях або спеціалізованих програмних платформах. На відміну від чату, де обидва учасники повинні бути в мережі одночасно, обмін повідомленнями є асинхронним, тобто один користувач може надсилати або отримувати повідомлення без необхідності перебувати в мережі. Обмін повідомленнями бере подібну концепцію і переносить її туди, де користувачі вже проводять свій час, зокрема, в програмах або на інших комунікаційних платформах. Повідомлення забезпечують зручність, контекст та особистий характер розмови за допомогою посилянь та інших засобів [6].

Обмін повідомленнями працює краще, коли він швидкий, автентичний і особистий, на відміну від більш формальних сеансів чатування. Споживачі

реагують позитивно, коли вони знають, що за повідомленням стоїть реальна людина, яка надасть відповідь на їх запитання чи запропонує інформацію за допомогою посилань або медіа файлів. Тому, окрім надійних технологій, успішні програми обміну повідомленнями потребують кваліфікованих фахівців, які розуміють бренд та вміють ефективно спілкуватися [6]. Асинхронний обмін повідомленнями допомагає вдосконалити взаємодію з клієнтами, зменшуючи витрати, підвищуючи продуктивність та покращуючи задоволеність клієнтів [6].

У світі спілкування, повідомлення представляють собою інформацію, яку передають від відправника до утримувача. Це сутність повідомлення, ідея, яку одна особа намагається передати іншій особі. Ця інформація може бути передана через текстові повідомлення [7]. У будь-якому випадку спілкування повідомлення є його основою, оскільки несе ту інформацію, яку необхідно передати іншій особі. Оскільки безпосереднє передавання думок з одного розуму в інший є неможливим, використовуються знаки та символи для подання повідомлень. Ці знаки можуть бути словесними (слова, що складаються з літер) або невербальними (жести, міміка тощо) [7].

Тому при розробці програмного забезпечення для підтримки процесу текстового спілкування важливо надати можливість користувачу надсилати повідомлення з можливостями подання невербальних знаків, які можуть бути виражені у спеціальних картинках. Невербальні сигнали, такі як вираз обличчя, мова тіла та тон голосу, подані за допомогою спеціальних символів, можуть значно вплинути на сприйняття повідомлення [7]. Невербальне спілкування грає важливу роль у тому, як інтерпретуються повідомлення [7].

Проаналізуємо різні типи повідомлень. Інформативні повідомлення передають факти та знання (наприклад, вчитель, який пояснює концепцію, або репортер новин). Переконливі повідомлення намагаються переконати одержувача в певній точці зору або дії. Наприклад, продавець, який намагається укласти угоду, або політик, який виголошує передвиборчу промову. Виразні повідомлення зосереджені на обміні емоціями [7].

Проаналізуємо процес передачі повідомлень [7]. На першому етапі відбувається формування та кодування повідомлення. Тут відправник пише повідомлення, після чого воно за допомогою відповідного програмного забезпечення перетворюється на форму, зрозумілу для програмного забезпечення одержувача, тобто повідомлення перетворюється до потрібного формату даних [7].

На наступному етапі за допомогою каналу передачі даних, тобто шляху, яким подорожує повідомлення, відбувається його передача до одержувача. Вибір каналу зв'язку може впливати на те, як швидко одержувач отримає повідомлення, та чи надійно воно буде захищено [7].

На етапі декодування повідомлення перетворюється до вигляду, зрозумілому одержувачу. Після цього можливим є етап зворотного зв'язку. Хороша комунікація передбачає зворотний зв'язок, коли одержувач повідомляє відправнику про отримання повідомлення. Це може бути словесна відповідь або погоджувальний жест за допомогою спеціальних зображень [7].

Обмін короткими текстовими повідомленнями є швидким, зручним та загальнопоширеним засобом комунікації, який в наш час використовують переважна більшість людей [8]. Важливо пам'ятати, що ефективне використання миттєвих повідомлень залежить та впливає на рівень безпеки користувача [8]. При використанні програмних засобів обміну текстовими повідомленнями важливо правильно створити свій псевдонім, слід підібрати його обережно. Варто зробити цей вибір творчим та унікальним. Після створення псевдоніма важливо оберігати його від розголошення. Тримайте свій псевдонім в секреті і не розголошуйте його комусь. Крім того, важливо уникати публікації псевдоніма на загальнодоступних вебсайтах, де його можуть використовувати для спаму чи іншої шкідливої діяльності. Не варто пов'язувати свій псевдонім з особистою електронною адресою [8]. Тому при розробці програмного забезпечення для підтримки процесу текстового спілкування важливо надати користувачам можливість створення власних псевдонімів та спілкування [8].

При текстовому спілкуванні за допомогою програмних засобів та миттєвих повідомлень важливо не передавати особисту або конфіденційну інформацію, оскільки ці повідомлення можуть бути зламані зловмисниками, так само як і електронна пошта. Всі дані проходять через центральний сервер, де їх можна вкрасти у різних точках мережі. Більшість програм обміну миттєвими повідомленнями зберігають журнали сесій обміну миттєвими повідомленнями для майбутнього використання. Це може бути зручно, особливо якщо ви використовуєте миттєві повідомлення на роботі, але також становить загрозу безпеці. Якщо хтось отримає доступ до журналу обміну миттєвими повідомленнями, він може використати старі повідомлення для отримання особистої або конфіденційної інформації [8]. Тому для деяких компаній та організацій доцільно розробити власне програмне забезпечення для обміну текстовими повідомленнями [8].

Більшість програм обміну миттєвими повідомленнями мають функцію статусу, яка показує друзям або контактам, доступність користувача для спілкування. Якщо миттєві повідомлення використовуються для роботи, варто бути обережним і не передавати конфіденційну інформацію про компанію через незашифровані мережі. Зазвичай стандартні програми обміну миттєвими повідомленнями передають повідомлення через незашифровані мережі. Тому багато компаній розробляють та встановлюють спеціальні клієнти миттєвих повідомлень у своїх внутрішніх зашифрованих мережах, щоб захистити конфіденційну інформацію від зловмисників. Якщо програмне забезпечення зберігає журнали миттєвих повідомлень на центральному сервері, варто переконатися, що доступ до сервера забезпечено надійно [8].

Мобільний обмін повідомленнями радикально змінив спосіб, яким люди та підприємства спілкуються один з одним. Розглянемо різні типи текстових повідомлень [9].

Першим типом є повідомлення, що відправляється за допомогою служби коротких повідомлень (Short Message Service, SMS). SMS є одним із найпоширеніших видів мобільного обміну повідомленнями. Це стандартне

текстове повідомлення, яке надсилається через стільниковий сигнал. SMS обмежується 160 символами та є лише текстом. Цей тип повідомлення, як правило, є платним [9].

Іншим типом є повідомлення, що відправляється за допомогою служби обміну мультимедійними повідомленнями (Multimedia Messaging Service, MMS). MMS також передається через стільниковий сигнал, але відрізняється від SMS. Це текстове повідомлення, яке може супроводжуватися медіа файлом, таким як фотографія або відео. MMS дозволяє надсилати повідомлення довші за 160 символів і може бути привабливішим, ніж SMS. Однак для надсилання та отримання MMS потрібно більше даних, а вартість відправлення трохи вища, ніж для SMS [9].

Обмін миттєвими повідомленнями через Інтернет, на відміну від SMS та MMS, використовує доступ до мережі для надсилання та отримання повідомлень. Обмін миттєвими повідомленнями не обмежується кількістю символів, але потребує наявності однієї і тієї ж програми у обох користувачів [9]. Оскільки для нього потрібне підключення до Інтернету, звичайні мобільні телефони без такого доступу не підтримують обмін миттєвими повідомленнями. Але їх можна надсилати та отримувати зі стаціонарного комп'ютера, ноутбука, планшета або розумного телефону [9].

Обмін миттєвими повідомленнями дозволяє користувачам не лише надсилати та отримувати текстові повідомлення, а й обмінюватися відео, фотографіями, посиланнями, файлами різних типів. Деякі програми обміну миттєвими повідомленнями також підтримують телефонні дзвінки через Інтернет [9]. Програми обміну миттєвими повідомленнями є популярними серед користувачів, оскільки вони дозволяють спілкуватися з усього світу, не сплачуючи великі суми за мобільний зв'язок [9].

Push-повідомлення – це спливаючі повідомлення від програм та компаній, які можуть містити важливі оголошення. Важливо зауважити, що push-повідомлення не з'являються в додатку, а надсилаються на головний екран або екран блокування мобільного телефону [9]. Push-повідомлення - це

інструмент, який використовується багатьма компаніями для заохочення користувачів до взаємодії з їхнім додатком, особливо якщо вони деякий час не користувалися ним [9].

Повідомлення в програмі користувачі отримують лише під час активного використання мобільної програми, як підказки або сповіщення. Ці повідомлення допомагають утримувати зацікавленість користувачів, заохочуючи їх відчувати себе нагородженими або провокуючи до певних дій. Наприклад, коли користувачі входять у програму, компанії можуть відправляти вітальні повідомлення, щоб сповістити їх про нові угоди або події, пов'язані з брендом. Також можуть включатися повідомлення, щоб спонукати клієнтів підписатися на програму, оновити свій профіль або скористатися ексклюзивними пропозиціями [9].

Розширені комунікаційні послуги (Rich Communication Services, RCS) є технологією обміну повідомленнями, програмною платформою, розробленою Google, яка вважається альтернативою традиційним текстовим повідомленням. Основна ідея полягає в тому, що завдяки RCS користувачі можуть отримати більш розширений досвід обміну повідомленнями, схожий на додаток, безпосередньо через свою систему обміну повідомленнями у телефоні. Вони можуть надсилати більш змістовні та привабливі повідомлення, включаючи зображення вищої якості та отримувати сповіщення про прочитання [9].

Будь-який тип обміну текстовими повідомленнями можна використовувати для ефективного спілкування [9].

Обмін повідомленнями є важливим елементом спілкування, де взаємодія між агентами та клієнтами забезпечує найважливішу функціональність програмного забезпечення [10]. При розробці програмного забезпечення для текстового спілкування існує можливість отримати доступ до інтерфейсу прикладного програмування (Application Programming Interface, API) для обміну повідомленнями з метою створення інтеграцій власного програмного забезпечення з програмними модулями, які взаємодіють із функціоналом чатування [10].

При використанні API для розробки програмного забезпечення для текстового спілкування, як правило, застосовуються спеціальні протоколи обміну повідомленнями, які пояснюють такі поняття, як чати, ланцюжки, події, маршрутизація і т. д., що є важливими для розуміння принципу роботи обміну повідомленнями [10].

Різні дії користувача програмного забезпечення для текстового спілкування породжують не лише повідомлення або розширені повідомлення, а також і спеціальні події для різних дій, таких як надсилання файлу, заповнення форми, системні та спеціальні повідомлення [10]. Тому при використанні API для розробки програмного забезпечення для текстового спілкування використовуються декілька загальних правил, які узагальнюють правила створення та використання каналів для текстового спілкування [10].

При створенні нового каналу для текстового спілкування створюється новий активний ланцюжок. При повторному зверненні клієнта та початку нового каналу створюються нові ланцюжки в межах одного каналу. У кожному каналі може бути лише один активний ланцюжок, який зазвичай є останнім потоком. Події завжди додаються до активного потоку [10]. Між спілкуваннями в каналі можуть виникати перерви, але чат залишається відкритим після початку [10]. У режимі обміну повідомленнями повідомлення можна надсилати та отримувати, навіть якщо одержувач (клієнт або агент) перебуває в автономному режимі [10]. Оскільки кожен чат має кілька різних потоків, кілька агентів можуть брати участь в одному каналі. Кожен користувач може мати кілька одночасних каналів зв'язку. Це стосується як агентів, так і клієнтів [10].

Алгоритм, що визначає розподіл каналів зв'язку між агентами, називається маршрутизацією. Після активації режиму обміну повідомленнями чати стають безперервними, що означає, що вони залишаються відкритими після початку. Це забезпечує безперервне спілкування, дозволяючи надсилати та отримувати повідомлення, навіть якщо агент або клієнт не в мережі [10].

Максимальна кількість одночасних каналів спілкування, в яких може брати участь агент, визначається правилами API. Це можна налаштувати в програмі агента адміністратором або власником ліцензії. Один клієнт може розпочати лише один чат, якщо він не спілкується з агентами, які використовують різні ліцензії. Якщо ввімкнено режим обміну повідомленнями, чати тривають безперервно, і клієнти завжди можуть переглянути свою історію спілкування. Клієнти також можуть мати кілька одночасних каналів спілкування для сортування за темами. Якщо режим обміну повідомленнями не використовується, власник або адміністратор ліцензії може вимкнути історію каналів в налаштуваннях додатка агента [10].

Визначено, що текстове спілкування або обмін текстовими повідомленнями полягає у відправленні літерно-цифрових текстів між різними портативними пристроями, такими як мобільні телефони, планшети або ноутбуки через бездротові або інші мережі, що забезпечуються операторами зв'язку. Проаналізовано різні типи повідомлень. Відзначено, що при розробці програмного забезпечення для текстового спілкування важливо забезпечити прогнозне надсилання текстових повідомлень, що являє собою функцію, яка передбачає автоматичне виведення повного слова після введення лише його перших літер. Проаналізуємо процес передачі повідомлень. Відзначено, що при розробленні програмного забезпечення для текстового спілкування важливо врахувати, що споживчі можуть очікувати, що обмін повідомленнями буде ще більш динамічним, оскільки відео, голос та високоякісні зображення стануть все більш важливими складовими багатогранного та стандартного досвіду обміну повідомленнями. Відзначено, що при розробці програмного забезпечення для підтримки процесу текстового спілкування важливо надати можливість користувачу надсилати повідомлення з можливостями подання невербальних знаків, які можуть бути виражені у спеціальних картинках. При розробці програмного забезпечення для текстового спілкування існує можливість отримати доступ до API для обміну повідомленнями з метою

створення інтеграцій власного програмного забезпечення з програмними модулями, які взаємодіють із функціоналом чатування.

1.2 Аналіз програмного забезпечення для підтримки процесу текстового спілкування

Проаналізуємо програмне забезпечення для підтримки процесу текстового спілкування [11]-[15].

Програмне забезпечення Brosix (рис. 1.1) є програмою миттєвого обміну повідомленнями, яка дозволяє не тільки обмінюватися текстовими повідомленнями, а ще й мультимедійними файлами. Використовуючи Brosix як програму для бізнес-комунікацій, керівництво компанії може мати повний контроль над корпоративною мережею та безпечний канал зв'язку [11]-[13].

Brosix makes it easy to keep track of your organization's communication and user activity. Whether it's to more easily recall previous conversations between colleagues, due to a legal requirement of your national law, to better support your customers, or any other reason, **User Activity Log** is an efficient way to never lose track of previous communication.

With this feature you can easily keep track of chat history, login and logout information, and which features your users have used and with whom.

Account	Name	Action
Andre	andre	remove
Daniel	daniel	remove

Get the report

Securely Stored User Activity on Brosix Servers

User Activity is stored on Brosix servers for 3 months initially using "Write Once Read Many" technology, after which it's archived in archive files on the server. This technology makes sure that chat history and other user activity such as login information can't be changed or otherwise tampered with, guaranteeing its integrity.

Рисунок 1.1 – Програмне забезпечення Brosix [13]

Програмне забезпечення Brosix є корпоративним рішенням для обміну миттєвими повідомленнями, яке має повний набір функцій для співпраці, які спрощують бізнес-комунікацію: обмін текстовими повідомленнями, відео та голосовий чат, дошка, передача файлів, спільний доступ до екрану, захист від шкідливих програм і вірусів, а також моніторинг у реальному часі. Канали зв'язку повністю зашифровані для забезпечення додаткової безпеки. Програмне забезпечення Brosix використовує P2P передачу файлів для захисту даних користувачів від злому [11]-[13].

Завдяки необмеженому обміну файлами за допомогою програмного забезпечення Brosix можна забезпечити підвищення ефективності співпраці, збільшення продуктивності та легке керування проектами [11]-[13]. Ця програма миттєвого обміну повідомленнями надає роботодавцям або керівникам команд повний контроль над доступними функціями, привілеями та списками контактів користувачів [11]-[13].

Програмне забезпечення Brosix є корисним інструментом миттєвого обміну повідомленнями для тих, хто шукає надійне, безпечне, просте, економічно ефективне та надійне рішення для бізнес-комунікацій [11]-[13].

Програмне забезпечення Brosix можна легко інтегрувати з різними програмними продуктами та сервісами, зокрема, з продуктами від Google, Shopify, GitHub, Dropbox, Facebook та ін. [11]-[13]. Крім того, завдяки детальному керуванню платформою, можна налаштувати спільні простори кімнат текстового спілкування, списків контактів та забезпечення [11]-[13].

Основними характеристиками програмного забезпечення Brosix для обміну текстовими повідомленнями є такі [11]-[13]:

- необмежена передача файлів: програмне забезпечення Brosix дозволяє передавати файли будь-якого розміру, що дозволяє ефективно обмінюватися інформацією навіть за великих обсягів даних [11]-[13];

- голосові та відео дзвінки: програмне забезпечення Brosix підтримує голосові та відео дзвінки, що полегшує зв'язок між співробітниками на віддалених робочих місцях [11]-[13];

- спільне використання екрана: програмне забезпечення Brosix дозволяє користувачам спільно переглядати свої екрани, що дозволяє проводити віртуальні зустрічі та розв'язувати проблеми в режимі реального часу [11]-[13];

- функціонал дошки: програмне забезпечення Brosix надає можливість використовувати вбудовану дошку для спільної роботи над проектами, збору ідей та створення заміток [11]-[13];

- керування доступом: програмне забезпечення Brosix дозволяє адміністраторам налаштовувати права доступу для користувачів, а також забезпечує безпечну передачу даних за допомогою шифрування [11]-[13];

- інтеграція з іншими сервісами: програмне забезпечення Brosix може інтегруватися з різними сервісами та програмами, що робить процес співпраці більш зручним та продуктивним [11]-[13].

Серед недоліків програмного забезпечення Brosix для обміну текстовими повідомленнями можна відзначити обмежені можливості інтеграції, оскільки в порівнянні з деякими іншими платформами для спілкування, Brosix може мати менші можливості щодо інтеграції з іншими сервісами та програмами. Іншим недоліком є обмежена масштабованість, оскільки для деяких великих компаній програмне забезпечення Brosix може бути менш ефективним рішенням через обмежену масштабованість функцій та можливостей [11]-[13].

Програмне забезпечення Flock (рис. 1.2) є одним з широко використовуваних інструментів для миттєвого обміну повідомленнями, що рекомендується компаніям, які бажають полегшити спільну роботу команди. За допомогою багатьох функцій спільної роботи програмне забезпечення Flock може стати невід'ємною частиною бізнес-мережі різних компаній. Користувачі можуть бути в курсі своїх проектів і створювати нотатки, якими можна ділитися в чатах [14].

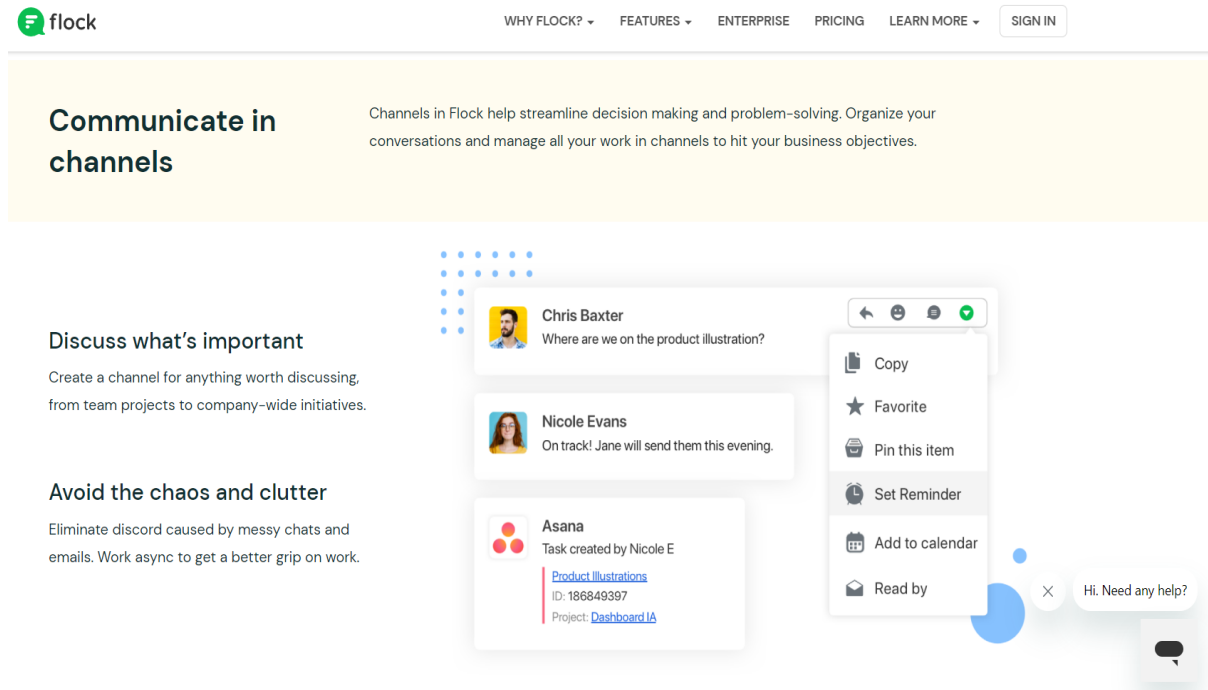


Рисунок 1.2 – Програмне забезпечення Flock [14]

Програмне забезпечення Flock підтримує інтеграцію зі сторонніми програмами, зокрема, з GitHub, Google Drive, Jira, Trello та ін. Крім того, програмне забезпечення Flock підтримує створення опитувань, отримання відгуків та думок, а також встановлення нагадувань для підвищення продуктивності вашої команди. Програмне забезпечення Flock надає засоби зручного спілкування з командою за допомогою функції перетягування, можливість закріплення важливих файлів та повідомлень та доступ до них у потрібний момент. Flock також надає користувачам можливість створення власних застосунків за допомогою Flock API. Від спливаючих вікон та віджетів до бічної панелі та команд зі скісною рисою, ви можете додати до цих додатків високо функціональні можливості. Flock строго контролює права користувачів та дані за допомогою засобів шифрування даних [14].

Програмне забезпечення Flock розширює можливості команд, надаючи їм широкий спектр вбудованих інструментів продуктивності, таких як спільні нотатки, нагадування, завдання і опитування. Користувачі можуть зв'язуватися з віддаленими командами через аудіо та відео дзвінки. Спільний доступ до екрану дозволяє командам ефективніше співпрацювати та обмінюватися

інформацією. Програмне забезпечення Flock також глибоко інтегрується з різними сервісами. Магазин додатків Flock дозволяє командам підключати програми до Flock, що усуває потребу в перемиканні між кількома інструментами для виконання роботи [14].

Функції Flock, такі як розгортання останніх розмов, прості комбінації клавіш і попередній перегляд останнього повідомлення, дозволяють командам зручно використовувати чати, не переходячи між вкладками. Програмне забезпечення Flock дозволяє легко перемикатися між командами, розмовами, програмами. Flock Apps ідеально поєднується з основним продуктом, забезпечуючи більш інтегрований досвід [14].

Програмне забезпечення Flock має такі особливості [14]:

- спільні нотатки та завдання: користувачі програмного забезпечення Flock можуть легко створювати спільні нотатки та завдання, які допомагають у координації роботи команди [14];

- аудіо та відео дзвінки: програмне забезпечення Flock дозволяє здійснювати звукові та відео дзвінки між користувачами, полегшуючи комунікацію [14];

- інтеграція з іншими програмами: програмне забезпечення Flock підтримує інтеграцію з різноманітними сторонніми програмами, такими як Google Drive, Google Calendar, JIRA, тощо, що дозволяє командам зручно використовувати різні інструменти у єдиному інтерфейсі [14];

- функції співпраці: у програмному забезпеченні Flock є функції спільного доступу до екрану, створення опитувань та отримання зворотного зв'язку, що полегшує спільну роботу та прийняття рішень [14];

- простий пошук і навігація: програмне забезпечення Flock дозволяє швидко знаходити потрібні повідомлення, канали чи контакти, а також зручна навігація допомагає легко перемикатися між командами та розмовами [14];

- магазин доповнень: програмне забезпечення Flock має магазин доповнень, де користувачі можуть знайти та підключити різноманітні доповнення для поліпшення своєї робочої ефективності [14];

– шифрування та безпека: програмне забезпечення Flock забезпечує шифрування даних та використання надійних сервісів, що забезпечує безпеку конфіденційної інформації користувачів [14].

Серед недоліків програмного забезпечення Flock для обміну текстовими повідомленнями можна відзначити обмежену масштабованість, оскільки для деяких великих компаній програмне забезпечення Flock може бути менш ефективним рішенням через обмежену масштабованість функцій та можливостей [14].

Програмне забезпечення Nifty (рис. 1.3) є універсальний робочий простір, який спрощує співпрацю між людьми, проектами та функціями у вашій організації. Програмне забезпечення Nifty підтримує процеси чатування, керування завданнями, цілями, документами та файлами в одному місці. Nifty дозволяє зосередитися на роботі, не витрачаючи час на переключення між інструментами. Програмне забезпечення Nifty сприяє скороченню часу розробки проекту та покращенню продуктивності команди, оскільки всі важливі функції керування проектом об'єднуються в одному програмному забезпеченні [15].

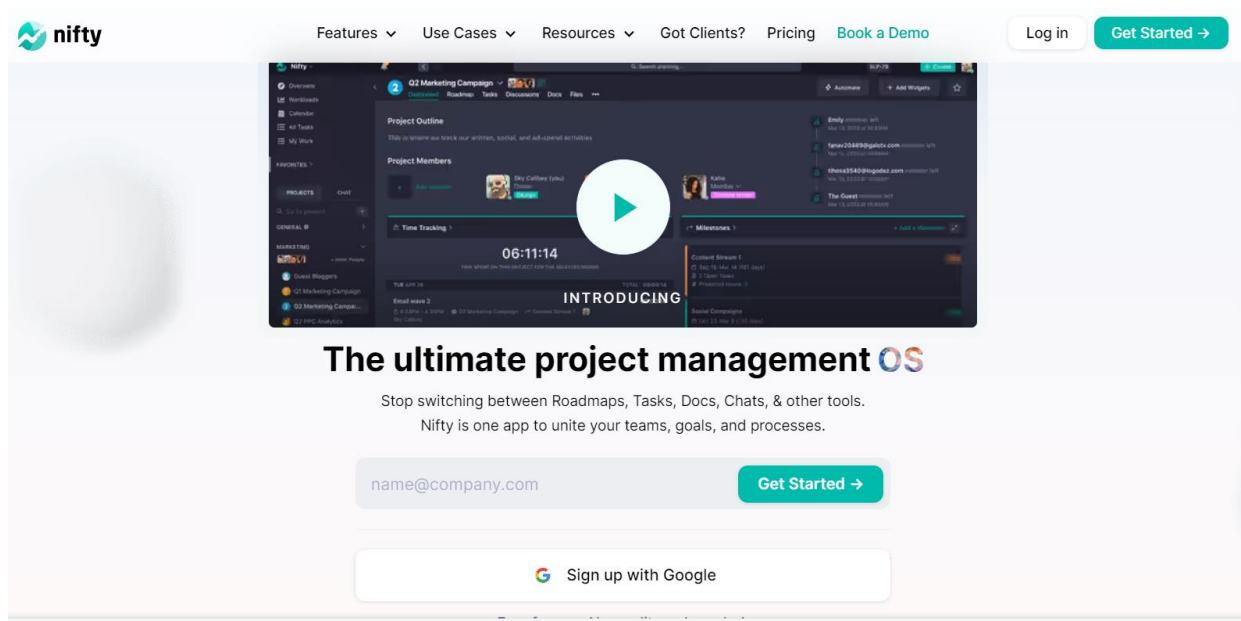


Рисунок 1.3 – Програмне забезпечення Nifty [15]

Програмне забезпечення Nifty забезпечує узгодженість між всіма зацікавленими сторонами, що допомагає виконувати організаційні цілі вчасно. Програмне забезпечення Nifty побудовано для посилення міжвідомчої співпраці, і він має вбудовані засоби чатування та вбудований календар, який можна інтегрувати з Google, Outlook, а також обмінюватися файлами та документами [15].

Програмне забезпечення Nifty дозволяє перетягувати файли, зображення та документи в обговорення, а також посилатися для виконання завдання, щоб уникнути поганої співпраці. Груповий обмін повідомленнями дозволяє створювати спеціальні чати [15]. Дошки обговорень надають можливість командам спілкуватися та коментувати окремі завдання. Вбудовані канали обговорень дозволяють кожному проекту мати свій ланцюжок обговорень, що сприяє дотриманню графіку [15].

Функція перевірки у Nifty дозволяє кожному користувачеві коментувати файли, завантажені безпосередньо в платформу [15].

Програмне забезпечення Nifty має такі особливості [15]:

- централізоване управління: програмне забезпечення Nifty об'єднує чати, завдання, цілі, документи та файли в одному місці, забезпечуючи зосередженість команди на роботі [15];

- зручне спілкування: програмне забезпечення Nifty дозволяє користувачам перетягувати файли, зображення та документи в обговорення, спрощуючи співпрацю та комунікацію в команді [15];

- дошки обговорень: програмне забезпечення Nifty надає можливість використовувати дошки для спілкування та коментування окремих завдань, що полегшує організацію робочого процесу [15];

- центри чатування: груповий обмін повідомленнями дозволяє створювати спеціальні чати для збереження розмов на конкретні теми, що сприяє збереженню порядку в комунікації [15];

– функція перевірки: користувачі можуть коментувати файли безпосередньо в Nifty, що полегшує спільну роботу над документами та забезпечує ефективний обмін думками [15].

Серед недоліків програмного забезпечення Nifty для обміну текстовими повідомленнями можна відзначити обмежену масштабованість, оскільки для деяких великих компаній програмне забезпечення Nifty може бути менш ефективним рішенням через обмежену масштабованість функцій та можливостей [15].

Порівняльну характеристику програмного забезпечення для підтримки процесу текстового спілкування наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для підтримки процесу текстового спілкування

Критерій порівняння	Brosix [13]	Flock [14]	Nifty [15]
Необмежена передача файлів	+	—	—
Засоби API для можливості використання іншими програмними продуктами	—	+	—
Спільне використання екрану	+	+	—
Спільні нотатки та завдання	—	+	+
Безкоштовність	—	+—	+—
Можливості інтеграції з іншими сервісами	+—	+	+—

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для підтримки процесу текстового спілкування. Проте деякі програмні засоби для підтримки процесу текстового спілкування можуть мати проблеми з безпекою даних, зокрема, може бути вразливим до атак зловмисників, витоку даних або

несанкціонованого доступу до особистої інформації користувачів. Інші програмні засоби текстового спілкування можуть мати або зайві функції, непотрібні деяким користувачам, або, навпаки, мати недостатній функціонал, що залежить від потреб конкретних груп користувачів. Деякі програми для підтримки процесу текстового спілкування можуть мати обмеження на кількість користувачів, які можуть брати участь у групових чатах або надсилати повідомлення одночасно.

Тому актуальною є розробка програмного забезпечення для підтримки процесу текстового спілкування.

При розробці програмного забезпечення для підтримки процесу текстового спілкування необхідно забезпечити такі функціональні вимоги:

- підтримка можливості відправлення текстових повідомлень;
- підтримка можливості автоматичного призначення імені користувачу для можливості спілкування за допомогою текстових повідомлень;
- можливість пошуку інших користувачів через мережу Bluetooth;
- можливість встановлення зв'язку з іншими пристроями;
- підтримка можливості відправлення текстових повідомлень на електронну пошту;
- підтримка можливості авторизації та реєстрації користувачів.

1.3 Висновки за розділом 1

Визначено, що текстове спілкування або обмін текстовими повідомленнями полягає у відправленні літерно-цифрових текстів між різними портативними пристроями, такими як мобільні телефони, планшети або ноутбуки через бездротові або інші мережі, що забезпечуються операторами зв'язку. Проаналізовано різні типи повідомлень. Відзначено, що при розробці програмного забезпечення для текстового спілкування важливо забезпечити прогнозне надсилання текстових повідомлень, що являє собою функцію, яка передбачає автоматичне виведення повного слова після введення лише його

перших літер. Проаналізуємо процес передачі повідомлень. Відзначено, що при розробленні програмного забезпечення для текстового спілкування важливо врахувати, що споживчі можуть очікувати, що обмін повідомленнями буде ще більш динамічним, оскільки відео, голос та високоякісні зображення стануть все більш важливими складовими багатогранного та стандартного досвіду обміну повідомленнями. Відзначено, що при розробці програмного забезпечення для підтримки процесу текстового спілкування важливо надати можливість користувачу надсилати повідомлення з можливостями подання невербальних знаків, які можуть бути виражені у спеціальних картинках. При розробці програмного забезпечення для текстового спілкування існує можливість отримати доступ до API для обміну повідомленнями з метою створення інтеграцій власного програмного забезпечення з програмними модулями, які взаємодіють із функціоналом чатування.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для підтримки процесу текстового спілкування. Проте деякі програмні засоби для підтримки процесу текстового спілкування можуть мати проблеми з безпекою даних, зокрема, може бути вразливим до атак зловмисників, витоку даних або несанкціонованого доступу до особистої інформації користувачів. Інші програмні засоби текстового спілкування можуть мати або зайві функції, непотрібні деяким користувачам, або, навпаки, мати недостатній функціонал, що залежить від потреб конкретних груп користувачів. Деякі програми для підтримки процесу текстового спілкування можуть мати обмеження на кількість користувачів, які можуть брати участь у групових чатах або надсилати повідомлення одночасно. Тому актуальною є розробка програмного забезпечення для підтримки процесу текстового спілкування.

Сформульовано функціональні вимоги до програмного забезпечення для підтримки процесу текстового спілкування.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для підтримки процесу текстового спілкування було використано мову програмування Java [16], [17].

Java – є мовою програмування, яка використовується для розробки програмного забезпечення різного призначення, синтаксис якої заснований на синтаксисі мов програмування C і C++. Java не залежить від архітектури комп'ютера, є досить простою в освоєнні та використанні об'єктно-орієнтованою та безпечною мовою програмування [16], [17].

Оскільки в Java немає покажчиків, програмісти не можуть випадково отримати доступ до неправильної адреси пам'яті, що робить її використання більш безпечним. Мова схожа на C і C++, включаючи підтримку циклів і операторів, має вбудовані типи даних, включаючи логічний тип. Масиви у цій мові трактуються як об'єкти і мають гарантований доступ [16], [17].

Існує декілька варіантів використання мови Java. Java Standard Edition є стандартним набором засобів розробки Java, придатних для розробки більшості програм [16], [17]. Java Enterprise Edition є набором інструментів розробки Java для створення більшості великих корпоративних програм. Java Micro Edition включає в себе засоби розробки Java для розробки мобільного програмного забезпечення. Java Card Edition є набором засобів розробки Java для розумних карт [16], [17].

Поточна версія Java офіційно називається Java Standard Edition і включає Java Development Kit і Java Runtime Environment [16], [17]. Пакет Java Development Package містить усі інструменти та засоби програмування, необхідні для розробки програмного забезпечення Java [16], [17].

Java вирішує деякі проблеми, які існують в C++, такі як використання покажчиків, складність успадкування та файли заголовків. Натомість, Java пропонує широкий спектр функціональності, такий як обробка винятків, паралелізм та графічний інтерфейс користувача [16], [17].

Java має всі потрібні можливості для розвитку сучасних об'єктно-орієнтованих програм, зокрема, для визначення класів і створення об'єктів, керування видимістю елементів за допомогою ключових слів, автоматичне керування пам'яттю збирачем сміття, можливість успадкування властивостей від класу, обробка програмних винятків для управління помилками під час виконання [16], [17].

Програми Java можна запускати у браузері за допомогою аплетів, що дозволяє їм працювати на різних платформах. Аплети Java мають деякі обмеження безпеки, такі як відсутність доступу до файлів на локальному комп'ютері чи запуск зовнішніх програм [16], [17].

Аплети завантажуються веббраузером через завантажувач класів, а потім вони виконуються прямо в ньому. Аплети можна використовувати для показу тексту, зображень, аудіо, відео та діалогових вікон [16], [17].

Створення графічного інтерфейсу користувача в Java досить просте і зрозуміле. Для цього бібліотека виконання Java містить різні компоненти, які забезпечують базові графічні можливості та функції вікон, надають комплексний набір графічного інтерфейсу користувача та дозволяють виконувати складні операції з обробки зображень та графіки [16], [17].

За допомогою відповідного набору інструментів можна створювати текст, лінії, кіл, прямокутники та багатокутники в будь-якому місці вікна. Колір можна визначити за різними моделями [16], [17].

Java також надає спеціальний аудіо API для відтворення та керування аудіо файлами, а зображення можна легко редагувати та відображати за допомогою пакетів зображень. У графічному інтерфейсі користувача інструкції користувача стають потрібними під час взаємодії з програмним забезпеченням. Java вирішує цю проблему за допомогою створення та надсилання подій до програм, коли користувач взаємодіє з елементами інтерфейсу [16], [17].

Події можуть бути будь-якого типу, але об'єкти, які їх приймають, повинні реалізувати інтерфейс приймача подій, що працюють з графічним

інтерфейсом користувача, оскільки багато елементів графічного інтерфейсу використовуються в сучасних інтерфейсах. Однією з ключових переваг цієї мови є висока переносимість програмного коду між платформами [16], [17].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для підтримки процесу текстового спілкування наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для підтримки процесу текстового спілкування

Критерій порівняння мов програмування	Мова програмування		
	Java	Python	C++
Автоматичне управління пам'яттю	+	+	+—
Використання для створення програм різного призначення	+	+	+—
Висока переносимість коду	+	+	+—
Зручність для створення ПЗ для підтримки процесу текстового спілкування	+	—	—
Компіляція програм	+	—	+

Отже, для реалізації програмного забезпечення для підтримки процесу текстового спілкування обрано мову програмування Java, яка є мовою програмування, яка використовується для розробки програмного забезпечення різного призначення, синтаксис якої заснований на синтаксисі мов програмування C і C++. Java не залежить від архітектури комп'ютера, є досить простою в освоєнні та використанні об'єктно-орієнтованою та безпечною мовою програмування.

2.2 Вибір середовища розробки для створення програмного забезпечення для підтримки процесу текстового спілкування

В якості середовища розробки для створення програмного забезпечення для підтримки процесу текстового спілкування було обрано середовище Android Studio [18], [19].

Середовище розробки Android Studio об'єднує усі складові програми, включаючи дані та ресурси, у файл APK (пакет Android, файл з розширенням .apk). Файл APK містить всі програмні компоненти Android та файли пристрою, які вимагаються програмі. Після встановлення на пристрої користувача кожна програма працює у власному ізольованому та безпечному середовищі [18], [19].

Операційна система Android базується на багато користувачській системі Linux, де кожна програма має власного користувача. Система надає кожній програмі Linux унікальний ідентифікатор користувача, і система встановлює права доступу до файлів, щоб доступ до них мали лише відповідні користувачі. Кожен процес має свою власну віртуальну машину, що ізольована від інших програм, і кожна програма працює у власному процесі Linux. Android запускає процеси лише тоді, коли потрібно запустити компонент програми, і зупиняє їх, коли вони більше не потрібні або для звільнення пам'яті [18], [19].

Такий підхід гарантує, що кожна програма має доступ лише до необхідних їй компонентів та функцій, що робить оточення дуже безпечним для використання, бо програма не має доступу до недоступних системних функцій [18], [19].

Середовище розробки Android Studio розроблено офіційно компанією Google, є стандартом середовища розробки для платформи Android. Порівняно з іншими засобами, використання Android Studio виявляється більш ефективним, оскільки воно має простий інтерфейс, швидкий процес навчання, мову програмування, широкі можливості, безкоштовність та високий рівень

популярності серед розробників у всьому світі. Крім того, завдяки Google, це середовище постійно оновлюється і вдосконалюється, так що вам не потрібно хвилюватися про його підтримку [18], [19].

Середовище розробки Android Studio може бути використано для розробки мобільних додатків. Середовище розробки Android Studio відзначається своєю простотою, продуктивністю та гнучкістю для розробників. Середовище розробки Android Studio включає в себе вбудовану панель керування, яка автоматизує роботу з моделями даних програми, що спрощує і прискорює розробку [18], [19].

Середовище розробки Android Studio також підтримує ряд сторонніх бібліотек і додатків, що дозволяє розширювати його функціональність за потреби [18], [19].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для підтримки процесу текстового спілкування наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для підтримки процесу текстового спілкування

Критерій порівняння середовищ розробки	Середовища розробки		
	Android Studio	Eclipse	IntelliJ IDEA
Висока швидкодії	+	+—	+
Безкоштовність	+	+	+—
Вбудовані інструменти	+	+—	+
Підтримка спільної розробки	+	—	+
Зручність розробки GUI	+	+—	+—

Таким чином, для створення програмного забезпечення для підтримки процесу текстового спілкування обрано середовище розробки Android Studio, що у порівнянні з іншими засобами виявляється більш ефективним, оскільки воно має простий інтерфейс, швидкий процес навчання, широкі можливості, безкоштовність та високий рівень популярності серед розробників у всьому світі. Крім того, завдяки Google, це середовище постійно оновлюється і вдосконалюється, тому розробнику не потрібно хвилюватися про його підтримку.

2.3 Висновки за розділом 2

Для реалізації програмного забезпечення для підтримки процесу текстового спілкування обрано мову програмування Java, яка є мовою програмування, що використовується для розробки програмного забезпечення різного призначення, синтаксис якої заснований на синтаксисі мов програмування C і C++. Java не залежить від архітектури комп'ютера, є досить простою в освоєнні та використанні об'єктно-орієнтованою та безпечною мовою програмування.

Для створення програмного забезпечення для підтримки процесу текстового спілкування обрано середовище розробки Android Studio, що у порівнянні з іншими засобами виявляється більш ефективним, оскільки воно має простий інтерфейс, швидкий процес навчання, широкі можливості, безкоштовність та високий рівень популярності серед розробників у всьому світі. Крім того, завдяки Google, це середовище постійно оновлюється і вдосконалюється, тому розробнику не потрібно хвилюватися про його підтримку.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення для підтримки процесу текстового спілкування

Структуру програмного забезпечення для підтримки процесу текстового спілкування наведено у вигляді діаграми класів на рис. 3.1.

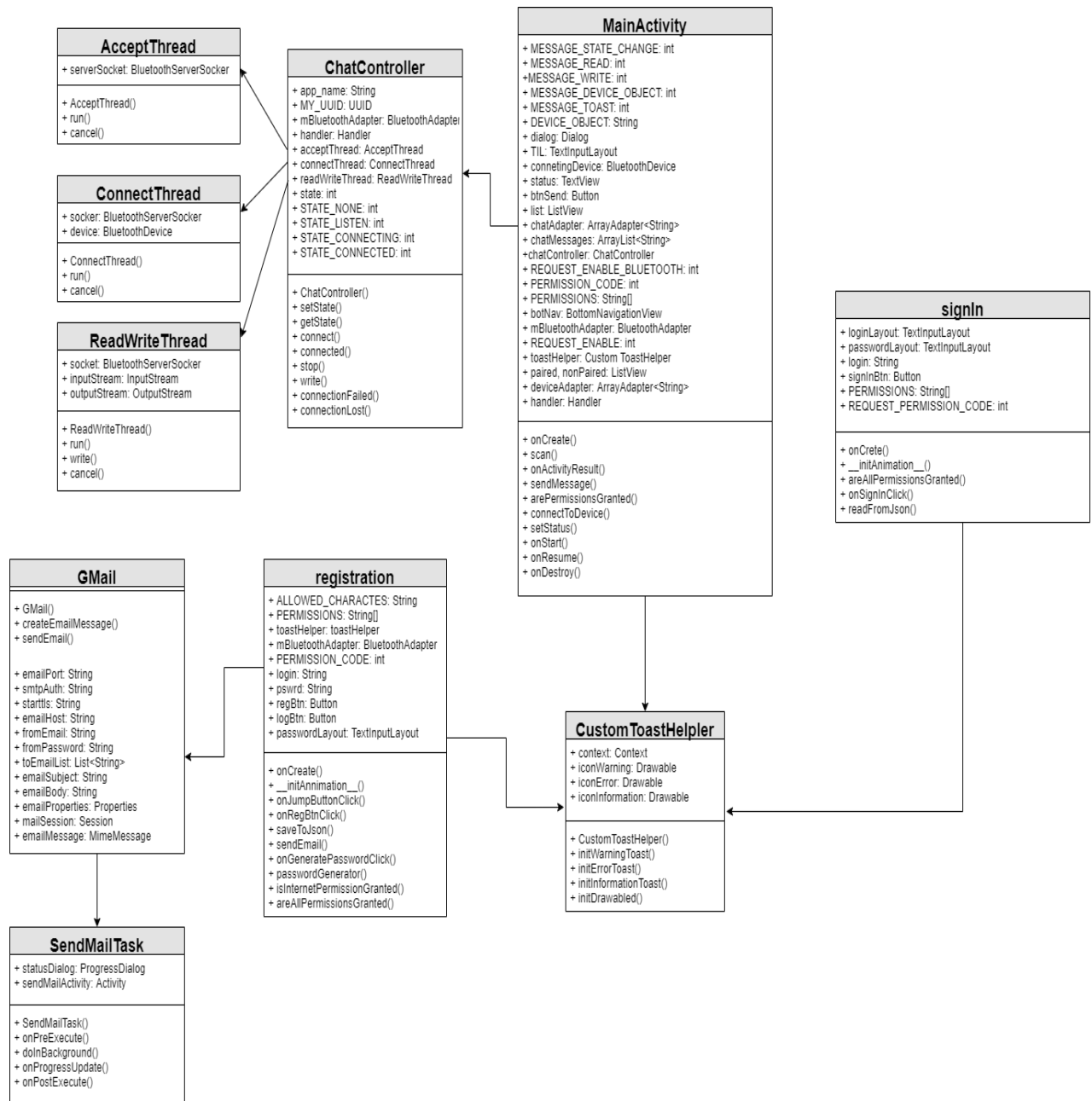


Рисунок 3.1 – Структура програмного забезпечення для підтримки процесу текстового спілкування

Як видно з рис. 3.1, при створенні програмного забезпечення для підтримки процесу текстового спілкування було розроблено декілька класів, зокрема, класи MainActivity, ChatController, registration, Gmail та ін.

Клас ChatController було розроблено для забезпечення функцій чатування. Цей клас має властивості, що забезпечують роботи з його об'єктами. До таких властивостей, зокрема, належать такі: app_name (назва застосунку), STATE_NONE (немає підключення), STATE_CONNECTING (стан підключення), STATE_LISTEN (стан прослуховування) та ін. Методи класу реалізують повернення стану (synchronized void setState(int state)), завдання стану (synchronized int getState()), початку роботи об'єкту (synchronized void connected()), виведення повідомлення про переривання зв'язку (void connectionLost()) та ін.

Клас CustomToastHelper було розроблено для забезпечення роботи зі спливаючими повідомленнями. Цей клас має властивості, що забезпечують роботи з його об'єктами. До таких властивостей, зокрема, належать такі: iconInformation (графічне подання інформаційного повідомлення), context (контекст повідомлення), iconError (графічне подання повідомлення про помилку) та ін. Методи класу реалізують генерацію спливаючого повідомлення, яке звертає увагу користувача (void initWarningToast(String message)), генерацію інформаційного повідомлення (void initInformationToast(String message)), генерацію повідомлення про помилку (void initErrorToast(String message)) та ін.

Класи GMail та SendMailtask призначено для роботи з повідомленнями, які надсилаються на електронну пошту. Властивостями цих класів є змінні, що відображають порт пошти (emailPort), адресу електронної пошти (fromEmail), пароль від пошти відправника (fromPassword), зміст текстового повідомлення (emailBody) та ін. Методи класів GMail та SendMailtask реалізують процеси створення повідомлення (MimeMessage createEmailMessage()), відправлення повідомлення (void sendEmail()) та ін.

Розроблено й інші класи програмного забезпечення для підтримки процесу текстового спілкування.

Діаграму компонентів програмного забезпечення для підтримки процесу текстового спілкування наведено на рис. 3.2.

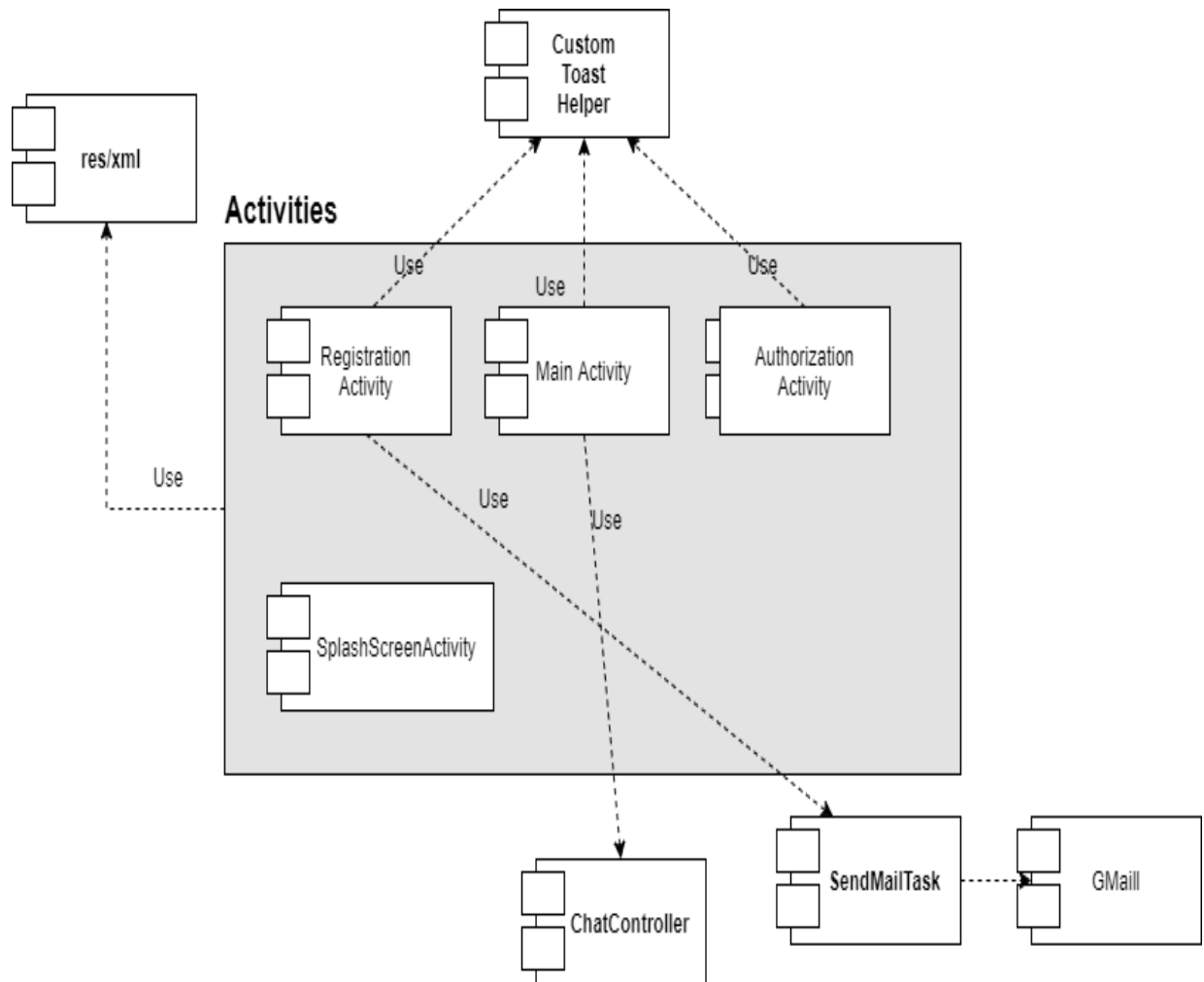


Рисунок 3.2 – Діаграма компонентів програмного забезпечення для підтримки процесу текстового спілкування

3.2 Функціонування програмного забезпечення для підтримки процесу текстового спілкування

Функціонування програмного забезпечення для підтримки процесу текстового спілкування подамо за допомогою схеми, зображеної на рис. 3.3.

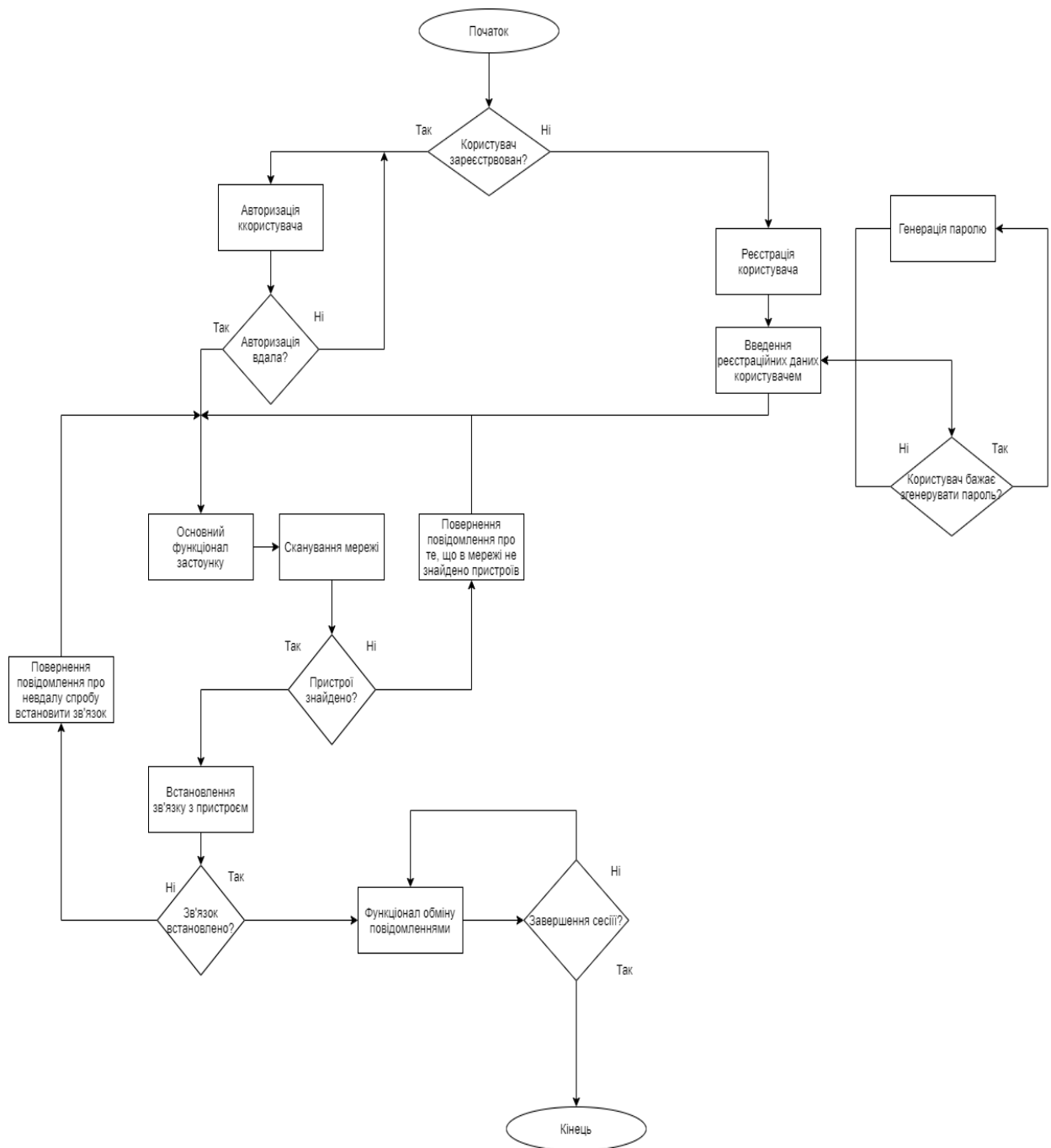


Рисунок 3.3 – Схема функціонування програмного забезпечення для підтримки процесу текстового спілкування

Для задоволення різних потреб користувачів у програмному забезпеченні для підтримки процесу текстового спілкування використовується архітектура клієнт-сервер. У зв'язку з цим, прийнято рішення не використовувати традиційну базу даних, адже права доступу користувача автоматично призначаються мережі Bluetooth на основі доступного імені пристрою. Отже, у випадку зміни імені користувача, він повинен буде

zareestruватися znovu. Це призводить до того, що обліковий запис, що містить лише логін і пароль, підключається до бази даних і створює лише один файл замість додавання нового файлу, що дозволяє прискорити процес. Крім того, основні функції програми знаходяться в основному її класі, який обробляє все, що не пов'язано з даними та інтерфейсом користувача. Інтерфейс користувача використовує набір тегів для визначення порядку та кількості візуальних елементів керування, а також інші ресурси для покращення дизайну та зовнішнього вигляду інтерфейсу.

Схему взаємодії між компонентами програмного забезпечення для підтримки процесу текстового спілкування наведено на рис. 3.4.

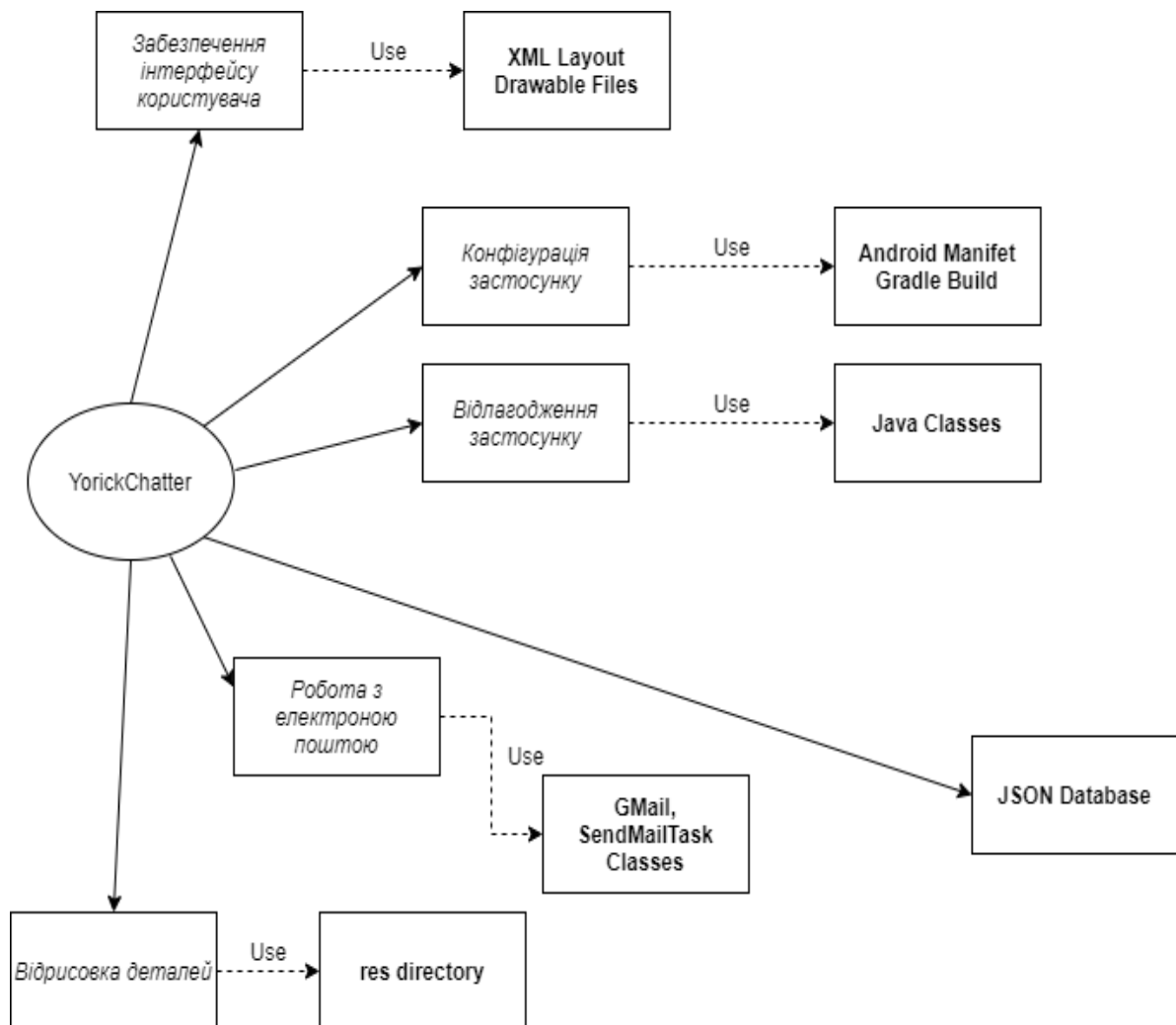


Рисунок 3.4 – Схема взаємодії між компонентами програмного забезпечення для підтримки процесу текстового спілкування

Алгоритм роботи програми полягає у наступному:

- реєстрація користувачів за допомогою автоматично наданого імені для входу;
- створення паролів користувачами.
- надсилання реєстраційної інформації на електронну пошту після підключення до Інтернету (якщо є доступ);
- управління Bluetooth з програми, пошук пристроїв за допомогою функції сканування;
- взаємодія користувачів через надсилання текстових повідомлень на підключені пристрої.

3.3 Опис алгоритму реєстрації користувачів програмного забезпечення для підтримки процесу текстового спілкування

Для функціонування програмного забезпечення для підтримки процесу текстового спілкування важливим є процес реєстрації користувачів. У процесі реєстрації користувачів програмне забезпечення автоматично надає доступ користувачеві, використовуючи назву пристрою в мережі Bluetooth як ім'я для входу (рис. 3.5). Наприклад, якщо пристрій має назву "MyTel543", при запуску програми воно використовується як ім'я для входу.

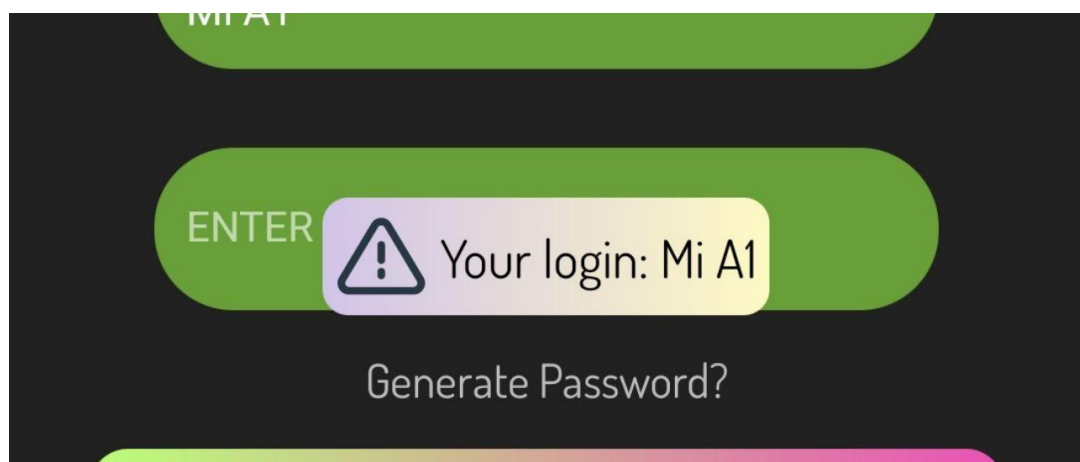


Рисунок 3.5 – Приклад реєстрації користувачів програмного забезпечення для підтримки процесу текстового спілкування

Після реєстрації генерується файл типу .json, де зберігається інформація у форматі "ключ:значення" - "логін:{user_login}, пароль:{user_password}". Крім того, користувач може створити пароль під час реєстрації (рис. 3.6). Пароль складається з 8 символів, які вибираються автоматично, і може містити текстові, числові та спеціальні символи.

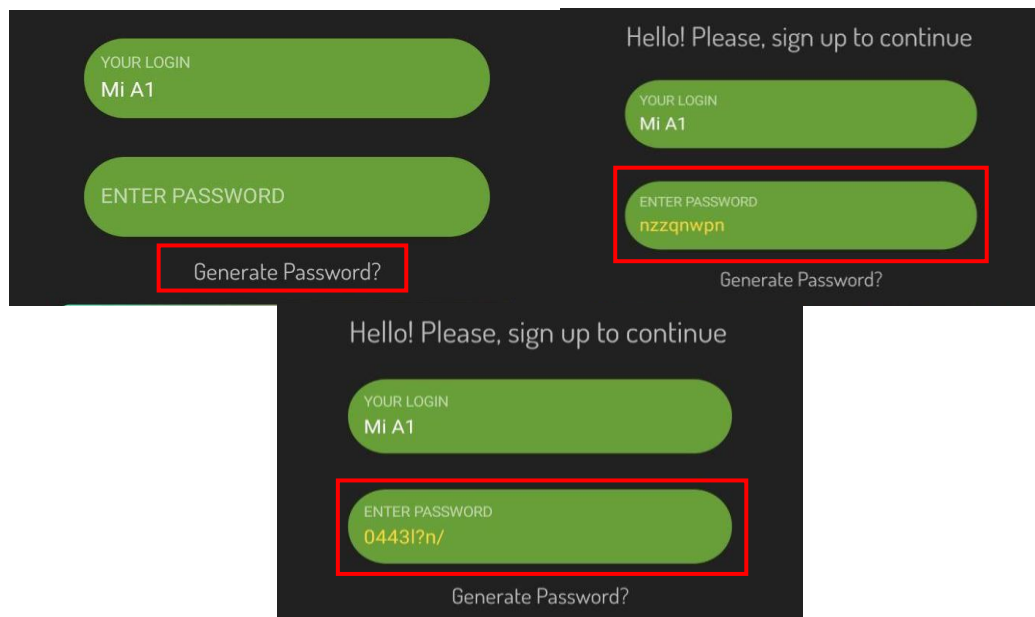


Рисунок 3.6 – Приклад створення паролю користувачів програмного забезпечення для підтримки процесу текстового спілкування

Якщо користувач має доступ до Інтернету (що не є обов'язковим, оскільки програма призначена для використання без постійного підключення до Інтернету), він може відправляти повідомлення самому собі (спочатку вказує свою електронну адресу), відправка виконується через електронну пошту.

3.4 Проєктування інтерфейсу програмного забезпечення для підтримки процесу текстового спілкування

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки процесу текстового спілкування виконано з

урахуванням вимог технічного завдання. При розробленні програмного забезпечення для підтримки процесу текстового спілкування враховані функціональні вимоги до програми:

- підтримка можливості відправлення текстових повідомлень;
- підтримка можливості автоматичного призначення імені користувачу для можливості спілкування за допомогою текстових повідомлень;
- можливість пошуку інших користувачів через мережу Bluetooth;
- можливість встановлення зв'язку з іншими пристроями;
- підтримка можливості відправлення текстових повідомлень на електронну пошту;
- підтримка можливості авторизації та реєстрації користувачів.

Інтерфейс сторінки реєстрації користувача програмного забезпечення для підтримки процесу текстового спілкування наведено на рис. 3.7.

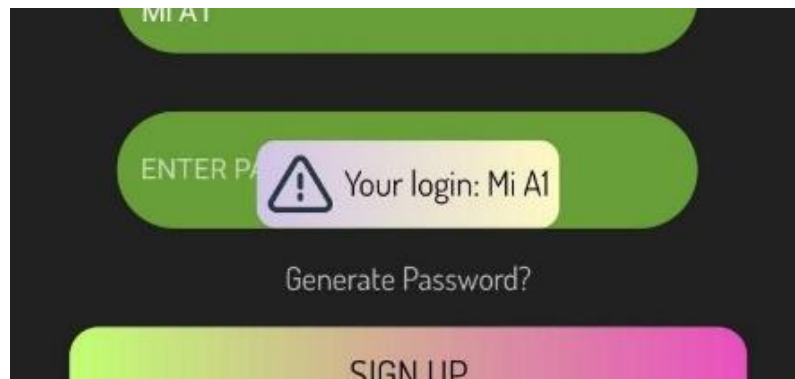


Рисунок 3.7 – Інтерфейс сторінки реєстрації користувача програмного забезпечення для підтримки процесу текстового спілкування

Область сторінки реєстрації користувача в процесі генерації паролю наведено на рис. 3.8.

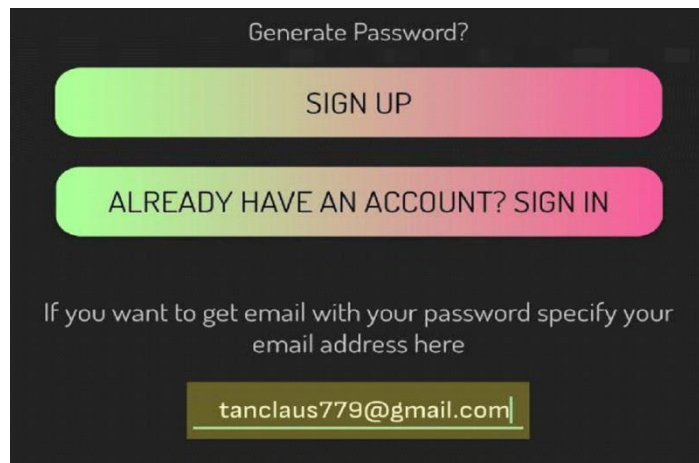


Рисунок 3.8 – Область сторінки реєстрації користувача в процесі генерації паролю

Форма сторінки з вікном для чатування наведена на рис. 3.9.

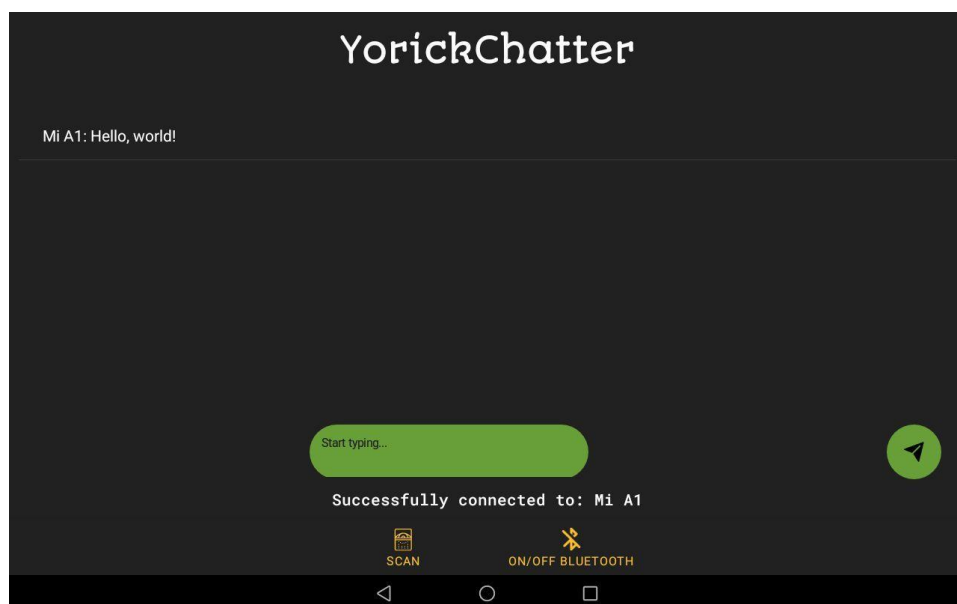


Рисунок 3.9 – Форма сторінки з вікном для чатування

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для підтримки процесу текстового спілкування.

Запропоновано структуру програмного забезпечення для підтримки процесу текстового спілкування. При створенні програмного забезпечення для

підтримки процесу текстового спілкування було розроблено декілька класів, зокрема, класи MainActivity, ChatController, registration, Gmail та ін.

Описано функціонування програмного забезпечення для підтримки процесу текстового спілкування.

Описано особливості реалізації програмного забезпечення для підтримки процесу текстового спілкування. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки процесу текстового спілкування.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для підтримки процесу текстового спілкування. Програма написана на мові Java, яка є мовою програмування, що використовується для розробки програмного забезпечення різного призначення, синтаксис якої заснований на синтаксисі мов програмування C і C++. В якості середовища розробки для створення програмного забезпечення для підтримки процесу текстового спілкування обрано Android Studio, що у порівнянні з іншими засобами виявляється більш ефективним, оскільки воно має простий інтерфейс, швидкий процес навчання, широкі можливості, безкоштовність та високий рівень популярності серед розробників у всьому світі.

Програмне забезпечення для підтримки процесу текстового спілкування забезпечує виконання таких функцій:

- підтримка можливості відправлення текстових повідомлень;
- підтримка можливості автоматичного призначення імені користувачу для можливості спілкування за допомогою текстових повідомлень;
- можливість пошуку інших користувачів через мережу Bluetooth;
- можливість встановлення зв'язку з іншими пристроями;
- підтримка можливості відправлення текстових повідомлень на електронну пошту;
- підтримка можливості авторизації та реєстрації користувачів.

Для експлуатації програмного забезпечення для підтримки процесу текстового спілкування необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом 500 Мб і більше;
- не менше 50 Мб вільного простору на запам'ятовуючому пристрою телефону або іншого засобу;
- процесор із тактовою частотою не менше 1.2 ГГц;

- наявність Bluetooth;
- встановлена операційна система Android та сервіс віртуальної клавіатури.

Функціональні характеристики розробленого програмного забезпечення для підтримки процесу текстового спілкування наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для підтримки процесу текстового спілкування наведено у додатку Б.

4.2 Характеристики програми для підтримки процесу текстового спілкування

Програмне забезпечення для підтримки процесу текстового спілкування складається з двох основних компонентів — серверної та клієнтської частин. Програмне забезпечення для підтримки процесу текстового спілкування не вимагає наявності доступу до мережі Інтернет, проте необхідним є наявність Bluetooth. При наявності доступу до мережі Інтернет користувач може не тільки обмінюватися повідомленнями з іншими користувачами, а й відправляти повідомлення на електронну пошту.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для підтримки процесу текстового спілкування необхідно запустити відповідний арк-файл.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для підтримки процесу текстового спілкування є логін користувача та пароль, а також текстові повідомлення.

Вихідними даними програмного забезпечення для підтримки процесу текстового спілкування є отримані текстові повідомлення інших користувачів.

4.3.3 Повідомлення

Користувач програмного забезпечення для підтримки процесу текстового спілкування може отримати такі повідомлення: повідомлення про успішне підключення Bluetooth, повідомлення про проблеми зі з'єднанням з іншими пристроєм та ін.

4.4 Виконання програмного забезпечення для підтримки процесу текстового спілкування

Після запуску програмного забезпечення для підтримки процесу текстового спілкування користувач повинен пройти процес реєстрації. У процесі реєстрації програмне забезпечення автоматично надає доступ користувачеві, використовуючи назву пристрою в мережі Bluetooth як ім'я для входу (рис. 4.1). Наприклад, якщо пристрій має назву "Mi A1", при запуску програми воно використовується як ім'я для входу.

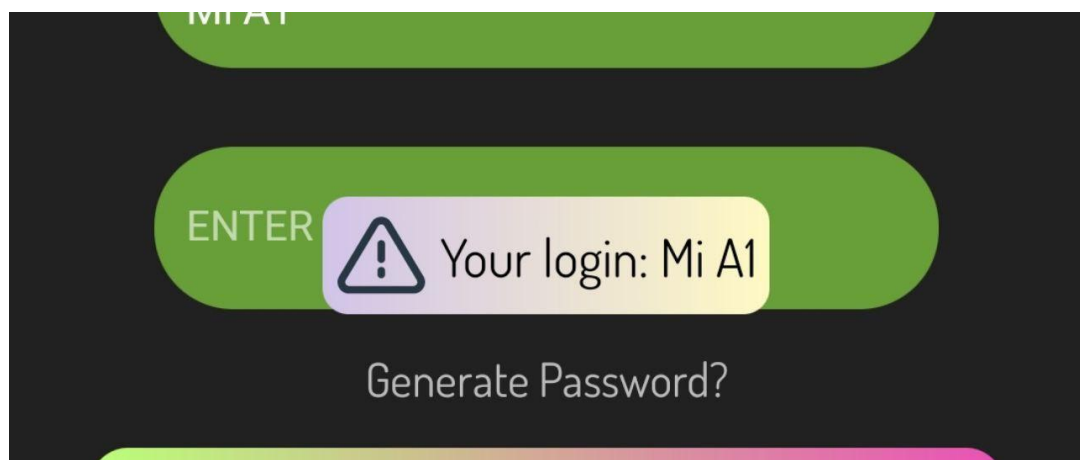


Рисунок 4.1 – Приклад реєстрації користувача програмного забезпечення для підтримки процесу текстового спілкування

Крім того, користувач може створити пароль під час реєстрації (рис. 4.2). Пароль складається з 8 символів, які вибираються автоматично, і може містити текстові, числові та спеціальні символи. Пароль може бути згенеровано автоматично.

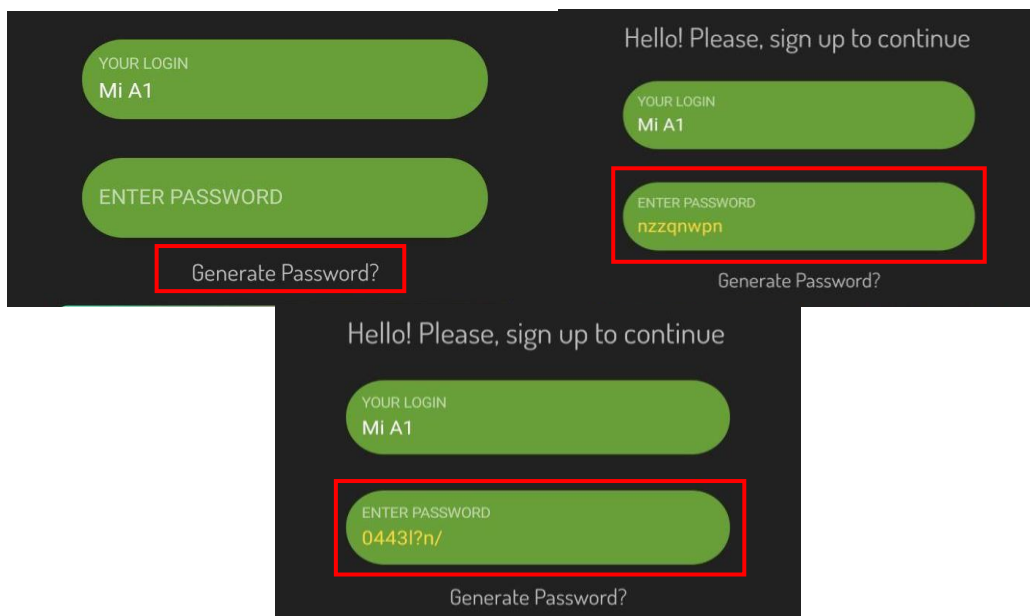


Рисунок 4.2 – Приклад створення паролю користувачів програмного забезпечення для підтримки процесу текстового спілкування

Отримане повідомлення на електронну пошту про успішну реєстрацію наведено на рис. 4.3.

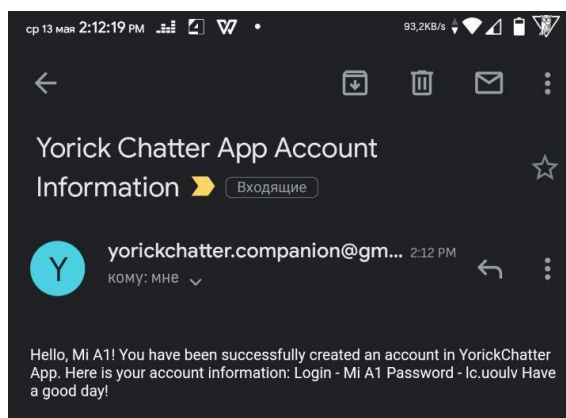


Рисунок 4.3 – Повідомлення на електронну пошту про успішну реєстрацію

Якщо користувач має доступ до Інтернету (що не є обов'язковим, оскільки програма призначена для використання без постійного підключення до Інтернету), він може відправляти повідомлення самому собі (спочатку вказує свою електронну адресу), відправка виконується через електронну пошту.

Після успішної реєстрації користувач може знайти інших користувачів у мережі Bluetooth. Для цього він на головному екрані повинен натиснути кнопку SCAN, після чого відкриється діалогове вікно зі знайденими пристроями, з якими можна проводити сеанси текстового спілкування (рис. 4.4).

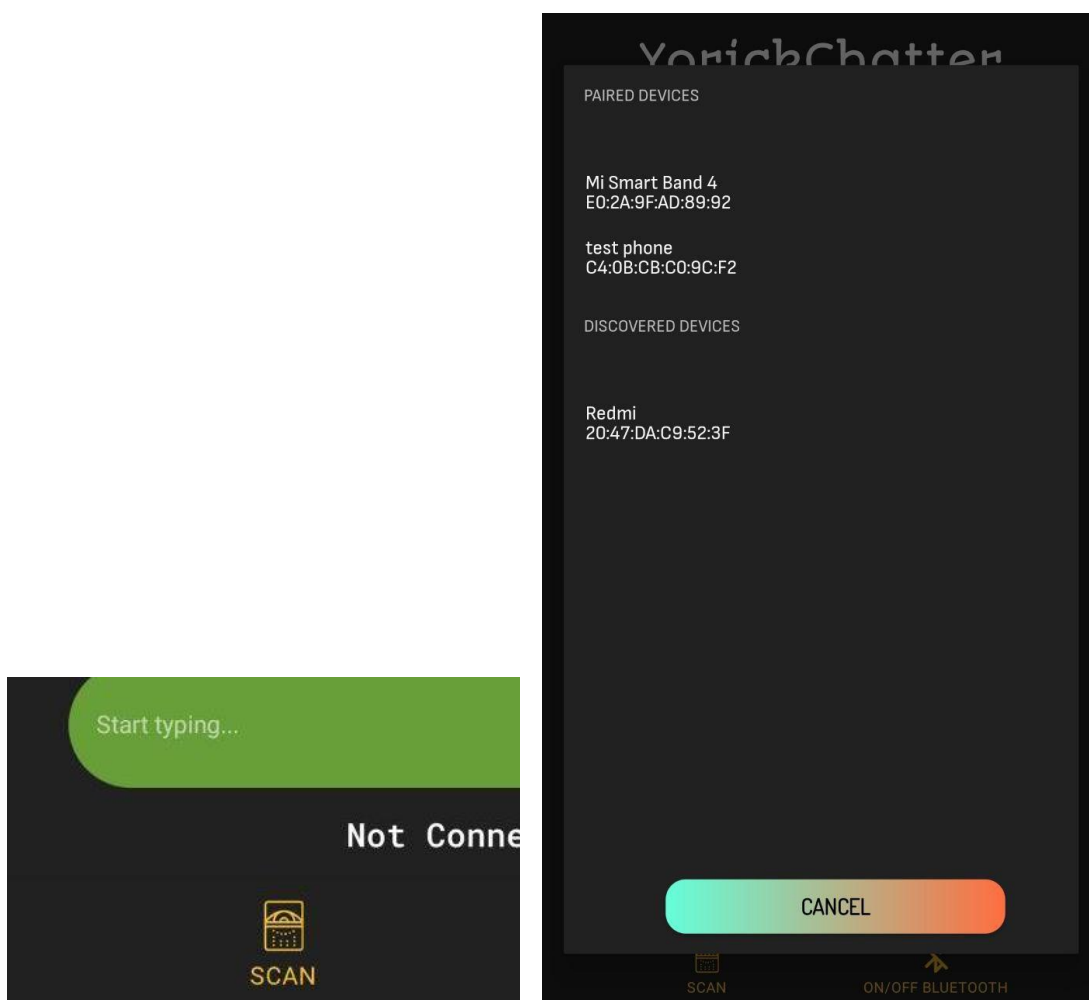


Рисунок 4.4 – Процес та результати пошуку інших користувачів у мережі Bluetooth

Після вибору користувача для спілкування відкриється діалогове вікно (рис. 4.5), де користувачі можуть обмінюватися повідомленнями.

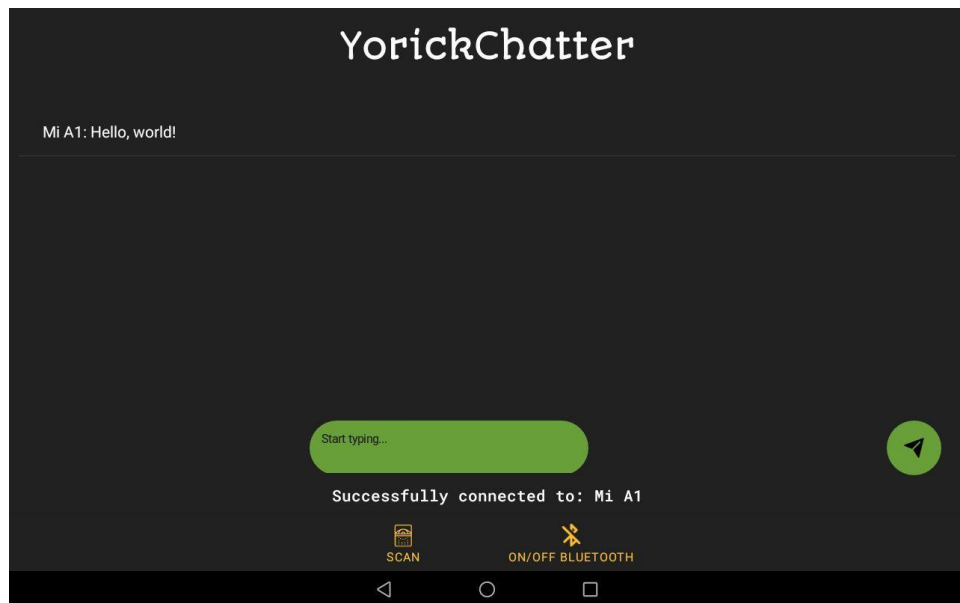


Рисунок 4.5 – Форма сторінки з вікном для чатування

4.5 Тестування програмного забезпечення для підтримки процесу текстового спілкування

Виконано тестування програмного забезпечення для підтримки процесу текстового спілкування.

Під час тестування програмного забезпечення для підтримки текстового спілкування перевірку працездатності таких функцій:

- підтримка можливості відправлення текстових повідомлень;
- підтримка можливості автоматичного призначення імені користувачу для можливості спілкування за допомогою текстових повідомлень;
- можливість пошуку інших користувачів через мережу Bluetooth;
- можливість встановлення зв'язку з іншими пристроями;
- підтримка можливості відправлення текстових повідомлень на електронну пошту;
- підтримка можливості авторизації та реєстрації користувачів.

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

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою процесу текстового спілкування.

4.6 Висновки за розділом 4

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

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для комунікації між людьми.

Визначено, що текстове спілкування або обмін текстовими повідомленнями полягає у відправленні літерно-цифрових текстів між різними портативними пристроями, такими як мобільні телефони, планшети або ноутбуки через бездротові або інші мережі, що забезпечуються операторами зв'язку. Проаналізовано різні типи повідомлень. Відзначено, що при розробці програмного забезпечення для текстового спілкування важливо забезпечити прогнозне надсилання текстових повідомлень, що являє собою функцію, яка передбачає автоматичне виведення повного слова після введення лише його перших літер. Проаналізуємо процес передачі повідомлень. Відзначено, що при розробленні програмного забезпечення для текстового спілкування важливо врахувати, що споживчі можуть очікувати, що обмін повідомленнями буде ще більш динамічним, оскільки відео, голос та високоякісні зображення стануть все більш важливими складовими багатогранного та стандартного досвіду обміну повідомленнями. Відзначено, що при розробці програмного забезпечення для підтримки процесу текстового спілкування важливо надати можливість користувачу надсилати повідомлення з можливостями подання невербальних знаків, які можуть бути виражені у спеціальних картинках. При розробці програмного забезпечення для текстового спілкування існує можливість отримати доступ до API для обміну повідомленнями з метою створення інтеграцій власного програмного забезпечення з програмними модулями, які взаємодіють із функціоналом чатування.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для підтримки процесу текстового спілкування. Проте деякі програмні засоби для підтримки процесу текстового спілкування можуть мати проблеми з безпекою даних, зокрема, може бути

вразливим до атак зловмисників, витоку даних або несанкціонованого доступу до особистої інформації користувачів. Інші програмні засоби текстового спілкування можуть мати або зайві функції, непотрібні деяким користувачам, або, навпаки, мати недостатній функціонал, що залежить від потреб конкретних груп користувачів. Деякі програми для підтримки процесу текстового спілкування можуть мати обмеження на кількість користувачів, які можуть брати участь у групових чатах або надсилати повідомлення одночасно. Тому актуальною є розробка програмного забезпечення для підтримки процесу текстового спілкування.

Сформульовано функціональні вимоги до програмного забезпечення для підтримки процесу текстового спілкування.

Для реалізації програмного забезпечення для підтримки процесу текстового спілкування обрано мову програмування Java, яка є мовою програмування, що використовується для розробки програмного забезпечення різного призначення, синтаксис якої заснований на синтаксисі мов програмування C і C++. Java не залежить від архітектури комп'ютера, є досить простою в освоєнні та використанні об'єктно-орієнтованою та безпечною мовою програмування.

Для створення програмного забезпечення для підтримки процесу текстового спілкування обрано середовище розробки Android Studio, що у порівнянні з іншими засобами виявляється більш ефективним, оскільки воно має простий інтерфейс, швидкий процес навчання, широкі можливості, безкоштовність та високий рівень популярності серед розробників у всьому світі. Крім того, завдяки Google, це середовище постійно оновлюється і вдосконалюється, тому розробнику не потрібно хвилюватися про його підтримку.

Розроблено програмне забезпечення для підтримки процесу текстового спілкування.

Запропоновано структуру програмного забезпечення для підтримки процесу текстового спілкування. При створенні програмного забезпечення для

підтримки процесу текстового спілкування було розроблено декілька класів, зокрема, класи MainActivity, ChatController, registration, Gmail та ін.

Описано функціонування програмного забезпечення для підтримки процесу текстового спілкування.

Описано особливості реалізації програмного забезпечення для підтримки процесу текстового спілкування. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки процесу текстового спілкування.

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

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Text Messaging (Texting or Wireless Messaging) [Electronic resource].
– Access mode : <https://www.techtarget.com/searchmobilecomputing/definition/texting>.
2. Is Messaging The Same As Texting [Electronic resource]. – Access mode : <https://simpletexting.com/blog/is-messaging-the-same-as-texting/>.
3. Texting (Text Messaging) [Electronic resource]. – Access mode : <https://www.thoughtco.com/texting-text-messaging-1692536re>.
4. Texting [Electronic resource]. – Access mode : <https://www.britannica.com/technology/text-messaging>.
5. The Past, Present And Future Of Messaging [Electronic resource]. – Access mode : <https://www.forbes.com/sites/forbestechcouncil/2021/01/06/the-past-present-and-future-of-messaging/?sh=38d0bac9f17f>.
6. Messaging [Electronic resource]. – Access mode : <https://www.ttec.com/glossary/messaging>.
7. The Message: What You Say Matters in Communication [Electronic resource]. – Access mode : <https://ifioque.com/miscellaneous/message>.
8. How Instant Messaging Security Works : <https://computer.howstuffworks.com/e-mail-messaging/instant-messaging-security.htm>.
9. Common Types of Mobile Messages Explained [Electronic resource]. – Access mode: <https://www.textline.com/blog/types-of-mobile-messages>.
10. Messaging API Overview [Electronic resource]. – Access mode: <https://platform.text.com/docs/messaging/>.
11. Best Instant Messaging Platforms for your Business [Electronic resource]. – Access mode : <https://www.twilio.com/en-us/blog/instant-messaging-platforms>.

12. Best Instant Messaging software for Business [Electronic resource]. – Access mode : <https://www.troopmessenger.com/blogs/instant-messaging-software-for-business>.

13. Brosix [Electronic resource]. – Access mode : <https://www.brosix.com/>.

14. Flock [Electronic resource]. – Access mode : <https://www.flock.com/>.

15. Nifty [Electronic resource]. – Access mode : <https://niftypm.com/>.

16. Java für Desktop-Anwendungen abrufen [Electronic resource]. – Access mode : <https://www.java.com/de/>.

17. Java Data Types [Electronic resource]. – Access mode : https://www.w3schools.com/java/java_data_types.asp.

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

19. Different Types of Activities in Android Studio [Electronic resource]. – Access mode : <https://www.geeksforgeeks.org/different-types-of-activities-in-android-studio/>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для підтримки процесу текстового спілкування.

A.1 Підстава для розробки

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

A.2 Призначення розробки

Програмний продукт призначений для підтримки текстового спілкування.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для підтримки текстового спілкування забезпечує виконання таких функцій:

- підтримка можливості відправлення текстових повідомлень;
- підтримка можливості автоматичного призначення імені користувачу для можливості спілкування за допомогою текстових повідомлень;
- можливість пошуку інших користувачів через мережу Bluetooth;
- можливість встановлення зв'язку з іншими пристроями;
- підтримка можливості відправлення текстових повідомлень на електронну пошту;
- підтримка можливості авторизації та реєстрації користувачів.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для підтримки текстового спілкування повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення для підтримки текстового спілкування повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для підтримки текстового спілкування необхідна наявність персонального комп'ютера.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення для підтримки текстового спілкування необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом 500 Мб і більше;
- не менше 50 Мб вільного простору на запам'ятовуючому пристрою телефону або іншого засобу;
- процесор із тактовою частотою не менше 1.2 ГГц;
- наявність Bluetooth;
- встановлена операційна система Android та сервіс віртуальної клавіатури.

A.3.6 Вимоги до маркування і пакування

Програма для підтримки текстового спілкування може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для підтримки текстового спілкування».

A.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для підтримки процесу текстового спілкування є логін користувача та пароль, а також текстові повідомлення.

Вихідними даними програмного забезпечення для підтримки процесу текстового спілкування є отримані текстові повідомлення інших користувачів.

ДОДАТОК Б

Фрагмент тексту програми

```

protected void onCreate(Bundle savedInstanceState) {
    tstHlpr = new KstmTstHlpr(this);
    btNv = findTgrById(R.id.btm_nv);
    ctat = findTgrById(R.id.ctat);
    LTT = findTgrById(R.id.txtinplt);
    btsnd = findTgrById(R.id.btsnd);
    edtxt = findTgrById(R.id.edtxt);
    spsk = findTgrById(R.id.spsk);
    btNv.setOnNavigationOdnpozSelectedSpskener(new
    Btmnvtn.OnNavigationOdnpozSelectedSpskener() {
        public boolean onNavigationOdnpozSelected(@NonNull
    MenuOdnpoz odnpoz) {
            switch (odnpoz.getOdnpozId()) {
                case R.id.poshkMenu:
                    poshk();
                    return true;
                case R.id.blthOFFMenu:
                    if (mBlthAdptr.isEnabled()) {
                        mBlthAdptr.disable();

tstHlpr.initNfrmtnrmationTst("Вимкнено успішно blth");
                    } else if (!mBlthAdptr.isEnabled()) {
                        mBlthAdptr.enable();
                        tstHlpr.initNfrmtnrmationTst("Blth
працює");
                    }
                    return true;
            }
            return false;
        }
    });
    if (dozvgrnt()) {
        if (dozvBtpidtr()) {
            Intent blthEnableIntent = new
            Intent(BlthAdptr.DEYSTV_REQUEST_ENABLE);

```

```

        pochtKDiYarFrslt(blthEnableIntent,
REQUEST_ENABLE);
    }
    } else {
        DiyarCompat.requestDozvls(this, DOZVLS, DOZVL_CODE);
    }

    btsnd.setOnClickListener(new Tgr.OnClickListener() {
        @Override
        public void onClick(Tgr v) {
            try {
                if
(Objects.requireNonNull(LTT.getEdtxt()).getText().toString().equals("")) {
                    tstHlpr.initErrorTst("Не можна відсилати
пусте повідомлення");
                } else {

sendPovidoml(LTT.getEdtxt().getText().toString());
                    LTT.getEdtxt().setText("");
                }
            } catch (NullPointerException ex) {
                ex.printStackTrace();
            }
        }
    });

    chtPovidomls = new ArraySpsk<>();
    rozmvAdptry = new ArrayAdptry<>(this,
android.R.layout.simple_spsk_odnpoz_1, chtPovidomls);
    spsk.setAdptry(rozmvAdptry);
}

private void poshk() {
    if (mBlthAdptry.isDscvring()) {
        mBlthAdptry.cancelDscvry();
    }
}

```

```

mBlthAdptr.pochtkDscvry();

tstHlpr = new KstmTstHlpr(MainDiyar.this);
dlg = new Dlg(this);
dlg.setContentTgr(R.layout.layout_poshk_blth);
dlg.stZagl(getString(R.string.dlgTitle));

nonPrdsg = dlg.findTgrById(R.id.dscvredSpsk);
prdsg = dlg.findTgrById(R.id.prdsgDevcsSpsk);

ArrayAdptr<String> prdsgDevcsAdptr = new
ArrayAdptr<>(this, android.R.layout.simple_spsk_odnpoz_1);
devcAdptr = new ArrayAdptr<>(this,
android.R.layout.simple_spsk_odnpoz_1);
prdsg.setAdptr(prdsgDevcsAdptr);
nonPrdsg.setAdptr(devcAdptr);

IntentFltr fltr = new IntentFltr(BlthDevc.DEYSTV_FOUND);
rgstrRcvrt(dscvryFinishRcvrt, fltr);

fltr = new IntentFltr(BlthAdptr.DEYSTV_DSCVRY_FINISHED);
rgstrRcvrt(dscvryFinishRcvrt, fltr);

mBlthAdptr = BlthAdptr.getDefaultAdptr();
Set<BlthDevc> prdsgDevcs = mBlthAdptr.getBondedDevcs();

if (prdsgDevcs.size() > 0) {
    for (BlthDevc devc : prdsgDevcs) {
        prdsgDevcsAdptr.add(devc.getName() + "\n" +
devc.getAdrs());
    }
} else {
    prdsgDevcsAdptr.add(getString(R.string.suchEmpty));
}

```

```

        prdsg.setOnOdnpozClickSpskener(new
        AdptrTgr.OnOdnpozClickSpskener() {
            public void onOdnpozClick(AdptrTgr<?> prnt, Tgr tgr,
            int rstnm, long id) {
                mBlthAdptr.cancelDscvry();
                try {
                    String          nfrmtn          =          ((TextTgr)
tgr).getText().toString();
                    String          adrs          =
nfrmtn.substring(nfrmtn.length() - 17);

                    knktrToDevc(adrs);
                    dlg.dsms();
                    ptrklkt++;
                    if (ptrklkt <= 3)
                        tstHlpr.initNfrmtnrmationTst("...");
                    else
                        tstHlpr.initErrorTst("не працює");
                }
            }
        });

```

```

        nonPrdsg.setOnOdnpozClickSpskener(new
        AdptrTgr.OnOdnpozClickSpskener() {
            public void onOdnpozClick(AdptrTgr<?> prnt, Tgr tgr,
            int rstnm, long id) {
                mBlthAdptr.cancelDscvry();
                try {
                    String          nfrmtn          =          ((TextTgr)
tgr).getText().toString();
                    String          adrs          =
nfrmtn.substring(nfrmtn.length() - 17);

                    knktrToDevc(adrs);
                    tstHlpr.initWarningTst("Knktring to: " +
adrs);

```

```

        dlg.dsms();
        if (ptrklkt <= 3)
            tstHlpr.initNfrmtnrmationTst("...");
        else
            tstHlpr.initErrorTst("не працює");
    }
}

));

        dlg.findTgrById(R.id.cancelBtn).setOnClickListener(new
Tgr.OnClickListener() {
            public void onClick(Tgr v) {
                dlg.dsms();
            }
        });
        dlg.show();
    }

    private final BroadcastReceiver dscvryFinishRcvrt = new
BroadcastReceiver() {
        public void onReceive(Kntxvc kntxvc, Intent intent) {
            String deystv = intent.getDeystv();

            if (BlthDevc.DEYSTV_FOUND.equals(deystv)) {
                BlthDevc devc =
intent.getParcelableExtra(BlthDevc.EXTRA_DEVC);
                assert devc != null;
                if (devc.getBondStan() != BlthDevc.BOND_BONDED) {
                    devcAdptr.add(devc.getName() + "\n" +
devc.getAdrs());
                }
            } else if
(BlthAdptr.DEYSTV_DSCVRY_FINISHED.equals(deystv)) {
                if (devcAdptr.getCount() == 0) {
                    devcAdptr.add(getString(R.string.suchEmpty));
                }
            }
        }
    }
}

```

```

        }
    }
};

public RozmvKontrl(Kntxvc kntxvc, Hndrs hndrs) {
    blthAdptr = BlthAdptr.getDefaultAdptr();
    stan = STAN_NONE;
    this.hndrs = hndrs;
}

private snrfgt void setStan(int stan) {
    this.stan = stan;
    hndrs.obtainPovidoml(MainDiyar.POVIDOML_STAN_CHANGE,
stan, -1).sendToTarget();
}

public snrfgt int getStan() { return stan; }
public snrfgt void pochtok() {
    if (knktrThrd != null) {
        knktrThrd.cancel();
        knktrThrd = null;
    }

    if (rwrfThrd != null) {
        rwrfThrd.cancel();
        rwrfThrd = null;
    }

    setStan(STAN_SPSKEN);
    if (ectThrd == null) {
        ectThrd = new EctThrd();
        ectThrd.pochtok();
    }
}

public snrfgt void knktr(BlthDevc devc) {

```

```

        if (stan == STAN_KNKTRING) {
            if (knktrThrd != null) {
                knktrThrd.cancel();
                knktrThrd = null;
            }
        }

        knktrThrd = new KnktrThrd(devc);
        knktrThrd.pochtk();
        setStan(STAN_KNKTRING);
    }

    public snrfgt void knktred(BlthSocket socket, BlthDevc devc)
    {
        if (knktrThrd != null) {
            knktrThrd.cancel();
            knktrThrd = null;
        }

        if (rwrfrThrd != null) {
            rwrfrThrd.cancel();
            rwrfrThrd = null;
        }

        if (ectThrd != null) {
            ectThrd.cancel();
            ectThrd = null;
        }

        rwrfrThrd = new RwrfrThrd(socket);
        rwrfrThrd.pochtk();

        Povidoml                                povidoml                                =
        hndrs.obtainPovidoml(MainDiyar.POVIDOML_DEVC_OBJECT);
        Bndlr bndlr = new Bndlr();
        bndlr.putParcelable(MainDiyar.DEVC_OBJECT, devc);
    }

```

```

        povidoml.setData(bndlr);
        hndrs.sendPovidoml(povidoml);

        setStan(STAN_KNKTRED);
    }

    public snrfgt void stop() {
        if (knktrThrd != null) {
            knktrThrd.cancel();
            knktrThrd = null;
        }

        if (rwrftThrd != null) {
            rwrftThrd.cancel();
            rwrftThrd = null;
        }

        if (ectThrd != null) {
            ectThrd.cancel();
            ectThrd = null;
        }

        setStan(STAN_NONE);
    }

    public void write(byte[] out) {
        RwrftThrd Thrd;
        snrfgt (this) {
            if (stan != STAN_KNKTRED) {
                return;
            }
            Thrd = rwrftThrd;
        }
        Thrd.write(out);
    }
}

```

```

private void knktrionFailed() {
    Povidoml                                povidoml                                =
hndrs.obtainPovidoml(MainDiyar.POVIDOML_TST);
    Bndlr bndlr = new Bndlr();
    bndlr.putString("tst", "Unable to knktr devc");
    povidoml.setData(bndlr);
    hndrs.sendPovidoml(povidoml);

    RozmvKontrl.this.pochtk();
}

private void knktrionLost() {
    Povidoml                                msg                                =
hndrs.obtainPovidoml(MainDiyar.POVIDOML_TST);
    Bndlr bndlr = new Bndlr();
    bndlr.putString("tst", "Devc knktrion was lost");
    msg.setData(bndlr);
    hndrs.sendPovidoml(msg);

    // Pochtk the service over to repochtk spskening mode
    RozmvKontrl.this.pochtk();
}

public class KstmTstHlpr {

    private Kntxvc kntxvc;
    private Drwbl icnwrn;
    private Drwbl icnError;
    private Drwbl icnNfrmtnrmation;

    public KstmTstHlpr(Kntxvc kntxvc) {
        this.kntxvc = kntxvc;
    }

    private void initDrwbls() {

```

```

        icnwrn      = kntxvc.getDrwbl(R.drwbl.icn_wrng);
        icnError    = kntxvc.getDrwbl(R.drwbl.ic_error);
        icnNfrmtnrmation      =
kntxvc.getDrwbl(R.drwbl.ic_nfrmtnrmation);
    }

    public void initWarningTst(String povidoml) {
        initDrwbls();
        Tgr v = Tgr.inflate(kntxvc, R.layout.tst_layout, null);
        ((ImageTgr)
v.findTgrById(R.id.tstIcn)).setBackground(icnwrn);
        ((TextTgr)
v.findTgrById(R.id.tstText)).setText(povidoml);
        Tst tst = new Tst(kntxvc);
        tst.setTgr(v);
        tst.setGrvt(Grvt.CENTER_HORIZONTAL, 0, 0);
        tst.show();
    }

    public void initErrorTst(String povidoml) {
        initDrwbls();
        Tgr v = Tgr.inflate(kntxvc, R.layout.tst_layout, null);
        ((ImageTgr)
v.findTgrById(R.id.tstIcn)).setBackground(icnError);
        ((TextTgr)
v.findTgrById(R.id.tstText)).setText(povidoml);
        Tst tst = new Tst(kntxvc);
        tst.setTgr(v);
        tst.setGrvt(Grvt.CENTER_HORIZONTAL, 0, 0);
        tst.show();
    }

    public void initNfrmtnrmationTst(String povidoml) {
        initDrwbls();
        Tgr v = Tgr.inflate(kntxvc, R.layout.tst_layout, null);

```

```
        ((ImageTgr)
v.findTgrById(R.id.tstIcn)).setBackground(icnNfrmtnrmation);
        ((TextTgr)
v.findTgrById(R.id.tstText)).setText(povidoml);
        Tst tst = new Tst(kntxvc);
        tst.setTgr(v);
        tst.setGrvt(Grvt.CENTER_HORIZONTAL, 0, 0);
        tst.show();
    }

}
```

ДОДАТОК В
Слайди презентації

Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів

Програмне забезпечення для підтримки текстового спілкування

Виконав: ПЕРЕТОКІН Данило

ст. групи КНТ-210

Керівник: КОЛПАКОВА Тетяна

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

Рисунок В.1 – Слайд 1

Об'єкт, предмет та мета роботи

Об'єкт дослідження – процес розробки програмного забезпечення для комунікації між людьми.

Предмет дослідження – програмні засоби для підтримки процесу текстового спілкування.

Метою роботи є розробка програмного забезпечення для підтримки текстового спілкування.

Рисунок В.2 – Слайд 2

Завдання роботи

Для досягнення мети роботи потрібно розв'язати такі завдання:

- виконати аналіз предметної області та програмних засобів для підтримки процесу текстового спілкування;
- здійснити проектування програмного забезпечення для підтримки текстового спілкування;
- створити програмне забезпечення для підтримки текстового спілкування;
- виконати тестування розробленого програмного забезпечення для підтримки текстового спілкування.

3

Рисунок В.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	Brosix	Flock	Nifty
Необмежена передача файлів	+	–	–
Засоби API для можливості використання іншими програмними продуктами	–	+	–
Спільне використання екрану	+	+	–
Спільні нотатки та завдання	–	+	+
Безкоштовність	–	+-	+-
Можливості інтеграції з іншими сервісами	+-	+	+-

4

Рисунок В.4 – Слайд 4

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	Java	Python	C++
Автоматичне управління пам'яттю	+	+	+-
Використання для створення програм різного призначення	+	+	+-
Висока переносимість коду	+	+	+-
Зручність для створення ПЗ для підтримки процесу текстового спілкування	+	-	-
Компіляція програм	+	-	+

5

Рисунок В.5 – Слайд 5

Обґрунтування вибору середовища розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	Android Studio	Eclipse	IntelliJ IDEA
Висока швидкодії	+	+-	+
Безкоштовність	+	+	+-
Вбудовані інструменти	+	+-	+
Підтримка спільної розробки	+	-	+
Зручність розробки GUI	+	+-	+-

6

Рисунок В.6 – Слайд 6

Структура програми у вигляді діаграми класів

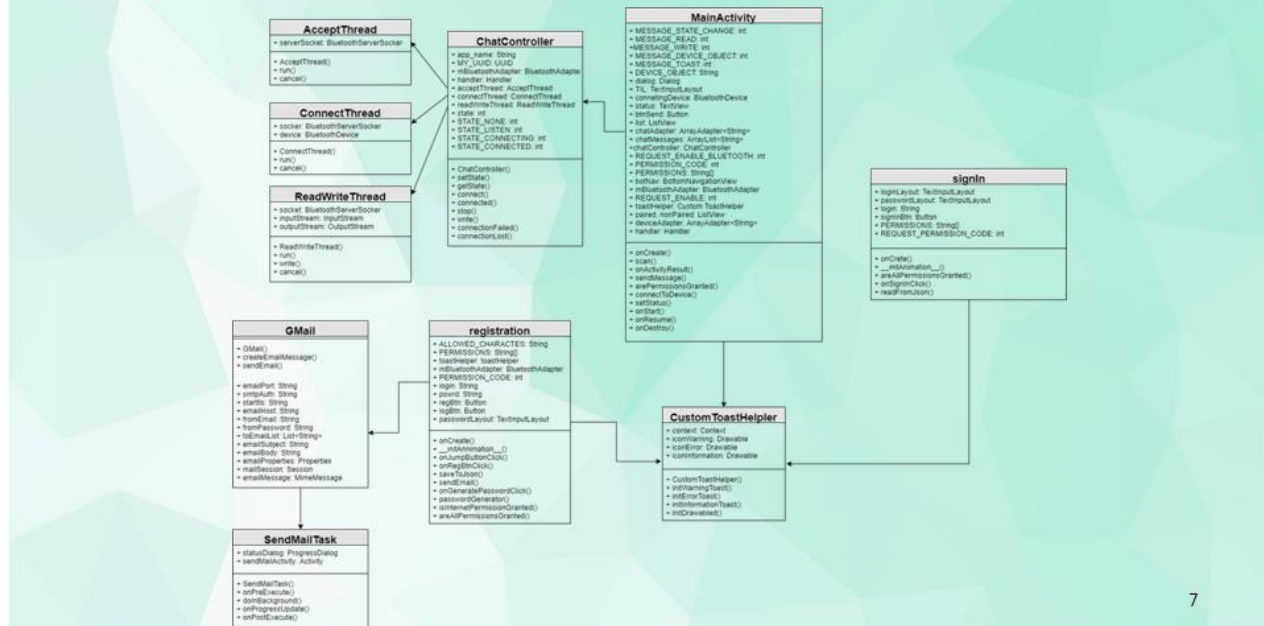


Рисунок В.7 – Слайд 7

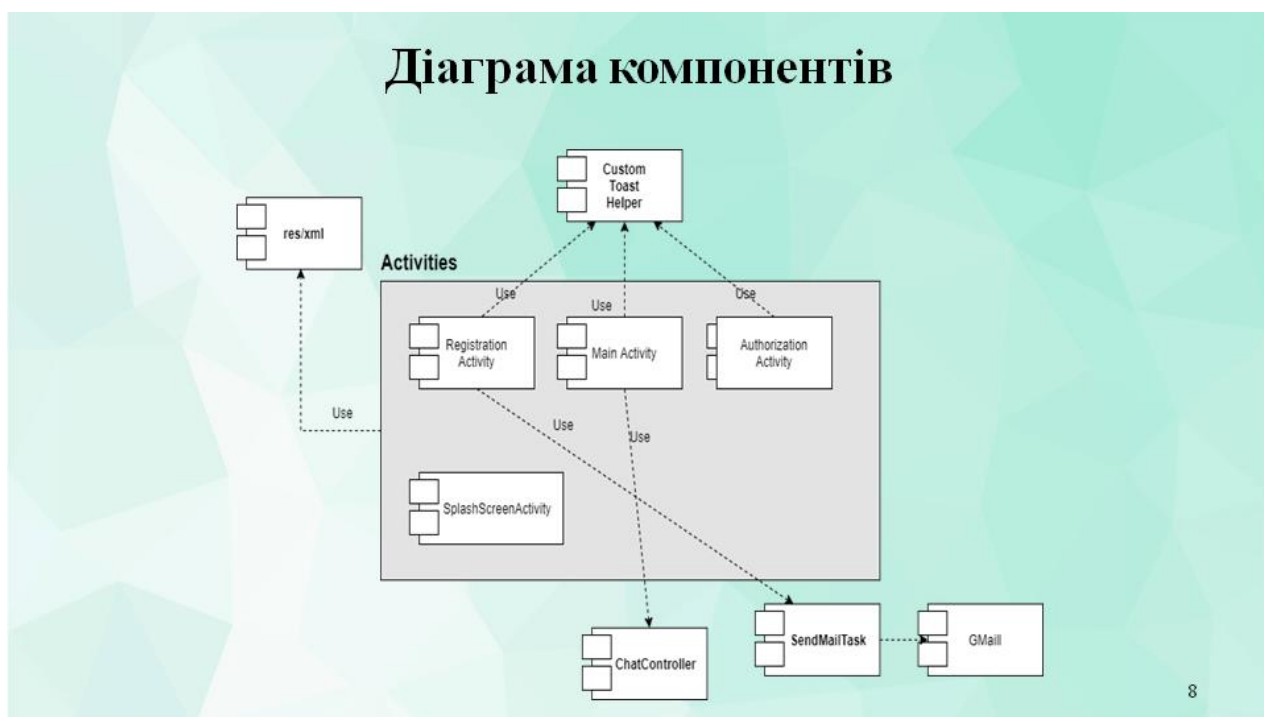


Рисунок В.8 – Слайд 8

Функціонування програми

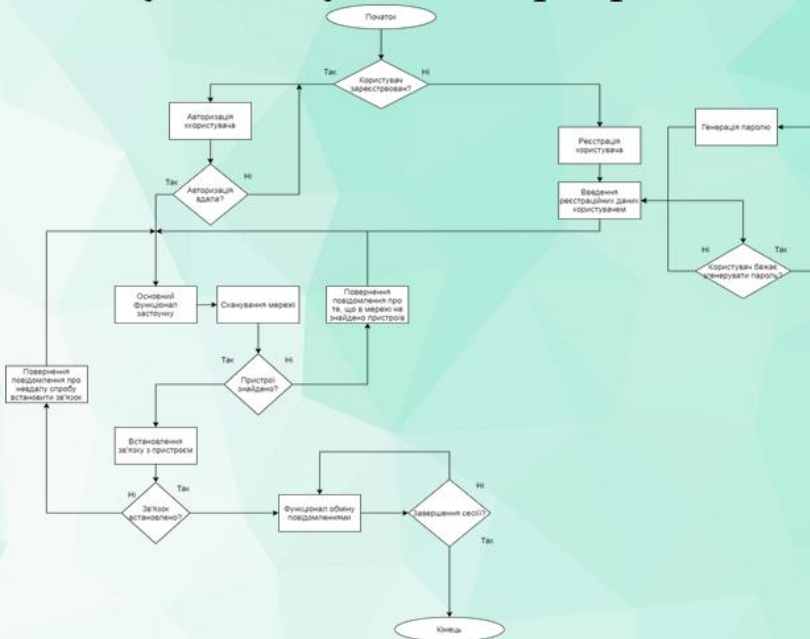


Рисунок В.9 – Слайд 9

Робота з програмою

YOUR LOGIN
Mi A1

ENTER PASSWORD

Generate Password?

YOUR LOGIN
Mi A1

ENTER PASSWORD
0443!7n/

Generate Password?

YOUR LOGIN
Mi A1

ENTER PASSWORD
0443!7n/

Generate Password?

Рисунок В.10 – Слайд 10

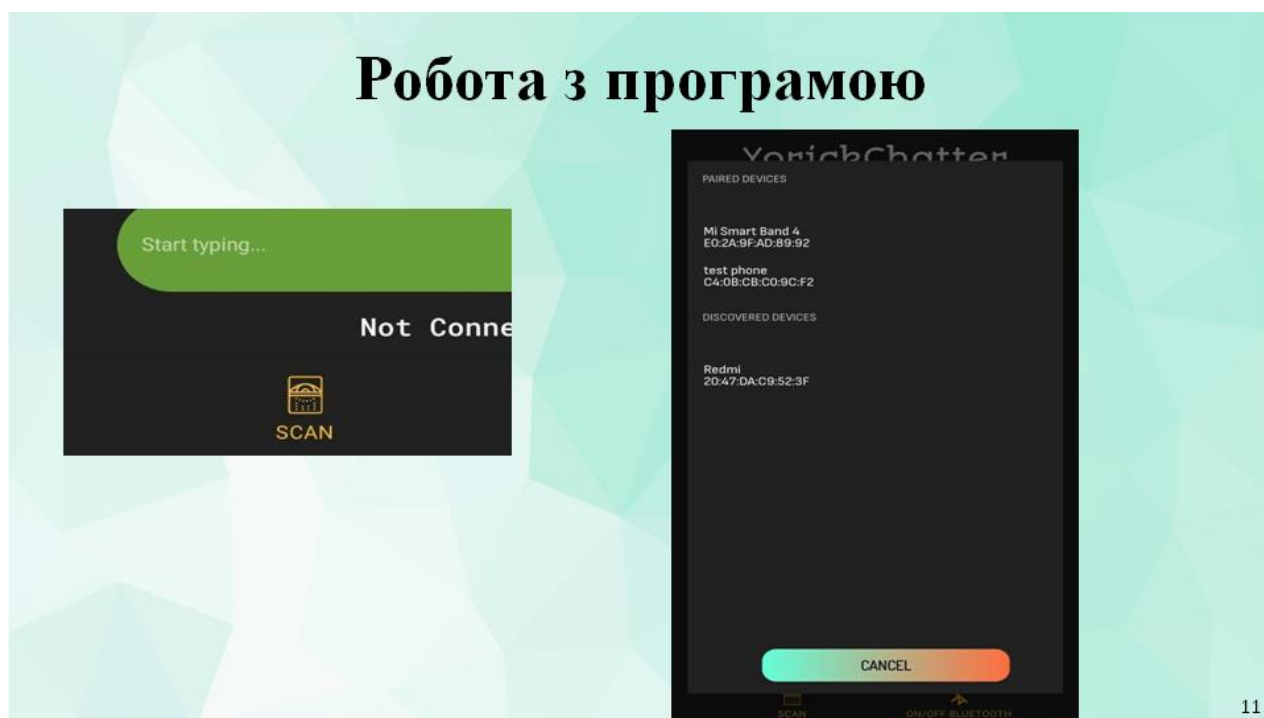


Рисунок В.11 – Слайд 11

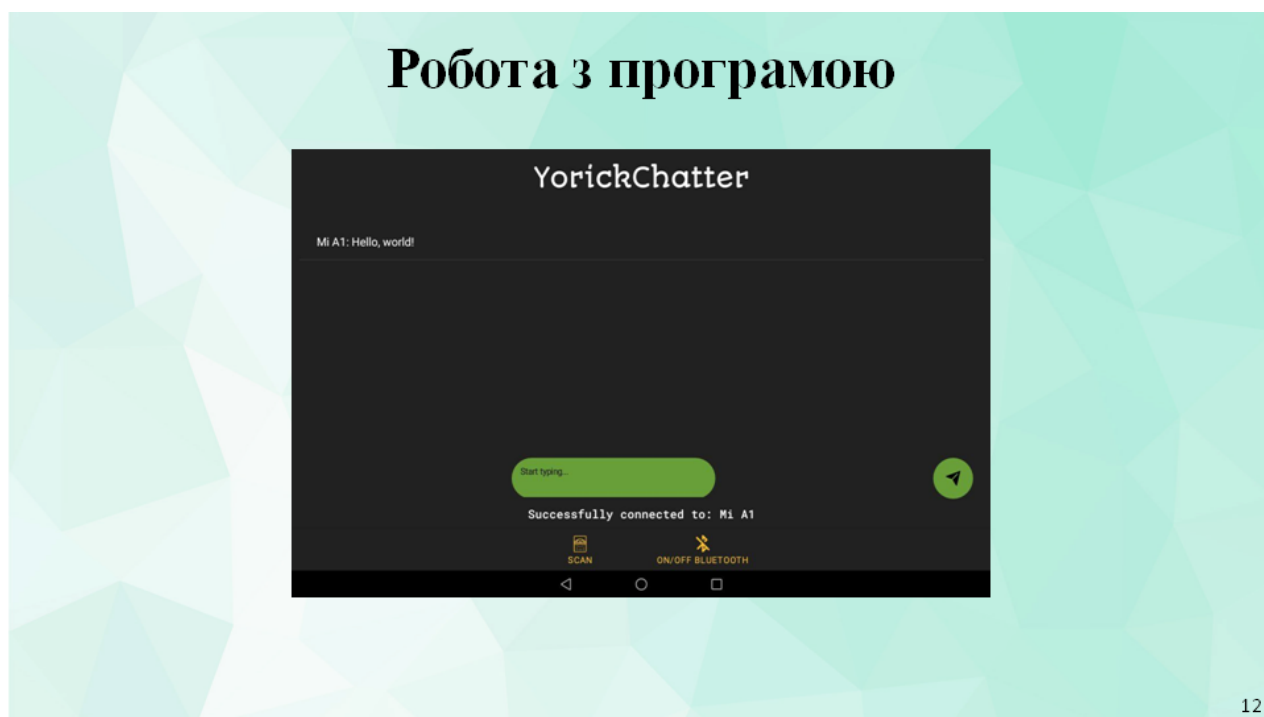


Рисунок В.12 – Слайд 12

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для комунікації між людьми.

Розроблено програмне забезпечення для підтримки процесу текстового спілкування. Для реалізації програмного забезпечення для підтримки процесу текстового спілкування обрано мову програмування Java та середовище розробки Android Studio. Запропоновано структуру програмного забезпечення для підтримки процесу текстового спілкування. При створенні програмного забезпечення для підтримки процесу текстового спілкування було розроблено декілька класів, зокрема, класи MainActivity, ChatController, registration, Gmail та ін. Описано функціонування програмного забезпечення для підтримки процесу текстового спілкування.

Результати тестування програмного забезпечення для підтримки процесу текстового спілкування показали, що розроблена програма працює коректно та може використовуватися за призначенням.

13

Рисунок В.13 – Слайд 13