

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Інститут (факультет) Комп'ютерні та інформаційні технології
Кафедра Обчислювальної техніки та програмування
Спеціальність 123 Комп'ютерна інженерія
Спеціалізація 123.01 Комп'ютерні системи та мережі
Освітня програма Сучасне програмування, мобільні пристрої та комп'ютерні ігри

До захисту допускаю

Завідувач кафедри

Семенов С.Г.

(прізвище та ініціали)

« »

(підпис, дата)

2021 р.

ДИПЛОМНА РОБОТА

першого (бакалаврського) рівня вищої освіти

Тема роботи Тестування захисту інформації сучасних Android-смартфонів

Шифр роботи КІТ-117Б.14

(група, номер теми за наказом)

Виконавець Серебрянський Віталій Сергійович

(прізвище, ім'я, по-батькові)

Керівник старший викладач Межерицький Сергій Геннадійович

(посада, прізвище, ім'я, по-батькові)

Харків 2021

[illegible]

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Інститут (факультет) Комп'ютерних та інформаційних технологій

Кафедра Обчислювальної техніки та програмування

Спеціальність 123 Комп'ютерна інженерія

Спеціалізація 123.01 Комп'ютерні системи та мережі

Освітня програма Сучасне програмування, мобільні пристрої та комп'ютерні ігри

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломної роботи

першого (бакалаврського) рівня вищої освіти

на тему Тестування захисту інформації сучасних Android-смартфонів

Виконав студент 4 курсу, групи KIT-1176

Серебрянський В.С.

(підпис, прізвище та ініціали)

Керівник Межерицький С.Г.

(підпис, прізвище та ініціали)

Рецензент Зиков І.С.

(підпис, прізвище та ініціали)

Нормоконтроль Межерицький С.Г.

(підпис, прізвище та ініціали)

Харків 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Факультет _____ Комп'ютерні та інформаційні технології
Кафедра _____ Обчислювальна техніка та програмування
Рівень вищої освіти _____ перший (бакалаврський)
Спеціальність _____ 123 Комп'ютерна інженерія
Спеціалізація _____ 123.01 Комп'ютерні системи та мережі
Освітня програма _____ Сучасне програмування, мобільні пристрої та комп'ютерні ігри

ЗАТВЕРДЖУЮ

Завідувач кафедри _____
(підпис)

С.Г. Семенов

«__» _____ 2021 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Серебрянському Віталію Сергійовичу

(прізвище, ім'я, по-батькові)

1 Тема роботи Тестування захисту інформації сучасних Android-смартфонів

Керівник роботи Межерицький Сергій Геннадійович, старший викладач
(прізвище, ім'я, по-батькові, науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від «16» лютого 2021 року № 245 СТ

2 Строк подання студентом роботи _____ 16.06.21 р.

3 Вихідні дані до роботи: Операційна система – Windows 10, середовище розробки – Android Studio, мова програмування — Java

4 Перелік питань, які потрібно розробити у пояснювальній записці

Аналітичний огляд, обґрунтування обраних технологій, розробка Android-програми, тестування програми, техніко-економічне обґрунтування розробки, охорона праці та навколишнього середовища

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

Презентація – 10-15 слайдів

6 Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Техніко-економічне обґрунтування	Погорелов С.М., професор		
Охорона праці і навк. середовища	Євтушенко Н.С., доцент		
Перевірка на плагіат та здача до репозиторію	Гайдарова С.С.		

7 Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

Номер етапу	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Обґрунтування теми, постановка проблем і завдань		
2	Аналітичний огляд джерел		
3	Економічне обґрунтування та підготовка висновків		
4	Виконання завдання з охорони праці і навкол. середовища		
5	Розробка, налагодження програм, тестування		
6	Підготовка і виконання пояснювальної записки		
7	Складання відомості документів оформлення ПЗ		
8	Виконання презентації, доповіді		
9	Подання ДР на відгук та внутрішню рецензію		
10	Подання ДР на допуск до захисту		
11	Захист ДР		

Студент

(підпис)

Серебрянський В.С.

(прізвище та ініціали)

Керівник роботи

(підпис)

Межерицький С. Г.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка дипломної роботи містить 65 с., 23 рис., 12 табл., 15 джерел.

Ключові слова: ПРОГРАМНИЙ ПРОДУКТ, БАЗА ДАНИХ, ЗАХИСТ ІНФОРМАЦІЇ, ТЕСТУВАННЯ ЗАХИСТУ, ANDROID ДОДАТОК.

Під час розробки проекту були проведені аналітичні огляди існуючих Android систем для виявлення слабких місць в захисті. У результаті роботи було розроблено Android-додаток для захисту, передачі та зберігання даних.

РЕФЕРАТ

Пояснительная записка дипломной работы содержит 65 с., 23 рис., 12 табл., 15 источников.

Ключевые слова: ПРОГРАММНЫЙ ПРОДУКТ, БАЗА ДАННЫХ, ЗАЩИТА ИНФОРМАЦИИ, ТЕСТИРОВАНИЕ ЗАЩИТЫ, ANDROID ПРИЛОЖЕНИЕ.

Во время разработки проекта были проведены аналитические осмотры существующих Android систем для выявления слабых мест в защите. В результате работы было разработано Android-приложение для защиты, передачи и хранения данных.

REFERAT

The explanatory note of the thesis contains 65 p., 23 fig., 12 tab., 15 sources.

Keywords: SOFTWARE PRODUCT, DATABASE, INFORMATION PROTECTION, PROTECTION TESTING, ANDROID APP.

During the development of the project, analytical examinations of existing Android systems were carried out to identify security vulnerabilities. As a result of the work, an Android application was developed for protecting, transferring and storing data.

ЗМІСТ

Перелік позначень та скорочень.....	4
Вступ.....	5
Аналітичний огляд захисту інформації сучасних android-смартфонів.....	6
1.1 Захист інформації.....	6
1.2 Відомі вразливості і версії Android.....	7
1.3 Контроль за дозволами додатків.....	10
1.4 Постановка завдання та вимоги до продукту.....	11
2 Обґрунтування вибору прийнятих до розгляду технологій.....	13
2.1 Мова програмування Java.....	13
2.2 Середовище розробки Android Studio.....	15
2.3 База даних.....	16
2.4 Критерії створення продукту.....	18
3 Розробка та тестування захищеної програми.....	20
3.1 Структура проекту.....	20
3.2 Тестування програми.....	31
3.3 Тестування захисту інформації.....	35
3.4 Способи захисту Android.....	39
4 Технічно-економічне обґрунтування.....	41
4.1 Вступ.....	41
4.2 Матеріальні витрати.....	41
4.3 Витрати на дослідження та розробку.....	42
4.4 Витрати на оплату праці.....	43
4.5 Додаткова заробітна плата персоналу.....	44
4.6 Єдиний соціальний внесок.....	44
4.7 Амортизаційні відрахування.....	45
4.8 Витрати на електроенергію.....	45

4.9 Накладні витрати.....	46
4.10 Калькуляція собівартості.....	46
4.11 Економічна ефективність.....	47
4.12 Висновки.....	48
5 Охорона праці та навколишнього середовища.....	49
5.1 Загальні питання охорони праці та навколишнього середовища.....	49
5.2 Структура управління Охорони на підприємстві.....	49
5.4 Характеристика приміщення.....	51
5.5 Виробнича санітарія.....	51
5.5.1 Мікроклімат.....	51
5.5.2 Природне освітлення.....	52
5.5.3 Штучне освітлення.....	52
5.5.4 Шум та вібрація у робочому приміщенні.....	53
5.5.5 Електромагнітне випромінювання.....	53
5.5.6 Електробезпека.....	53
5.6 Пожежна безпека.....	54
5.7 Охорона навколишнього природного середовища.....	55
Висновки.....	56
Список джерел інформації.....	57
Додаток А.....	59
Додаток Б.....	64

ПЕРЕЛІК ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ТЗ – Тестування захисту

САС – Сучасні android-смартфони

ОС – операційна система

ПЗ – програмне забезпечення

БД – база даних

ВСТУП

Особиста інформація людини – має велику цінність для кожного.

На сьогоднішній день, захист цієї інформації є найактуальнішою темою.

Контакти, фотографії, секретні дані, документи, або прихована інформація являє собою інструментом для маніпулювання як самої жертви, так і тих людей які на неї схожі.

У ході проведення дослідження були розглянуті питання про захист інформації таким чином, щоб зменшити ризик попадання її в руки злочинців, які можуть ці дані використовувати в поганих цілях.

Було поставлено наступні задачі:

- провести аналітичний огляд існуючих Android систем для виявлення вразливостей;
- проаналізувати систему яка зберігає інформацію;
- протестувати вразливості які знаходяться на стороні користувача;
- отримати загальний результат та проаналізувати його;

Були замічені такі переваги як потужний захист від сторонніх програм.

Тобто якщо користувач сам не дасть дозволу на встановлення програми, то і програма не буде встановлена. Також з початку користування Android забороняє користувачеві скачувати або встановлювати програми зі сторонніх сайтів або у не провірених виробників. Це говорить про те, що якщо користувач не буде давати дозвіл, то і інформація буде надійно захищена.

Але є один великий недолік. Сама операційна система має відкритий код, що означає, що любий програміст зможе знайти недоліки і скористатися ними. На сьогодні ситуація така, що чим більше людей користуються програмним продуктом, тим більше шанс, що вона буде мало захищена. Наприклад дуже популярна Windows. В інтернеті можна знайти багато її копії і багато із тих копій є небезпечними для користувача. Другий приклад, це система Android. Вважається, що ця система є самою безпечною, але ж все одно на телефони користувачів попадають віруси чи трояни, які просто накопичують інформацію.

Із цього можна зробити висновок, що не програма робить більше захисту, а людина. Все залежить від людини.

1 АНАЛІТИЧНИЙ ОГЛЯД ЗАХИСТУ ІНФОРМАЦІЇ СУЧАСНИХ ANDROID-СМАРТФОНІВ

1.1 Захист інформації

Особиста інформація людини – має велику цінність для кожного.

На сьогоднішній день, захист цієї інформації є найактуальнішою темою.

Контакти, фотографії, секретні дані, документи, або прихована інформація являє собою інструментом для маніпулювання жертви.

Тестування захисту (ТЗ) – це оцінка вразливості програмного забезпечення до різних атак.

Сучасні android-смартфони (САС) – мобільний телефон, з сенсорним екраном, доповнений функціональністю кишенькового персонального комп'ютера.

САС дуже часто є мішенню незаконного проникнення. Під проникненням розуміється широкий діапазон дій: спроби хакерів проникнути в систему із спортивного інтересу, помста розсерджених службовців, злом шахраями для незаконної наживи. Тестування безпеки перевіряє фактичну реакцію захисних механізмів, вбудованих в систему, на проникнення, та має шість основних аспектів безпеки, з яких складається тестування безпеки це:

1. конфіденційність
2. цілісність
3. автентифікація
4. доступність
5. авторизації
6. безвідмовність.

1.2 Відомі вразливості і версії Android

Згідно з рис. 1.1 є мінімум п'ять основних версій Android починаючи з Android 5 Lollipop і закінчуючи актуальним Android Pie в більш-менш рівних частках присутні на ринку.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.2%
4.2.x		17	1.5%
4.3		18	0.5%
4.4	KitKat	19	6.9%
5.0	Lollipop	21	3.0%
5.1		22	11.5%
6.0	Marshmallow	23	16.9%
7.0	Nougat	24	11.4%
7.1		25	7.8%
8.0	Oreo	26	12.9%
8.1		27	15.4%
9	Pie	28	10.4%

Рисунок 1.1 – Основні версії Android

Щомісячні патчі безпеки продовжують незалежно один від одного виходити для Android. Якщо виробнику захочеться, він випустить оновлення.

Або не захоче, тоді не випустить. Чим менше виробник підтримує версію, тим легше її зламати чи знайти вразливості.

Оновлення системи є більш пріоритетним завданням, тому що існує багато помилок, які потрібно виправляти.

В Android ізоляція додатків є головною вразливістю. Додатки можуть втручатися в роботу інших додатків. В першу чергу це різноманітні служби Accessibility, додатки, які можуть виводити інформацію поверх інших додатків, оверлеї, які можуть натискати кнопки, і сторонні клавіатури, які автоматично отримують можливість шпигувати за користувачем. Додатки можуть використовувати відомі уразливості для ескалації привілеїв. Можуть збирати дані про приблизний місцезнаходження користувача.

В Android є способи автентифікації. Найпоширеніший з них економ-варіант біометричної системи Apple Face ID. У багатьох смартфонах з Android присутній режим розблокування по обличчю користувача, який використовує єдину камеру і природне освітлення. Такі системи обманюються в кращому випадку тривимірної моделлю, а в гіршому – звичайної фотографією.

Google збирає величезну кількість інформації про користувачів Android. Її детальність і опрацювання просто фантастичні: можна в деталях побачити, де був і що робив користувач в такий-то час такого-то числа за останні кілька років. Що характерно, Google знає не тільки «де» був користувач в термінах географічних координат, а й «в якому місці» він побував. Готель, ресторан, боулінг, театр, парк – Google бачить координати і співвідносить їх з об'єктами на карті і місцем розташування інших користувачів. Результат – компанії відомо все аж до того, скільки часу користувач просидів за столиком в ресторані в очікуванні першої страви і скільки людей користується послугами закладу в даний момент часу.

Пошукові запити, відкриті веб-сторінки, історія покупок в онлайн-магазинах, квитки на літак і броні готелів – всі ці речі мало того, що доступні компанії, так ще й цілеспрямовано витягуються, аналізуються і зберігаються у виділеному, дистильованому і підготовленому для перегляду вигляді.

Немає особливого сенсу писати про те, які саме дані збирає Google і як компанія це робить. У «хмарі» Google знайдеться все: і історія місця розташування користувача за багато років, і паролі, і історія браузера, і журнали телефонних дзвінків і копії повідомлень, а також пошта, фотографії, тренування і параметри організму, резервні копії пристроїв з Android і файли в Google Drive. Сюди ж входять записи голосових запитів до асистента Google, історія перегляду роликів на YouTube, прослухана через Google Music музика, переглянуті фільми і багато іншого. За єдиним винятком, про

нього – нижче, всі ці дані зберігаються у відкритому вигляді. Неспроста: основний бізнес Google – аналіз даних своїх користувачів і продаж реклами з виключно чітким орієнтуванням. Відповідно, ваші дані і повинні надходити і зберігатися у відкритому вигляді, щоб компанія в будь-який момент могла отримати до них доступ.

У зашифрованому вигляді, ключ шифрування залежить від коду блокування смартфона, Google зберігає резервні копії пристроїв, на яких встановлено Android 9 (або більш свіжа версія). Чому саме резервні копії, а не, скажімо, паролі або параметри життєдіяльності організму? Мабуть, справа в тому, що в резервних копіях Google нічого аналізувати: розташування іконок на робочому столі і дані сторонніх додатків для компанії не представляють інтересу.

Стеження з боку «великого брата» і збір інформації можна обмежити, а вже зібрані дані – видалити, але різноманітність цих даних таке, що самий лише перелік займе кілька сторінок густого тексту. Більш того, час від часу Google без попередження включає збір нових типів. Відключити їх збір і зберігання, не завжди зрозуміло навіть фахівцям.

Для того, щоб витягти з «хмари» Google Account всі дані, а їх там не просто багато, а багато більше, ніж зберігається на одному конкретному пристрої, експерту цілком достатньо логіна і пароля, плюс другого чинника, якщо в облікового запису користувача активована двухфакторна автентифікація. Єдине раніше згадане обмеження – резервні копії Android 9, для доступу до яких потрібно код блокування смартфона, з якого ви зробили резервну копію.

1.3 Контроль за дозволами додатків

Питання гранулярного контролю за дозволами додатків в Android до сих пір не закрито. У «чистому» Android (AOSP) контроль за використанням дозволів з'явився тільки з виходом версії 6.0. У сторонніх прошивках він був і раніше, але працював надзвичайно нестабільно. При цьому контроль далекий від досконалості: заборонити додатком працювати в фоні штатними засобами неможливо, а заборона доступу до місця розташування призводить тільки до того, що додаток буде визначати місце розташування пристрою іншими способами за допомогою використання дозволів, гранулярний контроль яких неможливий навіть в свіжих версіях Android. Нарешті, деякі старі програми просто не підтримують гранулярне управління дозволами, і користувач як і раніше погоджується із запитом на надання цілого пакету дозволів, частина з яких зовсім не нешкідлива. Потрібно відзначити, що за допомогою грамотного налаштування корпоративних політик безпеки деякі очевидні уразливості Android можна закрити.

В силу масштабів, яких досяг феномен фрагментації Android, для платформи в цілому не існує якоїсь загальної архітектури безпеки. Десятки моделей смартфонів з Android, що розійшлися в сотнях тисяч примірників, мали шкідливі компоненти в своїх прошивках незважаючи на жорсткі умови сертифікації Google. Сотні моделей десятків виробників використовують відверто небезпечні способи розблокування.

Ряд виробників пропонує власні надбудови для забезпечення додаткової безпеки. Такі технології справляються зі своїми завданнями, але працюють строго в рамках заданих можливостей. Такий захист не є всебічної, і покриває тільки нижні рівні доступу, ніяк не впливаючи на збір і передачу інформації як вбудованими, так і сторонніми додатками.

1.4 Постановка завдання та вимоги до продукту

У ході проведення дослідження були розглянуті питання про захист інформації таким чином, щоб зменшити ризик попадання її в руки злочинців, які можуть ці дані використовувати в поганих цілях. Було поставлено наступні задачі:

- провести аналітичний огляд існуючих Android систем для виявлення вразливостей;
- проаналізувати систему яка зберігає інформацію;
- протестувати вразливості які знаходяться на стороні користувача;
- отримати загальний результат та проаналізувати його;

На основі цих задач створити програму яка зможе:

- зберігати інформацію;
- захищати інформацію від нападу;
- безпечно передавати дані;
- створюватиме простий інтерфейс;

Програма повинна бути максимально простою та безпечною. Для реалізації цього, необхідно використовувати безпечний код та простий інтерфейс.

Безпечне програмування(БП) – це вид програмування, мета якого забезпечити тривале функціонування певної частини коду програми під впливом непередбачуваних обставин.

Техніка безпечного програмування використовується особливо часто коли потрібно звести до мінімуму використання коду не за призначенням. Це дуже актуальна тема, тому що така вразливість як переповнення буферу – трапляється часто. Програма може мати безпечну архітектуру, базу даних чи простий інтерфейс, але якщо програма написана без безпечного програмування, вона має ризик бути зламаною. Якщо є код, який добре протестований і працює без помилок, то його повторне використання зменшить ймовірність появи помилок. Перед повторним використанням старого коду, бібліотек, API та інших компонент програм потрібно переконатися, що цей код можна використовувати повторно. Цілком можливо, що він може викликати проблеми наслідування.

Проблеми наслідування виникають коли старі конструкції мають працювати з новими вимогами. Особливо коли ці конструкції не розроблялися чи тестувалися для нових вимог.

2 ОБҐРУНТУВАННЯ ВИБОРУ ПРИЙНЯТИХ ДО РОЗГЛЯДУ ТЕХНОЛОГІЙ

2.1 Мова програмування Java

Java – об'єктно-орієнтована мова програмування. Мова програмування Java зберігає в собі декілька основних цілей:

- Синтаксис мови повинен бути «простим, об'єктно-орієнтовним та звичним»;
- Реалізація має бути «безвідмовною та безпечною»;
- Повинна зберегтися «незалежність від архітектури та переносність»;
- Висока продуктивність виконання;
- Мова має бути «інтерпретованою, мати багато потоків, динамічне зв'язуванням модулів»;

Java вже майже два десятиліття входить до трійки найпопулярніших мов, що показано на рис. 2.1. За цей час розроблено рішення практично для будь-яких сфер. Інтернет речей, блокчейн, штучний інтелект, Хмарні обчислення, – Java все це може. У мобільній розробці досі домінує частка Андроїд додатків, написаних на Java. JavaFX дозволить розробляти десктопні програми, а кількість фреймворків для веб-розробки величезне.




Jun 2021	Jun 2020	Change	Programming Language		Ratings	Change
1	1			C	12.54%	-4.65%
2	3	▲		Python	11.84%	+3.48%
3	2	▼		Java	11.54%	-4.56%
4	4			C++	7.36%	+1.41%
5	5			C#	4.33%	-0.40%

Рисунок 2.1 – Рейтинг Java

Так як в розробці ми мало зустрічаємося з унікальними завданнями, співтовариство допоможе заощадити купу часу. З чим би не зіткнулися, напевно хтось вже вирішував цю задачу. Досить погуглити потрібні бібліотеки. Спадщина спільноти дозволяє розробникам безкоштовно використовувати потужні IDE, менеджери залежностей і сервера. У 2018-му Oracle напружили програмістів заявою, що Oracle JDK стає платною для використання в продакшені. Насправді ж Java залишилася вільною для всіх бажаючих, просто тепер необхідно уважніше ставитися до використовуваного дистрибутива.

Щоб бути впевненим у легальності розробки, переконайтеся, що застосовуєте безкоштовний Oracle OpenJDK. Крім того, добре розвиваються сторонні реалізації. Наприклад, підтримуються спільнотою AdoptOpenJDK або Coretto, створене Amazon. На мові Java написані банківські системи і великі індустріальні проекти. Навіть якщо в недалекому майбутньому популярність мови спаде, Java розробники будуть потрібні багато десятиліть. Java – це ще й популярна віртуальна машина JVM, на якій працюють інші сучасні мови. Наприклад, Scala, Groovy і Kotlin. Вони принесли в Java функціональне програмування і Null безпеку. В ІТ-співтоваристві укорінився міф, що Java набагато повільніше C++. На старті JVM і правда працювала повільно. Сьогодні оптимізації під потреби Ентерпрайза збільшили продуктивність екосистеми Java на порядки, а JIT-компілятор і зовсім скоротив різницю з компіляції мови до нуля. Ці переваги успадкували і мови, що працюють на JVM.

Протягом 11 років після того, як JDK прийняв Oracle, швидкість розвитку залишала бажати кращого. Але починаючи з Java 9, компанія Oracle зобов'язалася випускати по великому оновленню кожні 6 місяців і успішно тримає темп вже три роки. Тому можна сміливо розраховувати на відповідність мови трендам розробки. Java містить великий інструментарій. Не встановлюючи додаткових бібліотек, ви можете: створювати GUI, використовувати багатопоточність, управляти потоками введення і виведення, працювати з мережею, отримувати доступ до баз даних і тому подібне.

2.2 Середовище розробки Android Studio

Перш ніж почати розробляти програми під Android, розглянемо існуючі інструменти, які підходять для цих цілей. Можна виділити необхідні інструменти, без яких розробка мобільних додатків під Android просто неможлива. З іншого боку, існує велика кількість допоміжних систем, в якійсь мірі спрощують процес розробки.

До обов'язкових інструментів відноситься Android SDK – набір засобів програмування, який містить інструменти, необхідні для створення, компіляції і збірки мобільного додатка. У сучасних умовах розробка ПО в більшості випадків ведеться з використанням інтегрованих середовищ розробки (IDE). IDE мають

безперечні переваги: процес компіляції, збирання і запуску програми зазвичай автоматизований, в зв'язку з чим для початківця розробника створити своє перше додаток труднощів не становить. Але щоб займатися розробкою всерйоз, необхідно витратити сили і час на вивчення можливостей самого середовища. Розглянемо IDE, придатні для розробки під Android, звично огляд не претендує на повноту і скоріше за все знайдуться не охоплені їм інструменти але найпопулярніші на кінець 2013 року буде розглянуто.

Android Studio – середовище розробки під Android, заснована на IntelliJ IDEA. Подібно Android IDE, вона надає інтегровані інструменти для розробки і налагодження. Додатково до всіх можливостей, очікуваним від IntelliJ, в Android Studio реалізовані:

- підтримка збірки додатку, заснованої на Gradle;
- специфічний для Android рефакторинг і швидке виправлення дефектів;
- lint інструменти для пошуку проблем з продуктивністю, з юзабіліті, з сумісністю версій та інших;
- можливості ProGuard (утиліта для скорочення, оптимізації і обфускації коду) і підписи додатків;
- засновані на шаблонах майстра для створення загальних Android конструкцій і компонентів;
- WYSIWYG редактор, який працює на багатьох розмірах екранів і дозволів, вікно попереднього перегляду, що показує запущене застосування відразу на декількох пристроях і в реальному часі;
- вбудована підтримка хмарної платформи Google.

2.3 База даних

Для проекту було вирішено використовувати максимально захищену базу даних – Firebase.

Firebase – це платформа для розробки програмного забезпечення, заснована у 2011 році Firebase inc, і придбана Google у 2014 році. Заснована як база даних у режимі реального часу, тепер вона має 18 служб (4 з них зараз у бета-версії) та спеціальні API. Вся платформа – це рішення Backend-service як

для мобільних, так і для веб-додатків, що включає послуги зі створення, тестування та управління додатками.

Рішення Backend-service дозволяють усунути необхідність в управлінні базовими базами даних та отриманні відповідного обладнання. Натомість ви можете підключити їх до свого додатку через спеціальні API для кожної окремої служби. У випадку з Firebase їх існує 7, які охоплюють весь спектр внутрішніх технологій для програми. Список платформ, з якими інтегрується Firebase, включає Android, iOS, Web та Unity.

Але про платформу можна дізнатися набагато більше, оскільки вона включає різні сервіси для роботи з керованою серверною базою. База даних у реальному часі Firebase стала першим продуктом, який з'явився під прапором Firebase, тож це найбільш усталений та стабільний сервіс на всій платформі. База даних реального часу - це по суті хмарне сховище NoSQL, яке можна підключити до програми, щоб забезпечити доступ до даних у реальному часі на різних платформах. Однією з переваг є те, що база даних може працювати в автономному режимі, кешуючи дані в пам'яті пристрою та після повторного підключення до Інтернету, синхронізуючи їх.

Дані зберігаються у форматі JSON і можуть запитуватися користувачами. З точки зору безпеки, база даних Realtime забезпечує доступ до даних на основі дозволів. Це можна зробити за допомогою автентифікації Firebase та надання дозволів за ідентифікацією користувача або правилами безпеки.

Cloud Firestore – це ще одна база даних NoSQL, розміщена в хмарі. На відміну від бази даних Firebase Realtime, Cloud Firestore розроблена для корпоративного використання, що передбачає масштабованість, складні моделі даних та розширені параметри запитів. Консоль Firebase може використовуватися для перегляду даних в обох базах даних. Іншим спільним моментом є те, що існують SDK для роботи з кодом на стороні сервера обох баз даних. На рис. 2.2 показана схема роботи Firebase.

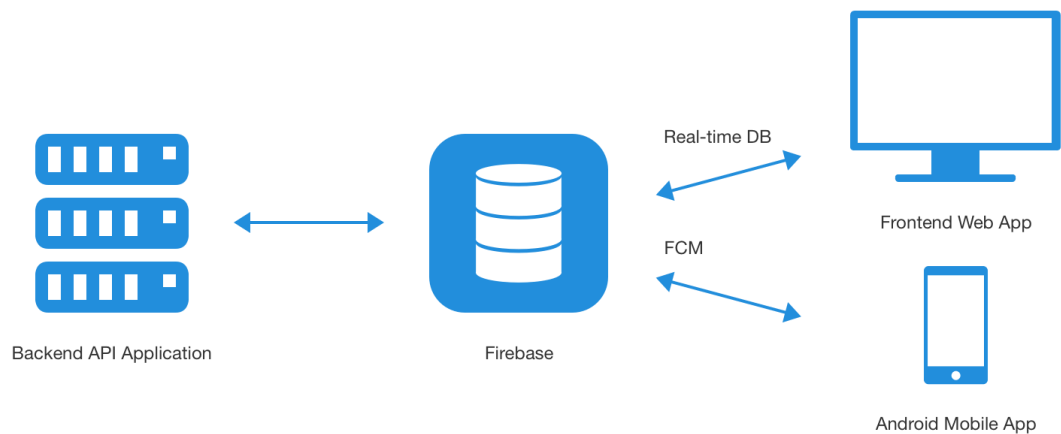


Рисунок 2.2 – Схема роботи Firebase

1.5 Критерії створення продукту

Опираючись на ці критерії, можна створити продукт, який не буде уступати в якості нинішнім безпечним програмам.

Що можна порадити, та якій потрібно вибрати робочий чат:

- Оцініть ризики. Прикиньте список цінностей, найважливіших даних, які можуть опинитися в чаті. Запитайте себе, що загрожує цим цінностям, наприклад, хто може хотіти отримати інформацію з чату?
- Визначте, який функціонал вам потрібен. Чи достатньо текстового чату, або ви хочете також спілкуватися голосом і проводити відеоконференції? Чи потрібен чат з декількома каналами? Чи потрібні розширені функції адміністрування, скажімо, можливість видалити з чату некоректно поводитьься людини?
- Подивіться різні технічні рішення, які мають за потрібне функціоналом. Чи дозволяють вони захистити вашу цінну інформацію на адекватному рівні? Оцініть, які ресурси знадобляться, щоб придбати і впровадити ту чи іншу рішення у

вашій команді. Наприклад, скільки чоловік з вашої команди вже використовують цей чат? Чи знадобиться тренінг? Чи доведеться допомагати колегам встановлювати додатки на мобільні пристрої?

На що потрібно звертати увагу в плані безпеки:

- Будете користуватися готовим сервісом або встановлювати програмне забезпечення самостійно (self-hosted)? Перший варіант передбачає певну довіру до розробника – зате не знадобляться технічні знання. У другому випадку можлива більш тонка настройка параметрів безпеки, а й зусиль доведеться докласти більше.
- Відкритий код. Продукт доступний користувачам не тільки готовим до запуску, але і у вигляді коду, на якому написаний. Кваліфікований програміст може перевірити якість коду і переконатися, що в ньому немає помилок або явних «закладок» – лазівок для доступу до призначених для користувача даних.
- Підтримка шифрування (в тому чи іншому вигляді). В епоху месенджерів наскрізне шифрування поступово стає одним з «золотих стандартів».
- Відсутність неодмінною прив'язки до телефонним номером – крок на шляху до анонімності учасників чату (якщо вона вам важлива).
- Юрисдикція: навряд чи можна радити продукт або сервіс, розроблений в країні з репресивним або авторитарним режимом.
- Популярне (поширене) рішення часто виграє тому, що користувачеві легше знайти кваліфіковану технічну підтримку «на місці» – в своєму регіоні і навіть у своїй організації.
- Важливо, щоб проект активно підтримувався розробником. Кому потрібен чат, повний дірок, які ніхто не поспішає латати?
- Є інформація про незалежний аудит коду продукту? Це добре.
- Якщо продукт або сервіс локалізований, ваші колеги, погано знаючи мову оригіналу, будуть допускати менше помилок під час роботи.

- Добре, якщо чат крос-платформний, тобто його можна використовувати на комп'ютерах і мобільних пристроях під управлінням різних операційних систем.

3 РОЗРОБКА ТА ТЕСТУВАННЯ ЗАХИЩЕНОЇ ПРОГРАМИ

3.1 Структура проекту

На рис. 3.1 показана структура усього проекту, яка буде потім розбиратися по частинам.

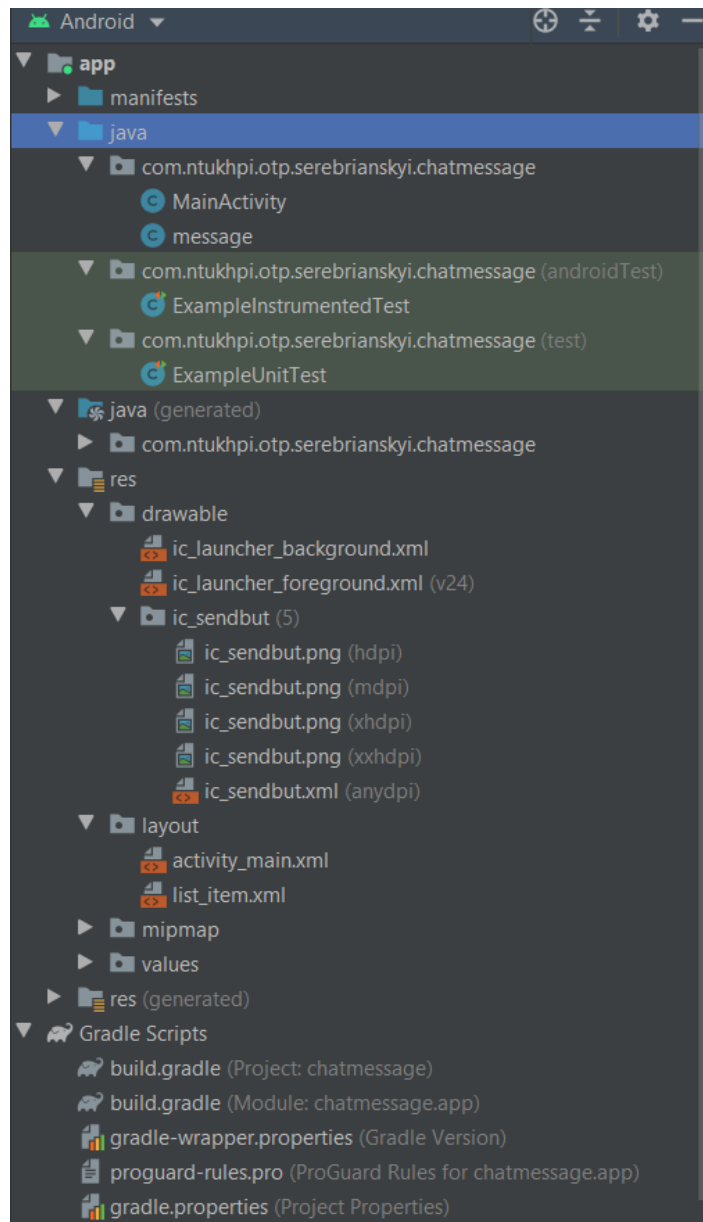


Рисунок 3.1 – Структура проекту

Для початку роботи потрібно підключити бібліотеки з функціями для зручнішого написання коду. А саме головні бібліотеки для Android та бази даних, які показані на рис 3.2


```

import android.content.Intent;
import android.os.Bundle;
import android.os.Message;
import android.view.View;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.text.format.DateFormat;
import com.firebase.ui.auth.AuthUI;
import com.firebase.ui.database.FirebaseListAdapter;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.FirebaseDatabase;

```

Рисунок 3.2 – Бібліотеки для Android

В Android Studio є можливість підключити безкоштовну базу даних Firebase, яка знаходиться на панелі задач інструментів. Перед початком її користування, потрібно мати гугл акаунт, а також зареєструватися на сайті Firebase для того, щоб мати доступ до безкоштовного користування.

Після створення облікового запису, можна підключатися. Саме підключення показано на рис. 3.3

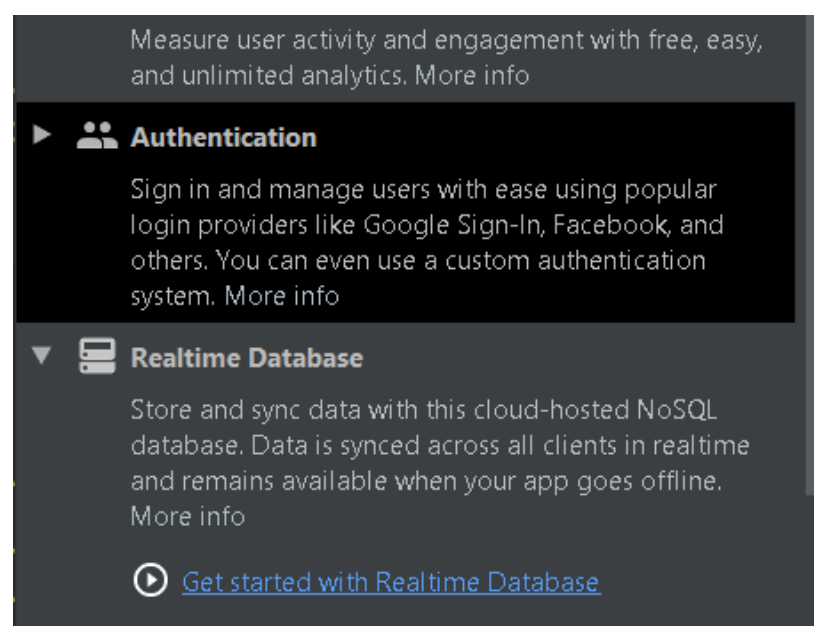


Рисунок 3.3 – Підключення до бази даних

Щоб програма коректно працювала, потрібно синхронізувати бібліотеки і додатки. В цьому випадку все дуже просто. Достатньо натиснути на кнопку і програма сама все підключить як це показано на рис. 3.4

Також потрібно бути дуже обережним з версіями Android та версіями API. Якщо вони не підходять до розробки, то може виникнути помилка. Зазвичай це можна виправити, убравши версії до більш менших чи більш стабільніших.

Якщо користувач не впевнений в версії, на якій він працює, то потрібно читати документації які знаходяться на офіційних сайтах. Також ця інформація частіше трапляється в Android Studio.

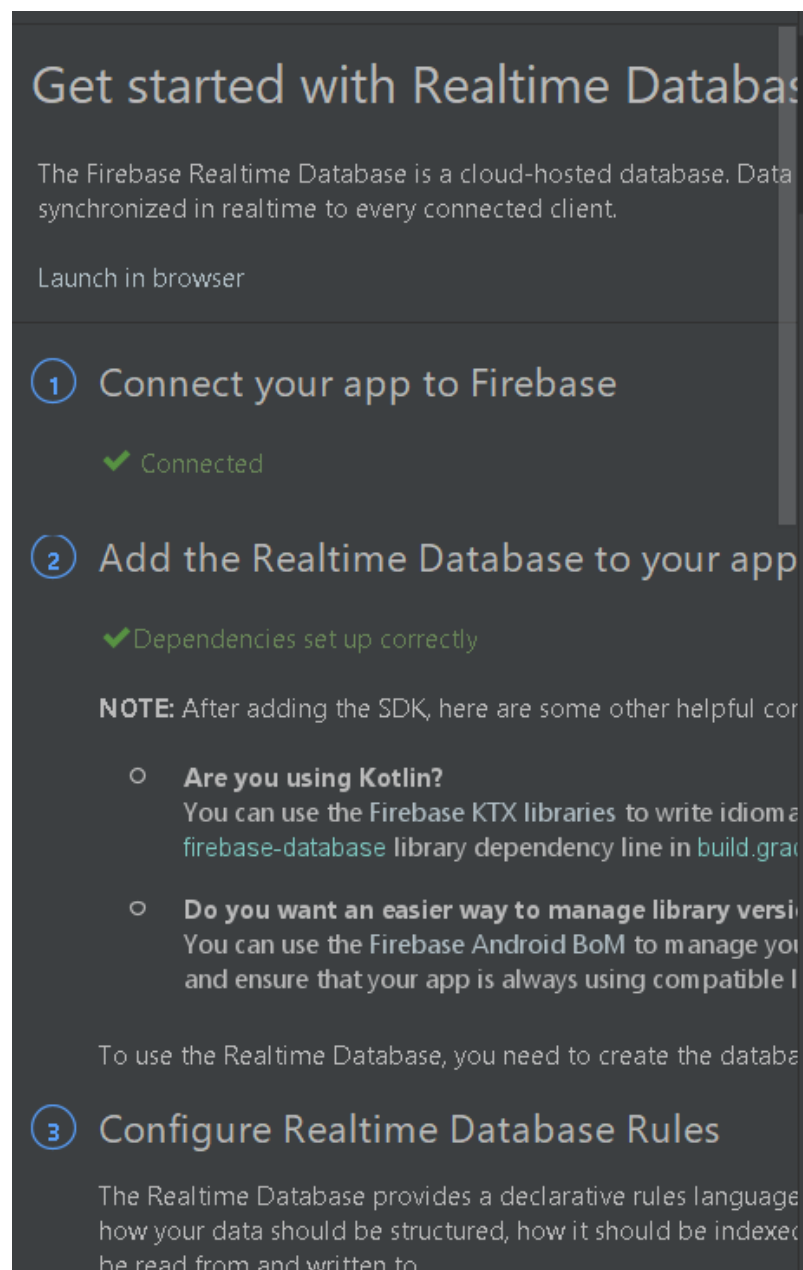


Рисунок 3.4 – Підключення бази даних

Якщо з цим покінчено, то потрібно переходити до основного коду.

Для початку потрібно представити схему чи намалювати каркас програми. Але в моєму випадку, програма буде дуже простою і безпечною, тому сам дизайн програми не має нічого красивого. На рис 3.5 можна побачити основний каркас програми.



Рисунок 3.5 – Каркас програми

Код, який відповідає цьому каркасу знаходиться нижче на рис 3.6

У нього є лист повідомлення, який відображатиме повідомлення від всіх користувачів які знаходяться на серверу. Кожен лист дійде до відправника і буде збережений в базу даних.

Також можна побачити розміри відступів та центр тексту. Це було зроблено для того, щоб не дезорієнтувати користувача. Зовні програма нагадує звичайні чати, тому легко сприймається.

На рис. 3.6 зображено код каркасу.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/btndsend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="3dp"
        android:layout_marginBottom="29dp"
        android:clickable="true"
        android:focusable="true"
        android:src="@drawable/ic_sendbut"
        app:fabSize="mini" />

```

Рисунок 3.6 – Код каркасу

Після дизайну потрібно розібратися з текстовими повідомленнями які буде бачити читач та той, хто надсилає повідомлення. Загальна структура програми є на рис. 3.7

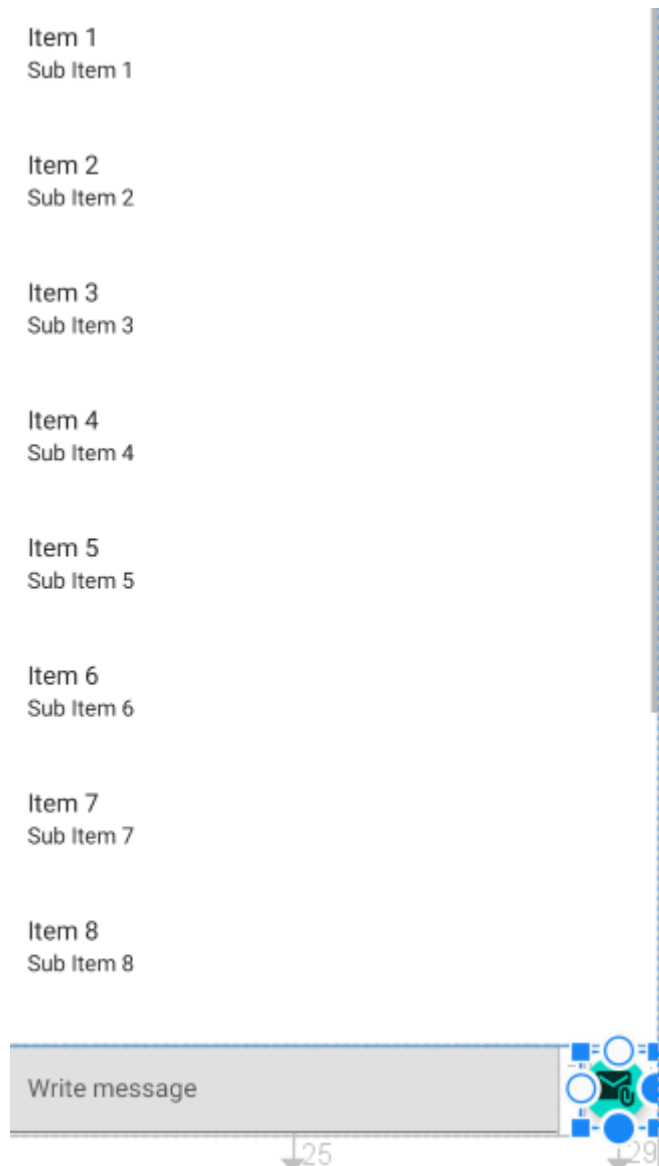


Рисунок 3.7 Загальна структура програми

З зовнішнім видом покінчено. Залишилася частина, яка зазвичай не видна користувачеві. Логіку програми написати не тяжко, тому що документації хватає. Починаючи з класу `MainActivity` ми створюємо дані, які далі будуть потрібні для важливої кнопки відправки текстового повідомлення. В частині функції під назвою `onActivityResult` ми отримуємо результат роботи програми. Якщо він повертає 1, то ми зареєструвалися. Якщо протилежне значення, то виникла помилка і програма повертає текстовий результат з надписом про те, що ми не увійшли. Детальніше про код на рис. 3.8

```

public class MainActivity extends AppCompatActivity {
    private static int SIGN_IN_CODE = 1;
    private RelativeLayout activity_main;
    private FirebaseListAdapter<message> adapter;
    private FloatingActionButton sendButton;
    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if(requestCode==SIGN_IN_CODE) {
            if(resultCode == RESULT_OK) {
                Snackbar.make(activity_main, text: "you are in",Snackbar.LENGTH_LONG).show();
                displayAllMessages();
            } else {
                Snackbar.make(activity_main, text: "you are not !!!in",Snackbar.LENGTH_LONG).show();
            }
        }
    }
}

```

Рисунок 3.8 – Код функції реєстрування

Саме головне – це функція onCreate. Діяльність у системі управляється як стеки дій. Коли починається нова активність, воно зазвичай розміщується у верхній частині поточного стеку і стає запущеним - попереднє завдання завжди залишається під ним у стеку і не вийде знову на передній план, поки нове не вийде. На екрані може бути видно один або кілька стеків активності.

Функція onCreate яка показана на рис 3.9 запускає першу активність. Поки програміст не напише код і не запустить інші активності, будуть виконуватись стандартні цикли активності.

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    activity_main = findViewById(R.id.activity_main);
    sendButton = findViewById(R.id.btndsend);
    sendButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            EditText textField = findViewById(R.id.edit_tex_lay);
            if(textField.getText().toString().isEmpty())
                return;
            FirebaseDatabase.getInstance().getReference().push().setValue(
                new message(FirebaseAuth.getInstance().getCurrentUser().getEmail(),
                    textField.getText().toString())
            );
            textField.setText("");
        }
    });
}

```

Рисунок 3.9 – Функція onCreate

Весь час дії відбувається від першого виклику onCreate до одного остаточного виклику onDestroy. Діяльність виконає всі налаштування "глобального" стану в onCreate і звільнить усі залишені ресурси в onDestroy. Наприклад, якщо у нього в потоці працює потік для завантаження даних з мережі, він може створити цей потік у onCreate, а потім зупинити потік у onDestroy.

Видима тривалість дії відбувається між викликом onStart до відповідного виклику onStop. Протягом цього часу користувач може бачити діяльність на екрані, хоча вона може бути не на передньому плані та взаємодіяти з користувачем. Між цими двома методами ви можете підтримувати ресурси, необхідні для показу активності користувачеві. Наприклад, ви можете зареєструвати BroadcastReceiver в onStart, щоб стежити за змінами, що впливають на ваш інтерфейс, і скасувати його реєстрацію в onStop, коли користувач більше не бачить, що ви відображаєте. Методи onStart та onStop можна викликати кілька разів, оскільки діяльність стає видимою та прихованою для користувача.

Час дії на передньому плані відбувається між викликом onResume до відповідного виклику onPause. Протягом цього часу діяльність є видимою, активною та взаємодіє з користувачем. Діяльність може часто переходити між відновленим та призупиненим станами - наприклад, коли пристрій переходить у режим сну, коли результат діяльності доставляється, коли доставляється новий намір – тому код у цих методах повинен бути досить легким. Вся ця інформація показана в схемі циклу життя активності на рис. 3.10

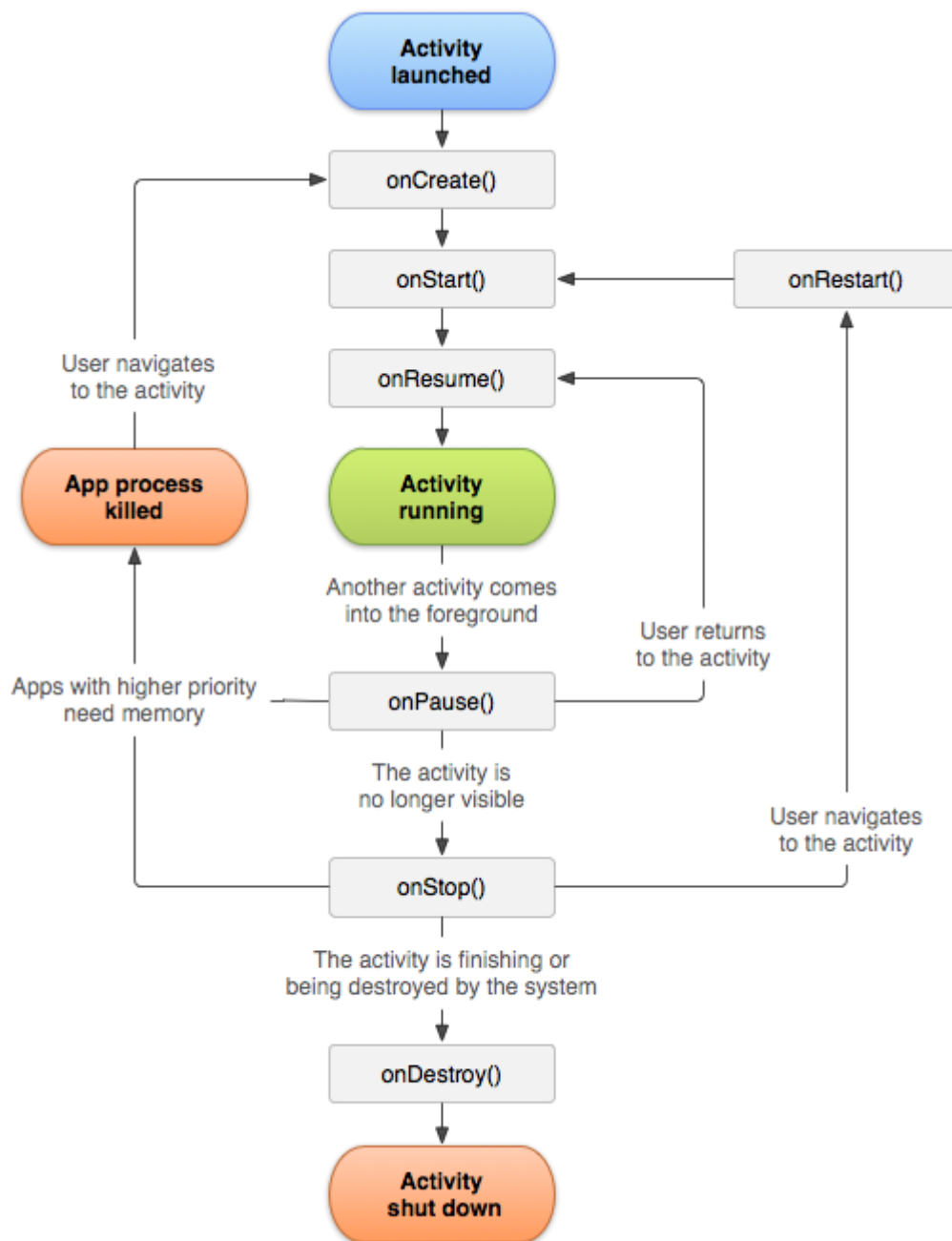


Рисунок 3.10 – Схема життєвого циклу активності

Логіка відправки тексту на екран знаходиться на рис. 3.11.

Функція `displayAllMessages` збирає в себе всю інформацію щодо розташування елементів на екрані. За допомогою їх персонального id номеру, можна напряму взаємодіяти з різними частинами інтерфейсу.


```

    if(FirebaseAuth.getInstance().getCurrentUser()== null)
        startActivityForResult(AuthUI.getInstance().createSignInIntentBuilder().build(),SIGN_IN_CODE);
    else
        Snackbar.make(activity_main, text: "you are in",Snackbar.LENGTH_LONG).show();
    displayAllMessages();
}

private void displayAllMessages() {
    ListView listofmessages = findViewById(R.id.list_messages);
    adapter = new FirebaseListAdapter<message>( activity, this,message.class,R.layout.list_item, FirebaseDatabase.getInstance().getReference()) {
        @Override
        protected void populateView(View v, message model, int position) {
            TextView mess_user, mess_time, mess_text;
            mess_user = v.findViewById(R.id.message_user);
            mess_time = v.findViewById(R.id.message_time);
            mess_text = v.findViewById(R.id.message_text);
            mess_user.setText(model.getUserName());
            mess_text.setText(model.getTextMessage());
            mess_time.setText(DateFormat.format( informat: "dd-mm-yyyy HH:mm:ss",model.getMessagetime()));
        }
    };
    listofmessages.setAdapter(adapter);
}
}

```

Рисунок 3.11 – Функція displayAllMessages

На рисунку 3.12 знаходяться основні забезпечення, без яких база даних не запустилася б.

com.google.android.material.floatingactionbutton.FloatingActionButton
відповідає за кнопку.

com.google.android.material:material:1.3.0 – відповідає за дизайн.

com.google.firebase:firebase-auth:21.0.1 – відповідає за авторизацію

com.google.firebase:firebase-core:19.0.0 – відповідає за базу даних

com.google.firebase:firebase-database:20.0.0 – відповідає за збереження

com.firebaseui:firebase-ui:0.6.2 – відповідає за інтерфейс

```
dependencies {  
    implementation 'com.google.android.material:material:1.3.0'  
    implementation platform('com.google.firebase:firebase-bom:28.1.0')  
    implementation 'androidx.appcompat:appcompat:1.3.0'  
    implementation 'com.google.android.material:material:1.3.0'  
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'  
    implementation 'com.google.firebase:firebase-auth:21.0.1'  
    testImplementation 'junit:junit:4.13.2'  
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'  
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'  
    implementation 'com.google.firebase:firebase-core:19.0.0'  
  
    implementation 'com.google.firebase:firebase-database:20.0.0'  
    implementation 'com.firebaseui:firebase-ui:0.6.2'  
    implementation 'com.google.android.gms:play-services-auth:19.0.0'
```

Рисунок 3.12 Забезпечення для Android

3.2 Тестування програми

Щоб перевірити працездатність програми, потрібно протестувати програму. Для початку відкриємо програму та поспробуємо зайти в неї. На рис. 3.13 знаходиться перша сторінка програми. На ній відбувається автентифікація за поштою. Строчка для вводу даних працює нормально. Знаходиться на своєму місці. Літери відображаються на екрані правильно. Поки що програма працює як потрібно. Автентифікація - процедура перевірки автентичності, наприклад перевірка справжності користувача шляхом порівняння введеного ним пароля з паролем, збереженим в базі даних.

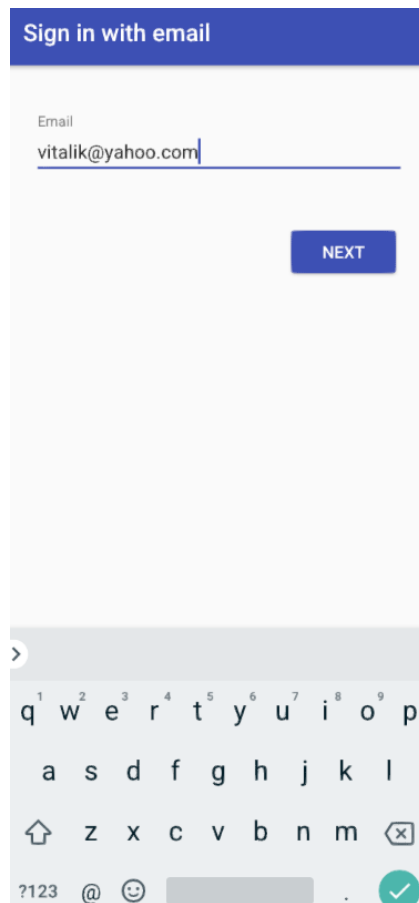


Рисунок 3.13 – Автентифікація користувача

На рис. 3.14 зображено тест відправки текстового повідомлення.

Тест пройшов успішно. Помилки не знайшлося. Тут потрібно відмітити таку особливість, що на першій стадії розробки програми, адресу можна вводити будь-яку. Тобто якщо ми введемо неіснуючу адресу, то це не буде помилкою. На даному етапі такий слабкий захист не є чимось поганим. Якщо використовувати чат локально, то буде достатньо одного імені чи прізвища. Самі ж дані зберігаються на сервері не залежно від його імені. Це зроблено для того, щоб ідентифікувати людину. Це допоможе в тих випадках, якщо трапилось щось важливе і нам потрібно знати час та особу, яка заходила.

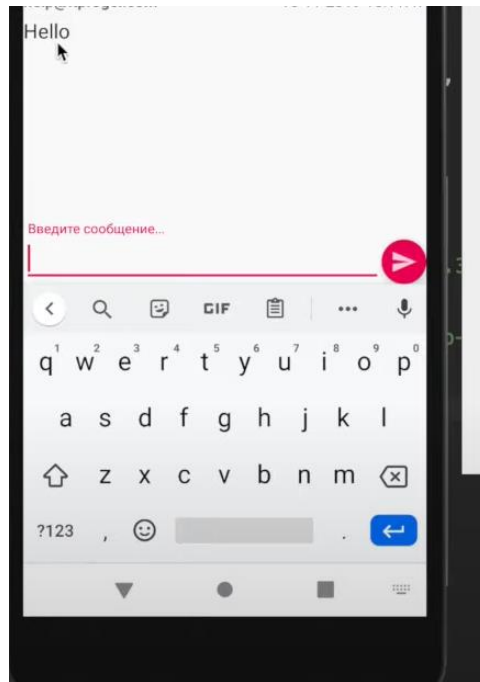


Рисунок 3.14 – Тест відправки повідомлення

Також можна побачити на сервері Firebase в базах даних моє повідомлення, яке прийшло та збереглося. Його можна побачити на рис. 3.15

- Це спряцювало наступним чином:
- Сбрали дані в JSON-формат;
- Упакували їх в спеціальний запит;
- Вбудованими засобами JavaScript відправити цей запит на сервер за потрібною адресою;

Готова інформація відправляється іншим користувачам на телефон. Вони можуть в будь який час прочитати повідомлення.



Рисунок 3.15 – База даних Firebase

На рис 3.16 зображений тест режим в якому показана синхронізація між двома пристроями.

Для забезпечення синхронізації всіх клієнтів найпростіше зробити так, щоб клієнт відправляв серверу оновлення з фіксованим інтервалом. Для прикладу візьмемо інтервал в 30 мілісекунд. Оновлення містить введені користувачем дані, які можуть також містити значення, що вводяться нові дані.

Отримавши дані, що вводяться від всіх користувачів, сервер може перейти до наступного такту з урахуванням цих даних. Ця схема показана на рис. 3.17

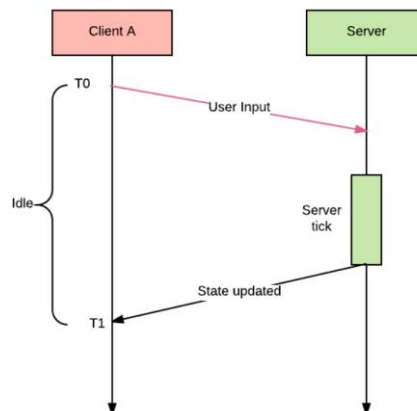


Рисунок 3.17 – Схема синхронізації

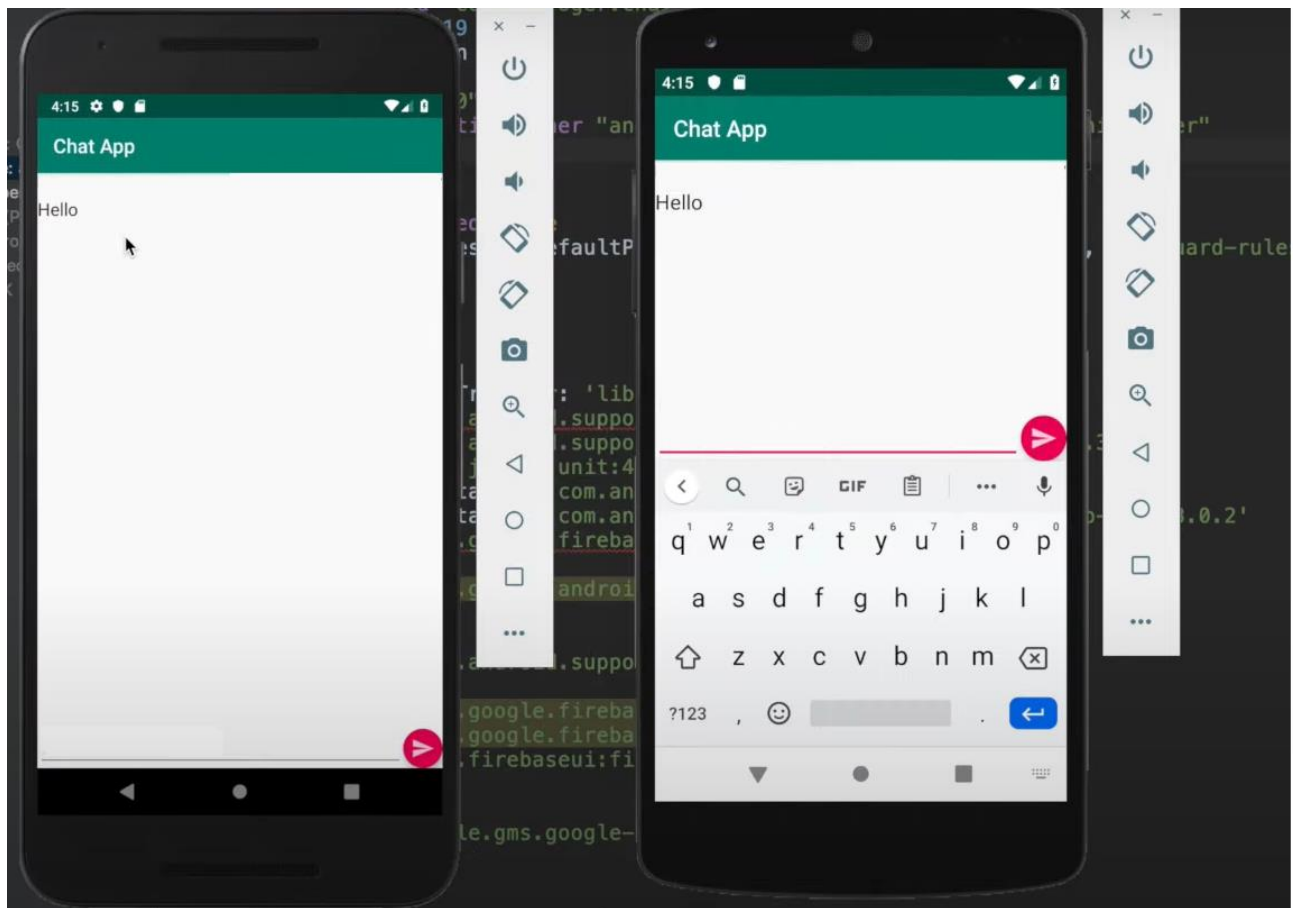


Рисунок 3.16 – Синхронізація між клієнтами

3.3 Тестування захисту інформації

Є така вразливість, як MitM – Man in the Middle атака. Зловмисник може легко здійснити атаку посередника, використовуючи техніку, звану ARP-спуфінга. Будь у вашій мережі Wi-Fi може послати вам підроблений ARP-пакет, через нього ви неусвідомлено будете посилати весь ваш трафік через зловмисника замість маршрутизатора. Після цього зловмисник отримує повний контроль над трафіком і може відстежувати запити, що посилаються в обидві сторони.

Для запобігання таких атак була створена захищена версія протоколу HTTP. Transport Layer Security (TLS) і його попередник, Secure Socket Layer (SSL), є криптографічними протоколами, які забезпечують безпеку передачі даних по мережі. Отже, захищений протокол буде називатися HTTPS. Саме цим протоколом ми і користуємося для захисту наших даних. З точки зору хакера, компрометації будь-якого протоколу зв'язку зводиться до того, щоб знайти слабку ланку серед компонентів: приватність, автентичність і цілісність. SSL використовує асиметричний алгоритм шифрування. У симетричному шифруванні проблема полягає в тому, що для шифрування і дешифрування даних використовується один і той же ключ, такий підхід неприпустимий для

інтернет-протоколів, оскільки зломисник може простежити цей ключ. Асиметричне ж шифрування включає в себе 2 ключа для кожної сторони: відкритий ключ, який використовується для шифрування, і конфіденційний ключ, який використовується для дешифрування даних показаних на рис. 3.18

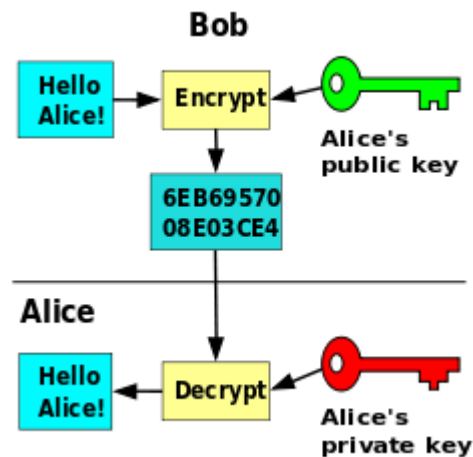


Рисунок 3.18 – Алгоритм шифрування

Як SSL забезпечує три властивості, необхідні для безпечного зв'язку? Оскільки для шифрування даних використовується асиметрична криптографія, SSL забезпечує приватне з'єднання. Це шифрування не так вже й легко зламати і залишитися непоміченим. Сервер підтверджує свою легітимність, посилаючи клієнтові SSL-сертифікат, виданий центром сертифікації - довіреною третьою стороною. Якщо зломисникові якимось чином вдасться роздобути сертифікат, він може створити умови для атаки посередника. Таким чином, він створить 2 з'єднання - з сервером і з жертвою. Сервер в цьому випадку думає, що зломисник - це звичайний клієнт, а у жертви немає можливості ідентифікувати зломисника, оскільки той надав сертифікат, який доводить, що він сервер. Ваші повідомлення доходять і приходять в зашифрованому вигляді, проте проходять по ланцюжку через комп'ютер кіберзлочинця, де у нього є повний контроль.

Сертифікат не обов'язково повинен бути підроблений, якщо у зломисника є можливість скомпрометувати браузер жертви. У цьому випадку він може вставити самостійно підписаний сертифікат, який буде довіреною за замовчуванням. Так і реалізуються більшість атак посередника. У більш складних випадках хакер повинен піти іншим шляхом - підробити сертифікат.

Схема показана на рис 3.19

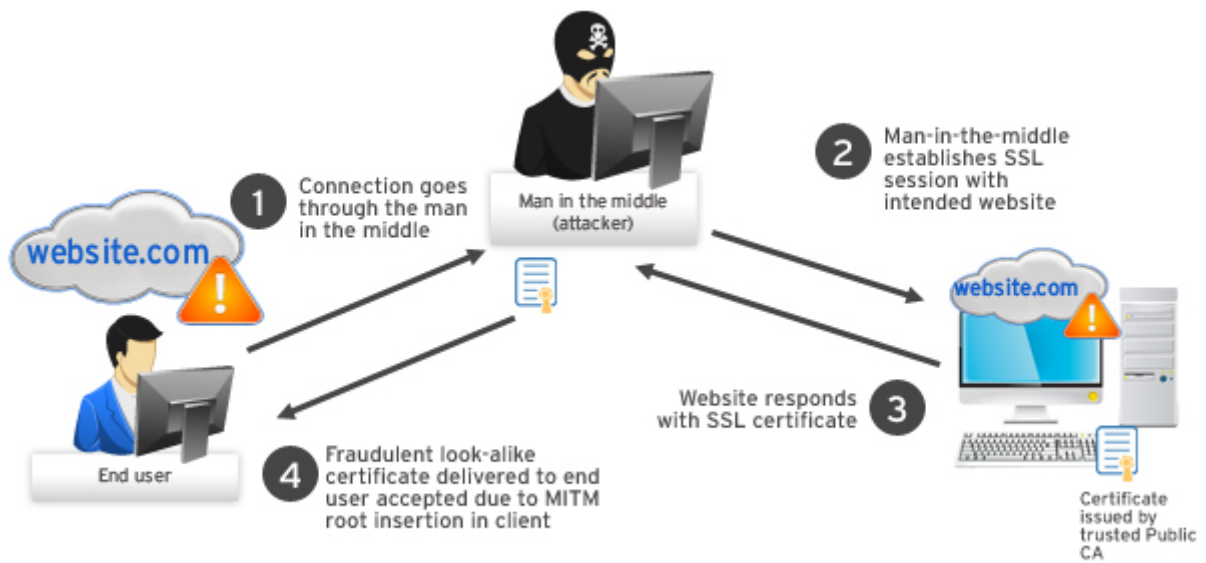


Рисунок 3.19 – Схема атаки при сертифікаті

Є проблеми центрів сертифікації, які можуть призводити до того, що ваш сертифікат можуть перехватити або підробити.

Відправляється сервером сертифікат виданий і підписаний центром сертифікації. В кожному браузері є список довірених центрів сертифікації, і ви можете додавати або видаляти їх. Проблема тут полягає в тому, що якщо ви вирішите видалити великі центри, ви не зможете відвідувати сайти, які використовують підписані цими центрами сертифікати. Сертифікати та центри сертифікації завжди були найслабшою ланкою HTTPS-з'єднання. Навіть якщо все було реалізовано правильно і кожен центр сертифікації має солідний авторитет, все одно складно змиритися з фактом, що доводиться довіряти безлічі третіх сторін. На сьогоднішній день існує більше 650 організацій, здатних видавати сертифікати. Якщо зловмисник зламає будь-яку з них, він роздобуде будь-які сертифікати, які забажає. Навіть коли існував всього один центр сертифікації, VeriSign, побутувала проблема – люди, які повинні були запобігати атакам посередника, продавали послуги перехоплення. Також багато сертифікатів були створені завдяки злому центрів сертифікації. Різні прийоми і трюки використовувалися, щоб змусити атакується користувача довіряти шахрайським сертифікатами.

Коли у вашій Wi-Fi мережі використовується шифрування за допомогою технології WPA2, безпеку сесії заснована на двох складових. По-перше, пароль, який використовується для генерації набагато більш довгого числа (PSK або pre-shared key). По-друге, власне саме рукописання, що відбувається під час установки з'єднання. Якщо зловмисник отримує PSK до Wi-Fi і спостерігає, як ви підключаєтеся до мережі (або на деякий час скидає ваше з'єднання), то згодом зможе розшифрувати ваш Wi-Fi трафік і дізнатися, чим ви займаєтеся.

Вміст HTTPS сайтів буде недоступно, однак HTTP сайти і інші небезпечні HTTP запити додатків на вашому телефоні будуть у відкритому вигляді. Начебто на перший погляд не така велика удача, однак дуже швидко можна дізнатися багато про тип пристрою, і які програми запущені. Крім того, будуть видні DNS запити для резолвінг доменів, що допоможе в ідентифікації активних додатків і сервісів.

Для реалізації атаки необхідно виконання кількох умов. По-перше, потрібен пароль. Крім того, потрібно бути поруч з жертвою для перехоплення трафіку і вміти відключити цільове пристрій від мережі для повторного підключення. Ми запустимо Wireshark, виконаємо налаштування, пов'язані з дешифруванням Wi-Fi пакетів, додамо PSK і дочекаємося EAPOL пакетів з цільового пристрою, що намагається підключитися до мережі.

Для аналізу поведінки цільового пристрою використовуватимемо фільтри з метою виділення шуканих DNS і HTTP пакетів. Повний список доменів, для

яких пристрій виконує резолвінг, також буде доступний після завершення перехоплення. Ця інформація стане в нагоді для з'ясування, які служби використовуються навіть у фоновому режимі.

3.4 Способи захисту Android

Існує кілька способів, якими ви можете захистити себе від хакерських атак:

- Оновлення. Регулярно оновлюйте фірмове ПО. Це допоможе вашому смартфону завжди мати найактуальнішу захист.
- Додатки: Користувачі Android і пристроїв, яким були проведені операції джейлбрейка або отримання Root-прав, повинні бути обережніше при установці нового софту. Якщо ви хочете встановити програму, яку знайшли не в відомих магазинах додатків, то пам'ятайте, що вона може містити віруси і вкрасти ваші дані.
- Wi-Fi: Підключайтеся тільки до тих бездротових мереж, які захищені паролем. Підключаючись до суспільного Wi-Fi, ви, можна сказати, запрошуєте хакерів на екскурсію по своєму смартфону.
- Bluetooth: Ще одна уразливість – це підключення через Bluetooth. Тут хакери можуть підключитися до вашого телефону, перебуваючи недалеко від вас, а ви цього навіть не помітите. Тому вимикайте Bluetooth, коли ви його не використовуєте. Це також допоможе економити заряд акумулятора.
- Повідомлення: Ні в якому разі не відкривайте посилання з листів або SMS-повідомлень, відправників яких ви не знаєте. Радимо вам відразу ж видаляти MMS-повідомлення від невідомих контактів. Це відноситься і до спаму.
- Захист: Особливо на смартфонах Android рекомендуємо встановлювати додаткове додаток для перевірки завантажених файлів. Хорошу і при цьому безкоштовний захист пропонують Avast Security,

Kaspersky Antivirus, Bitdefender Mobile Security i Avira Antivirus Security.

4 ТЕХНІЧНО-ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ

4.1 Вступ

Особиста інформація людини – має велику цінність для кожного.

На сьогоднішній день, захист цієї інформації є найактуальнішою темою. Контакти, фотографії, секретні дані, документи, або прихована інформація являє собою інструментом для маніпулювання як самої жертви, так і тих людей які на неї схожі.

Розробка програмного продукту з економічно вигідною з точки зору економіки на сьогоднішній день, там потреб споживачів у яких на першому місці – це захист власної інформації.

4.2 Матеріальні витрати

В цьому розділі наведено витрати на сировину і матеріали, які розраховані по формулі 4.1:

$$Z_{mat} = \sum_{i=1}^m H_{pi} \cdot Y_i \quad (4.1)$$

де H_{pi} норма витрати 1-го матеріалу на одиницю продукції;

Y_i – ціна одиниці 1-го матеріалу;

m – кількість видів матеріалу.

Розрахунки наведено в табл. 4.1.

Таблиця 4.1 – Матеріали для розробки продукту

Найменування матеріалу	Норма витрат, од.	Ціна, грн. за од.	Сума, грн
Папір	60	0,2	12
картридж для принтеру	1	170	170
CD Диск	3	10	30
Сума			212

4.3 Витрати на дослідження та розробку

В даний пункт витрат входить основна і додаткова заробітна плата за окладами і тарифами наукових співробітників, зайнятих розробкою даної теми, а також єдиний соціальний внесок (ЄСВ).

Розрахунок тривалості робіт за допомогою експертного оцінювання визначається по формулі 4.2:

$$t_j = \frac{3t_{\min} + 2t_{\max}}{5} \quad (4.2)$$

де t_j - очікувана тривалість j -ї роботи;

t_{\min} та t_{\max} – найменша та найбільша тривалість робіт на думку експерта.

Для розрахунку витрат на етапі проектування необхідно визначити тривалість роботи. Розрахунки на тривалість розробки знаходяться в таблиці 4.2.

Розрахунок трудомісткості проекту кожного виконавця наведено в таблиці 4.3.

Таблиця 4.2 – Тривалість етапів розробки

№	Найменування роботи	Тривалість роботи, люд.- год.		
		t_{\min}	t_{\max}	t_i
1	2	3	4	5
1	Уточнення технічного завдання на виконання ДР	3	4	3,4
2	Робота з літературними джерелами, патентний пошук	20	25	22
3	Розробка алгоритму дослідження	20	21	20,4
4	Опис економіко-математичних постановок задач	40	45	42
5	Опис інформаційного забезпечення завдань	1	2	1,4
6	Розробка вихідних форм представлення інформації	18	20	18,8
7	Розробка алгоритмів вирішення задач	30	33	31,2
8	Розроблення та налагодження програмних модулів завдань	25	30	27
9	Тестова налагодження програмних модулів завдань	1	2	1,4
10	Розрахунок кошторису витрат на виконання	30	31	30
11	Оцінка економічної ефективності	20	25	22
12	Оформлення робочого проекту	20	22	20,8
13	Підписання документації	5	8	6,2
Разом				246,6

Таблиця 4.3 Розрахунок витрат на основну заробітну плату

Виконавиць (посада)	Найменування роботи													Трудоміс- кість реалізації проекту
	1	2	3	4	5	6	7	8	9	10	11	12	13	
	Трудоміс- кість, люд- год.													
Керівник	3,4	0,1	1	1	1	0,1	0,1	0,1	0,1				0,1	7
Дослідник (інженер, студент)	3,4	22	20	42	1,4	18	31	27	1,4	30	22	20	6,2	244,4

4.4 Витрати на оплату праці.

До складу оплати праці включаються основна та додаткова заробітна плата за окладами і тарифами, зайнятих безпосередньо виконанням конкретної теми. Розрахунок витрат на основну заробітну плату наведено в таблиці 4.4 .

Таблиця 4.4 Розрахунок витрат на основну заробітну плату

Посада	Розряд	Тарифна ставка, грн	Трудоємність Реалізації проекту	Коефіцієнт Трудової участі	Основна заробітна плата, грн
1	2	3	4	5	6
Керівник проекту	17	8010	3	0,1	2403
Інженер-програміст	10	4859,4	3	1	14578,2
Разом:					16981,2

4.5 Додаткова заробітна плата персоналу

Додаткова плата включає доплату і надбавку до тарифних ставок і посадових окладів у розмірах передбачених чинним законодавством України. Розрахунок додаткової плати наведено в таблиці 4.5

Додаткова заробітна плата становить 12% від основної плати:

$$З_{\text{доп}} = З_{\text{осн}} * 0.12$$

Таблиця 4.5 Розрахунок додаткової заробітної плати

Посада	Додаткова заробітна плата, грн
Керівник проекту	288,36
Інженер-програміст	1749,38
Разом:	2037,74

4.6 Єдиний соціальний внесок

Єдиний соціальний внесок є загальнообов'язковим державно соціальним страхування та розраховується за формулою 4.3:

$$З_{\text{есв}} = (З_{\text{осн}} + З_{\text{доп}}) * 0.22 \quad (4.3)$$

Розрахунок ЄСВ наведено в таблиці 4.6

Таблиця 4.6 Розрахунок єдиного соціального внеску

Посада	Єдиний соціальний внесок, грн
Керівник проекту	592,09
Інженер-програміст	3592,06
Разом:	4184,15

4.7 Амортизаційні відрахування

До основних засобів обладнання необхідних для розробки ДР належать: комп'ютер, принтер.

Нарахування амортизації розраховується за формулою 4.4:

$$A_{\text{нап}} = \frac{Ц * H_a * K_{\text{міс}}}{12} \quad (4.4):$$

$$A_{\text{нар}} = \frac{21199,57 * 0,65 * 3}{12} = 1861,59$$

Розрахунок амортизації обладнання наведено в таблиці 4.7.

Таблиця 4.7 Розрахунок амортизації обладнання

Обладнання	Амортизація обладнання, грн
Комп'ютер	20000
Принтер	1199,57

4.8 Витрати на електроенергію

Витрати на електроенергію визначаємо за формулою 4.5:

$$3e = W_h \cdot P \cdot T_h \quad (4.5)$$

де W_h – потужність використовуваного h-го виду обладнання, кВт.

P – вартість 1 кВт/год електроенергії, грн.

T_h – час роботи h-го виду обладнання, година.

Розрахуємо кількість витрачаємої електроенергії комутатора та принтера

$$3e_{\text{ком}} = 0,02 * 1.68 * 5 = 0,16 (\text{за день})$$

$$0,16 * 91 = 14,56 (\text{за місяць})$$

$$3e_{\text{при}} = 0,04 * 1.68 * 2 = 0,13 (\text{за місяць})$$

4.9 Накладні витрати

До накладних витрат відносяться витрати на опалення, освітлення, орендна плата, адміністративні та загальногосподарські витрати тощо.

Формула для підрахунку накладних витрат:

$$З_{\text{нак}} = З_{\text{осн}} * k_{\text{н}} = 16981,2 * 30\% = 5094,36$$

4.10 Калькуляція собівартості

На підставі проведених розрахунків складаємо калькуляцію собівартості на програмний продукт, яка наведена в таблиці 4.8

Таблиця 4.8 Калькуляція собівартості

Найменування статей	Сума, грн
Матеріали та сировина	212
Основна заробітна плата	16981,2
Додаткова заробітна плата	2037,74
Єдиний соціальний внесок	4184,15
Амортизація основних засобів	1861,59
Витрати на електроенергію	14,69
Накладні витрати	5094,36
Собівартість	30385,73

4.11 Економічна ефективність

Для оцінки діяльності наукових підрозділів підприємств або самостійних наукових організацій також можна використовувати показники економічного ефекту та економічної ефективності.

За рахунок скорочення потрібного часу на моделювання:

$$E_1 = S_{1_{\text{час}}} \Delta t$$

$$\Delta t = t_{\text{max}} - t_{\text{min}} = 272 - 233 = 39$$

$$S_{1_{\text{час}}} = (8010 + 4859.4) / 160 = 80.43$$

$$E_1 = S_{1_{\text{час}}} \Delta t = 39 * 80.43 = 3136.77$$

де $S_{1_{\text{час}}}$ - заробітна плата конструктора,

Δt - різниця потрібного часу на розрахунок у першому та другому варіантах (експертна оцінка).

Розрахунку економічного ефекту:

$$E\phi = \sum_{i=1}^n E_i \cdot N - E_n \cdot \Delta K = (3136.77 * 3) - (30385.73 * 0.25) = 1813.88$$

де i - напрямки економії, $i = (1, n)$,

E_i - економія за напрямками;

N - кількість разів використання інновації (в даному випадку);

ΔK - кошторис витрат на розробку і впровадження інновацій;

E_n - нормативний коефіцієнт ефективності капіталовкладень (0,25-0,3)

Термін окупності НДР розраховується за формулою:

$$T_{ок} = \frac{K}{E\phi}$$

$$\frac{2199,57}{1813,88} = 1,2$$

де $E\phi$ - економічний ефект грн;

K – капіталовкладення (вартість основних засобів (обладнання)), грн.

4.12 Висновки

Проведений аналіз дозволяє зробити висновок що вся розробка програмного продукту вийде дуже дорого , як я і казав це економічно не вигідно але якщо виготовляти більшу кількість продукцій, тобто налагодити масове виробництво, то прибуток буде складати близько одного мільярда гривень у рік.

5 ОХОРОНА ПРАЦІ ТА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

5.1 Загальні питання охорони праці та навколишнього середовища

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

Роботодавець – власник підприємства, установи, організації або уповноважений ним орган, незалежно від форм власності, виду діяльності, господарювання, і фізична особа, яка використовує найману працю.

Працівник - особа, яка працює на підприємстві, в організації, установі та виконує обов'язки або функції згідно з трудовим договором (контрактом). (закон України „Про охорону праці”) [1].

Тема дипломної роботи це "Тестування захисту інформації сучасних Adnroid-смартфонів".

5.2 Структура управління Охорони на підприємстві.

В таблиці 5.1 наведено структуру управління питаннями на підприємстві

Таблиця 5.1 – Структура управління

	Директор, заст. директора, технічний відділ, відділ реалізації продукції, бухгалтерія	60/4	Відділ з охорони праці
--	---	------	------------------------

За цією структурою можна скласти схему підприємства яка показана на рисунку 5.1.

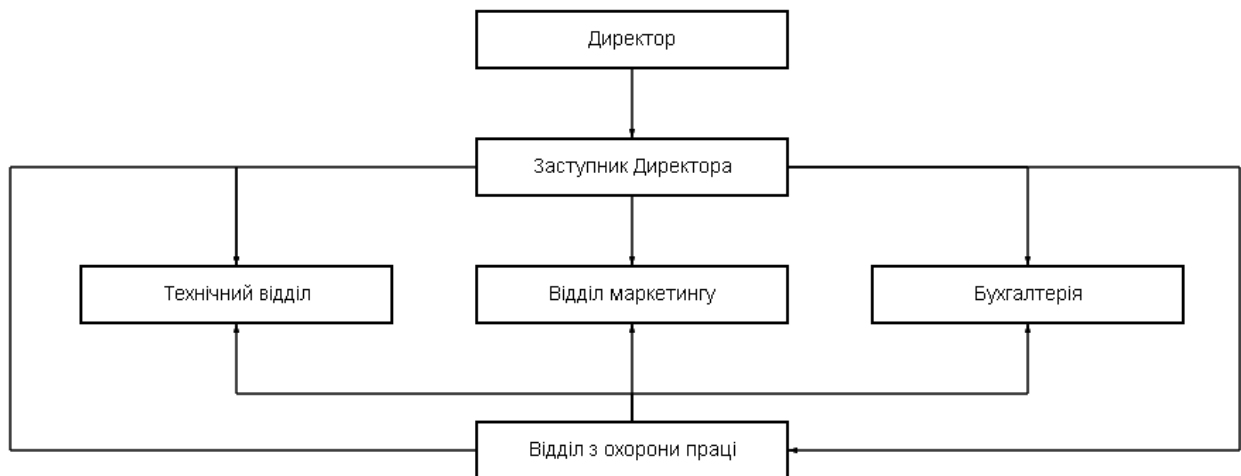


Рис. 5.1 – Структура СУОПІ на підприємстві

В таблиці 5.2 згідно з ДСанПіН 3.3.2-007-98 [2] наведено перелік шкідливих та небезпечних виробничих факторів, які супроводжують працю бакалавра за профілем обраної професії.

Таблиця 5.2 - Перелік шкідливих та небезпечних виробничих факторів

Електрична напруга вище 127 В; Шум;	Вентиляція	Розміри приміщення, м:
Випромінювання – електромагнітні, радіаційні, теплові;	4-ПЕОМ	Довжина – 5;
Пил;		Ширина – 5;
Статична електрика;		Висота – 3.
Іонізація повітря;	Папір	Кількість працюючих – 4
Пожежна небезпека у приміщенні;	Світильники (лампи)	
Неякісне освітлення		

5.4 Характеристика приміщення

Згідно з ДСанПіН 3.3.2.007-98 [2] площа на одну людину повинна бути не менше 6 м², а об'єм не менше ніж 20,0 м³. В нашому приміщенні такі параметри: довжина А= 5м, ширина В=5м, висота Н=3м. Кількість працюючих = 4 людини. Площа S = 25 м². Об'єм v = 75м³.

На одне робоче місце площі виділяється $25 \text{ м}^2 / 4 = 6,25 \text{ м}^2$, що задовольняє нормам. На одне робоче місце об'єму виділяється $75 \text{ м}^3 / 4 = 18,75 \text{ м}^3$, що не задовольняє нормам.

5.5 Виробнича санітарія.

5.5.1 Мікроклімат

Під час роботи в приміщеннях стан людини залежить від мікроклімату.

Такі фактори як температура повітря, вологість, швидкість руху повітря та навіть період року – сильно впливають на працездатність, та здоров'я. Згідно з нормами ДСН 3.3.6.042-99 [3], були визначені оптимальні параметри мікроклімату які наведені в таблиці 5.3.

Таблиця 5.3 – Оптимальні параметри мікроклімату

Категорія робіт	Період року	Температура t , °C	Відносна вологість ϕ , %	Швидкість руху повітря v , м/с
Легка Іа	Холодний	22-24	40-60	$\leq 0,1$
Легка Іа	Теплий	23-25	40-60	$\leq 0,1$

Щоб захиститися від шкідливої температури, в приміщенні розміщений кондиціонер. Для визначення вологості – поставлено вологомір. Таким чином ми захистилися від шкідливого мікроклімату.

5.5.2 Природне освітлення

Приміщення з постійним перебуванням людей повинні мати природне освітлення. Без природного освітлення допускається проектування приміщень, які визначені відповідними державними будівельними нормами та стандартами, а також приміщення, розміщення яких дозволено в підвальних поверхах будівель. В нашому випадку природне освітлення поділяється на бокове, верхнє і комбіноване (верхнє і бокове).

5.5.3 Штучне освітлення

Згідно державних будівельних норм ДБН В. 2.5-28:2018 [4] характеристика зорової роботи має високу точність(III). Щоб покращити роботу та уникнути небезпеки для зору – використовують не тільки штучне, але й природне освітлення. Норми для безпечного освітлення наведені в таблиці 5.4

Таблиця 5.4 – Характеристика виробничого освітлення

Характеристика зорової праці	Мінімальний розмір об'єкта розрізнення	Фон	Контраст	Розряд, підрозряд	Нормоване значення характеристик освітлення		
					Природне освітлення	Штучне освітлення E_{min} , лк	
					$D_{н\ min}$, %	Комбіноване	Загальне
Висока точність	Від 0,3 до 0,5 мм	світлий	середній	III, г	1,2	400	200

5.5.4 Шум та вібрація у робочому приміщенні

У технічному відділі шум та вібрацію створюють комп'ютери, кондиціонер, лазерні-принтери та звичайні паперові-принтери. Згідно з ДСН 3.3.6.037-99[8] рівень шуму у приміщенні не повинен перевищувати 50 дБ(А). Для того аби знизити рівень шуму, використовують: звукопоглинальні перегородки та оббиті стіни звукопоглинальною бавовною. Згідно з ДСН 3.3.6.037-99[8] рівень вібрації у приміщенні не повинен перевищувати 75 дБ(А). Для того аби знизити рівень вібрації, використовують: віброізолюючі перегородки та стійкі меблі.

У моєму приміщенні рівень шуму складає 49 дБ(А), який не перевищує норм ДСН 3.3.6.037-99[8] та рівень вібрації 12 дБ(А) який також не перевищує рівень норм ДСН 3.3.6.037-99[8].

5.5.5 Електромагнітне випромінювання

Для підтримки безпечного рівня електромагнітного випромінювання застосовують наступні заходи: захист збільшенням відстані, захист часом.

Випромінювання в приміщенні становить 15 кВ/м, що відповідає нормам згідно з ДСанПіН 3.3.2-007-98 [5].

Під час роботи в приміщенні, рівень випромінювання може змінюватися, тому потрібно брати перерву чи покидати зону електромагнітного випромінювання.

5.5.6 Електробезпека

Технічний відділ за небезпекою ураження електричним струмом відноситься до приміщень з підвищеною небезпекою відповідно до ПУЕ-2017 [9]. Згідно з НПАОП 0.00-7.15-18 [5] приймаємо І клас захисту від ураження електричним струмом персоналу, тому що комп'ютер має ізоляцію та елементи заземлення, що забезпечують безпеку дотику людини до металевих частин електроприладів.

Параметри електричної мережі:

- рід струму - змінний;
- напруга в мережі - 220В;
- частота – 50 Гц;

Електробезпека електричних приладів забезпечується комплексом конструктивних (прилади виконані в захисних корпусах), схемо-конструктивних (захисне заземлення) та експлуатаційних (мала напруга, вирівнювання потенціалів, ізоляція дротів, огорожувальні пристрої) засобів та способів захисту.

В виробничому приміщенні, з метою захисту від ураження електричним струмом, застосовується захисне заземлення, яке відповідає вимогам ПУЕ-2017 [9] та з'єднується із заземленою нейтралю електроустановки.

5.6 Пожежна безпека

Категорія приміщення згідно з ДСТУ Б В.1.1-36:2016[10] відноситься до категорії В за вибуховою, пожежно-вибуховою та пожежною небезпекою. В приміщенні є легко-займисті матеріали: папір, пластикова стеля, кожані крісла, дерев'яні столи.

Згідно з ДБН В.1.1-7:2016[11], ступінь захисту будівлі відноситься до II рівня. Будівля відноситься до будинків з несучими та огорожувальними конструкціями з цегли та залізобетону.

Первинним засобом захисту пожежогасіння згідно з ДБН В.2.5-56-2015 [12] є вогнегасник. У приміщенні використовують один вуглекислотних вогнегасників ВВК-2. Також у приміщенні встановлена аварійна система пожежогасіння , та звукова аварійна система.

5.7 Охорона навколишнього природного середовища

Згідно Закону України «Про питну воду» (Закон України «Про охорону навколишнього природного середовища»)[7], нормативи питного водопостачання - розрахункова кількість питної води, яка необхідна для забезпечення питних, фізіологічних, санітарно-гігієнічних та побутових потреб однієї людини протягом доби. Методика визначення нормативів питного водопостачання населення розробляється та затверджується центральним органом виконавчої влади з питань житлово-комунального господарства.

Виробництво використовує такі природних ресурсів як: Папір, електрика, вода. Ці природні ресурси не перевищують норму витрат природних ресурсів згідно Закону України «Про охорону навколишнього природного середовища»)[7].

Можливі джерела забруднення навколишнього природного середовища це: відпрацьоване канцелярське приладдя, відходи паперу, побутова сміття. Спосіб захисту – утилізація сміття. Зменшення часу користування електрикою.

ВИСНОВКИ

Під час дипломної роботи було створено програмний продукт, який пройшов тестування на захист інформації.

Програма може використовуватися в усіх сферах, так як має міцний захист від хакерів.

Для розробки даного програмного продукту були використані сучасні технології, що забезпечують довголіття програми.

Проведений аналіз дозволяє зробити висновок що вся розробка програмного продукту вийде не дуже дорого. Це економічно вигідно.

Я помітив, що якщо користувач сам не дасть дозволу на встановлення програми, то і програма не буде встановлена. Також з початку користування Android забороняє користувачеві скачувати або встановлювати програми зі сторонніх сайтів або у не перевічених виробників. Це говорить про те, що якщо користувач не буде давати дозвіл, то і інформація буде надійно захищена.

Але є один великий недолік. Сама операційна система має відкритий код, що означає, що любий програміст зможе знайти недоліки і скористатися ними.

Основною ж вразливістю є сам користувач. Здебільшого крадії інформації використовують соціальну інженерію яка є потужним недоліком як для звичайних користувачів, так і для програмістів. Підсумком є те, що щоб захистити свою інформацію, потрібно придержуватися безпечних правил не тільки в житті, але і в мережі.

Сама програма та її тестування виявились цікавими, хоча б тому, що я набув багато вмінь та навичок в сфері захисту та тестування програмного продукту та коду.

СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ

1. Закон України «Про охорону праці» у редакції від 27.12.2019, підстава - 341-IX
2. ДСанПіН 3.3.2-007-98 Державні санітарні правила і норми. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. – Чинний від 10.12.1998 р.
3. ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень Затверджено постановою Головного санітарного лікаря України від 01 грудня 1999 року, №42
4. ДБН В. 2.5-28:2018 Державні будівельні норми України. Природне і штучне освітлення. – Чинний від 01.03.2019 р.
5. НПАОП 0.00-7.15-18. Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями. Затверджено наказом Міністерства соціальної політики України від 14.02.2018 № 207.
6. ДБН В.1.1-7:2016. Державні будівельні норми України. Пожежна безпека об'єктів будівництва. Загальні положення.- Чинний від 01.01.2017.
7. Закон України «Про охорону навколишнього природного середовища» від 26.06.1991 р. №1264-XII у редакції Редакція від 18.12.2019, підстава - 139-IX
8. ДСН 3.3.6.037-99. Санітарні норми виробничого шуму, ультразвуку та інфразвуку Затверджено постановою Головного санітарного лікаря України від 01 грудня 1999 року, № 37.
9. ПУЕ-2017. Правила улаштування електроустановок. Електрообладнання спеціальних установок. – К., 2017
10. ДСТУ Б.В.1.1:36-2016. Визначення категорій приміщень, будинків та зовнішніх установок за вибух пожежною та пожежною небезпекою. –України, 2016
11. <https://zakon.rada.gov.ua/rada/show/va042282-99#Text>
12. <https://habr.com/ru/post/543346/>
13. <https://dou.ua/lenta/articles/language-rating-jan-2021/?from=doufp>
14. <https://andrax.thecrackertechnology.com/>
15. <https://developer.android.com/reference/android/app/Activity>

ДОДАТОК А

Фрагмент головного файлу MainActivity.java

```
package com.ntukhpi.otp.serebrianskyi.chatmessage;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Message;
import android.view.View;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.text.format.DateFormat;
import com.firebase.ui.auth.AuthUI;
import com.firebase.ui.database.FirebaseListAdapter;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.FirebaseDatabase;
public class MainActivity extends AppCompatActivity {

    private static int SIGN_IN_CODE = 1;
    private RelativeLayout activity_main;
    private FirebaseListAdapter<message> adapter;
    private FloatingActionButton sendButton;

    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if(requestCode==SIGN_IN_CODE) {
            if(resultCode == RESULT_OK) {
                Snackbar.make(activity_main,"you are in",Snackbar.LENGTH_LONG).show();
                displayAllMessages();
            }
        }
    }
}
```

```

    } else {
        Snackbar.make(activity_main,"you are not !!!in",Snackbar.LENGTH_LONG).show();
    }

}

}

@Override

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    activity_main = findViewById(R.id.activity_main);
    sendButton = findViewById(R.id.btndsend);
    sendButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            EditText textField = findViewById(R.id.edit_tex_lay);
            if(textField.getText().toString()=="")
                return;
            FirebaseDatabase.getInstance().getReference().push().setValue(
                new message(FirebaseAuth.getInstance().getCurrentUser().getEmail(),
                    textField.getText().toString()
                )
            );
            textField.setText("");
        }
    });
    if(FirebaseAuth.getInstance().getCurrentUser()== null)

startActivityForResult(AuthUI.getInstance().createSignInIntentBuilder().build(),SIGN_IN_CODE);
    else
        Snackbar.make(activity_main,"you are in",Snackbar.LENGTH_LONG).show();

```

```

        displayAllMessages();
    }

    private void displayAllMessages() {
        ListView listofmessages = findViewById(R.id.list_messages);

        adapter = new FirebaseListAdapter<message>(this,message.class,R.layout.list_item,
        FirebaseDatabase.getInstance().getReference()) {

            @Override

            protected void populateView(View v, message model, int position) {

                TextView mess_user, mess_time, mess_text;

                mess_user = v.findViewById(R.id.message_user);
                mess_time = v.findViewById(R.id.message_time);
                mess_text = v.findViewById(R.id.message_text);

                mess_user.setText(model.getUserName());
                mess_text.setText(model.getTextMessage());

                mess_time.setText(DateFormat.format("dd-mm-yyyy
                HH:mm:ss",model.getMessagetime()));

            }

        };

        listofmessages.setAdapter(adapter);
    }
}

```

message.Java

```

package com.ntukhpi.otp.serebrianskyi.chatmessage;

import java.util.Date;

public class message {

    public String userName;
    public String textMessage;
    private long messagetime;

    public message(){ }

    public String getUserName() {

        return userName;
    }

}

```

```
public void setUsername(String userName) {  
    this.userName = userName;  
}  
public String getTextMessage() {  
    return textMessage;  
}  
  
public void setTextMessage(String textMessage) {  
    this.textMessage = textMessage;  
}  
  
public long getMessageTime() {  
    return messageTime;  
}  
  
public void setMessageTime(long messageTime) {  
    this.messageTime = messageTime;  
}  
  
public Message(String userName, String textMessage){  
    this.userName = userName;  
    this.textMessage = textMessage;  
    this.messageTime = new Date().getTime();  
  
}  
}
```

ДОДАТОК Б

Код файлів конфігурації проекту

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/btnsend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="3dp"
        android:layout_marginBottom="29dp"
        android:clickable="true"
        android:focusable="true"
        android:src="@drawable/ic_sendbut"
        app:fabSize="mini" />
    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/text_layuot"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="-3dp"
        android:layout_marginBottom="25dp"
        android:layout_toStartOf="@id/btnsend">
        <EditText
            android:id="@+id/edit_tex_lay"
            android:layout_width="348dp"
            android:layout_height="match_parent"
            android:autofillHints=""
            android:hint="@string/write_message"
            android:inputType="text" />
    </com.google.android.material.textfield.TextInputLayout>
    <ListView
        android:id="@+id/list_messages"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_above="@id/text_layuot"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:divider="@android:color/transparent"
```



```
        android:dividerHeight="16dp"/>
</RelativeLayout>
```