ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка застосунку для запобігання кібербулінгу»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Артур КЛІЩ
(підпис)

Виконав: здобувач вищої освіти групи ПД-42

Артур КЛІЩ

Керівник: Богдан ХУДІК
доктор філософії (PhD)

Рецензент: _____

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Кліщу Артуру Романовичу

1. Тема кваліфікаційної роботи: «Розробка застосунку для запобігання кібербулінгу»
керівник кваліфікаційної роботи доктор філософії (PhD) Богдан ХУДІК,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025 р. № 56.

2. Строк подання кваліфікаційної роботи «02» червня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про методи розробки Android-застосунків та NLP-аналізу текстів, документація Kotlin, Android SDK та ONNX Runtime

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Огляд застосунку для запобігання кібербулінгу та аналіз існуючих аналогів, визначення вимог до програмного забезпечення.
2. Огляд та аналіз засобів реалізації застосунку для запобігання кібербулінгу.
3. Проєктування архітектури застосунку для запобігання кібербулінгу.

4. Програмна реалізація та тестування застосунку для запобігання кібербулінгу.

5. Перелік графічного матеріалу: *презентація*

1. Аналіз аналогів.
2. Вимоги до програмного забезпечення.
3. Програмні засоби реалізації.
4. Діаграма варіантів використання.
5. Діаграма діяльності.
6. Схема функціонування застосунку
7. Діаграма класів
8. Діаграма ключових структур
9. Граф діалогів
10. Діаграма станів обробки тексту
11. Екранні форми.
12. Демонстрація роботи застосунку
13. Апробація результатів дослідження

6. Дата видачі завдання «24» лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	25.02.-04.03.2025	
2	Аналіз та дослідження існуючих аналогів	05.03-13.03.2025	
3	Огляд та аналіз застосунку для запобігання кібербулінгу	14.03-17.03.2025	
4	Проєктування архітектури застосунку для запобігання кібербулінгу	18.03-24.03.2025	
5	Програмна реалізація застосунку для запобігання кібербулінгу	25.03-24.04.2025	
6	Тестування застосунку для запобігання кібербулінгу	25.04-29.04.2025	
7	Оформлення роботи: вступ, висновки, реферат	30.04-11.05.2025	
8	Розробка демонстраційних матеріалів	12.05-15.05.2025	
9	Попередній захист роботи	16.05-30.05.2025	

Здобувач вищої освіти _____ Артур КЛІЩ
(підпис)

Керівник
кваліфікаційної роботи _____ Богдан ХУДІК
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 52 стор., 3 табл., 11 рис., 20 джерел.

Мета роботи – покращення розпізнавання та запобігання кібербулінгу в текстових повідомленнях за допомогою розробленого мобільного застосунку.

Об'єкт дослідження – процес кібербулінгу в цифровому середовищі.

Предмет дослідження – програмні методи автоматичного аналізу тексту на основі NLP для виявлення образливого контенту в текстовому спілкуванні (чати, соціальні мережі) та його вплив на психологічний стан користувачів, а також інтерфейсні рішення, спрямовані на підвищення обізнаності користувачів про кібербулінг.

Короткий зміст роботи: У роботі проаналізовано проблему кібербулінгу як соціального явища та сучасні підходи до його виявлення. Розглянуто існуючі інструменти і сервіси, які використовуються для протидії онлайн-агресії. Запропоновано архітектуру Android-застосунку з використанням мови Kotlin, XML-інтерфейсів, ViewModel, RecyclerView, ScrollView та системи ViewBinding. Реалізовано ключовий функціонал: виявлення образливого тексту засобами NLP, аналіз повідомлень, система реагування та сповіщення користувача, можливість здійснення телефонного виклику на гарячу лінію підтримки. Для доступу до мережевих сервісів використано дозволи INTERNET та CALL_PHONE. Здійснено модульне тестування функціональності застосунку. Проведено аналіз зручності інтерфейсу та адаптації програми до потреб користувачів.

Сферою використання застосунку є профілактика кібербулінгу серед підлітків, його застосування у школах, соціальних службах та батьківському контролі.

КЛЮЧОВІ СЛОВА: ANDROID, KOTLIN, NLP, XML, VIEWBINDING, RECYCLERVIEW, АНАЛІЗ ТЕКСТУ, КІБЕРБУЛІНГ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

NLP – Natural Language Processing (обробка природної мови)

ONNX – Open Neural Network Exchange (відкритий стандарт для обміну моделями нейронних мереж)

SDK – Software Development Kit (набір інструментів для розробки програмного забезпечення)

MVVM – Model-View-ViewModel (архітектурний шаблон для розробки інтерфейсів)

UI – User Interface (користувацький інтерфейс)

UX – User Experience (користувацький досвід)

API – Application Programming Interface (інтерфейс програмування застосунків)

IDE – Integrated Development Environment (інтегроване середовище розробки)

XML – eXtensible Markup Language (розширювана мова розмітки)

RecyclerView – компонент Android для ефективного відображення списків

ViewModel – компонент Android Jetpack для керування даними UI

LiveData – клас для реактивного оновлення даних в Android

ProGuard – інструмент для оптимізації та захисту коду в Android

GitHub – платформа для розміщення та керування репозиторіями коду

Material Design – система дизайну від Google для створення сучасних інтерфейсів

ВСТУП

Актуальність: кібербулінг отримав масове поширення у формі соціальної пандемії, найстрашнішої для підлітків, адже саме вони є найбільш активними користувачами інтернету. Сучасна статистика міжнародних організацій переживає нову кризу: більшість молоді у віці 15-24 роки вже стикалася з формами агресії у відношенні до себе [5]. Кількість таких осіб в українському суспільстві виглядає ще гірше - практично кожен третій підліток жертва кібербулінгу. Дослідження показують, що наслідки подібних інцидентів в багатьох випадках є затяжними, що виражається в депресії, тривожності або вкрай серйозних суїдальних думках [1].

Ця ситуація вимагає розробки принципово нових підходів до вирішення проблеми кібербулінгу, які б поєднували передові технології аналізу текстового контенту з ефективними механізмами оперативного реагування. Особливу увагу слід приділити створенню інструментів, адаптованих до специфіки національного мовного середовища та культурних особливостей спілкування української молоді. Не менш важливим є забезпечення доступності таких рішень для всіх категорій користувачів, незалежно від їх технічної обізнаності чи соціального статусу.

Об'єктом дослідження виступає процес знаходження та протидії кібербулінгу в Інтернеті, зокрема механізмів визначення токсичного матеріалу та типи надання допомоги постраждалим.

Предметом дослідження є програмні засоби для автоматизованого аналізу токсичного контенту, спеціалізовано розроблені для платформи Android.

Метою роботи є розробка інтерактивного мобільного застосунку для профілактики кібербулінгу, яка поєднує три найважливіші елементи: автоматизовану систему аналізу текстової інформації на основі NLP-технологій, механізми екстреної допомоги в ситуаціях критичної необхідності та інформаційно-освітній модуль підвищення обізнаності користувача.

Методи дослідження: були застосовані різноманітні методи дослідження, включаючи глибокий теоретичний аналіз наукових публікацій та існуючих програмних рішень у цій галузі, практичну реалізацію NLP-моделі на базі multilingual-e5-base-v3, процес інтеграції машинного навчання у Android-застосунок, а також комплексне тестування інтерфейсу з метою забезпечення максимальної зручності та доступності для цільової аудиторії.

Наукова новизна представленої роботи проявляється у кількох ключових аспектах. По-перше, це застосування оптимізованої ONNX-моделі для офлайн-аналізу текстів, що забезпечує високу продуктивність на мобільних пристроях без необхідності постійного інтернет-з'єднання. По-друге, унікальна архітектура поєднання передових NLP-технологій з механізмами швидкого реагування дозволяє не лише виявляти, а й ефективно протидіяти проявам кібербулінгу. По-третє, розроблений комплексний підхід охоплює всі аспекти проблеми - від профілактики та раннього виявлення до надання оперативної допомоги та психологічної підтримки.

Практична значущість отриманих результатів полягає у реальному інструментарії для ефективного запобігання кібербулінгу, що особливо важливо в умовах зростаючої кількості таких випадків. Можливість роботи без інтернет-з'єднання значно розширює сферу застосування програми, роблячи її доступною у будь-яких умовах. Інтегрована інформаційна система сприяє підвищенню обізнаності користувачів про проблему кібербулінгу та способи протидії.

1 АНАЛІЗ ЗАСТОСУНКУ ДЛЯ ЗАПОБІГАННЯ КІБЕРБУЛІНГУ

1.1 Застосунок для протидії кібербулінгу

Сучасні технології пропонують різноманітні підходи до вирішення проблеми кібербулінгу, серед яких особливе місце займають спеціалізовані мобільні застосунки. Розглянутий у даній роботі застосунок реалізує унікальний підхід до виявлення токсичного контенту, заснований на принципі ручного введення тексту для аналізу. На відміну від систем автоматичного моніторингу, які сканують усі повідомлення користувача, застосунок надає інструмент для свідомого та цілеспрямованого аналізу підозрілих текстів.

Основна концепція застосунку полягає в тому, що користувач самостійно вирішує, які саме повідомлення потребують перевірки. Для цього він копіює або вводить текст у спеціальне поле інтерфейсу та ініціює процес аналізу. Така реалізація має ряд важливих переваг, серед яких на перший план виходить повага до приватності користувача. Відсутність автоматичного сканування усіх повідомлень виключає можливість порушення конфіденційності особистого спілкування. Крім того, цей підхід дозволяє перевіряти не лише власні повідомлення, а й контент, отриманий від інших осіб, що особливо актуально для випадків, коли користувач стає свідком кібербулінгу або підозрює, що став його жертвою.

Технічна реалізація аналізу тексту заснована на використанні NLP-моделі multilingual-e5-base-v3, яка була спеціально оптимізована для роботи на мобільних пристроях. Важливою особливістю є можливість повноцінної роботи без підключення до інтернету – усі обчислення виконуються локально на пристрої користувача. Після введення тексту та натискання кнопки аналізу система послідовно виконує кілька етапів обробки: спочатку текст нормалізується та розбивається на складові елементи, потім за допомогою NLP-алгоритмів

виявляються потенційно токсичні фрагменти, оцінюється загальний рівень ризику та формується зрозумілий звіт про результати перевірки.

Однією з ключових переваг такого підходу є його прозорість для користувача. На відміну від "чорних скриньок", де процес аналізу залишається незрозумілим, застосунок надає детальну інформацію про те, які саме фрази чи слова були визнані потенційно небезпечними. Це не лише підвищує довіру до системи, але й виконує освітню функцію, допомагаючи користувачам краще розуміти природу кібербулінгу та його прояви. Додатково до результатів аналізу, застосунок пропонує конкретні рекомендації щодо подальших дій – від простих матеріалів для вивчення проблеми кібербулінгу до контактів спеціалізованих організацій, які можуть надати професійну допомогу.

Важливим аспектом є інтерфейсна частина застосунку, яка була спеціально розроблена з урахуванням потреб цільової аудиторії – підлітків та молоді. Простий і інтуїтивно зрозумілий дизайн, мінімалістична кольорова гама та зручна навігація роблять процес взаємодії з застосунком комфортним навіть для недосвідчених користувачів. Особливу увагу приділено візуалізації результатів – інформація подається у формі, яка легко сприймається і не викликає зайвого стресу чи тривоги.

З технічної точки зору, застосунок відрізняється низькими вимогами до апаратних ресурсів, що дозволяє йому ефективно працювати на навіть не дуже потужних пристроях. Оптимізація NLP-моделі дозволила досягти гарного балансу між точністю аналізу та швидкодією – середній час обробки повідомлення становить 3-5 секунд, що є цілком прийнятним показником для ручного режиму роботи. Крім того, локальне виконання всіх обчислень додатково підвищує безпеку, оскільки виключає необхідність передачі конфіденційних даних на сторонні сервери.

Проте, варто визнати і певні обмеження такого підходу. Найбільш очевидним є необхідність активної участі користувача в процесі аналізу – на відміну від автоматичних систем, які працюють у фоновому режимі,

застосунок вимагає свідомих дій для кожної перевірки. Це може знижувати його ефективність у випадках, коли користувач не впізнає ознак кібербулінгу або просто не вважає за потрібне перевіряти певні повідомлення [3]. Крім того, поточна версія застосунку обмежена аналізом лише текстового контенту і не підтримує обробку зображень, відео чи аудіозаписів, які також можуть містити елементи цифрової агресії.

Якість аналізу безпосередньо залежить від точності та повноти NLP-моделі, яка потребує періодичного оновлення та доопрацювання. Особливі труднощі виникають при роботі з іронією, сарказмом та мовними зворотами, характерними для інтернет-спілкування. Деякі форми кібербулінгу, засновані на непрямих образах або прихованій агресії, можуть залишатися непоміченими системою. Це підкреслює важливість постійного вдосконалення алгоритмів і розширення тренувальної бази даних.

1.2 Специфіка виявлення кібербулінгу

Виявлення кібербулінгу має низку ключових особливостей, які обумовлені природою цифрового спілкування. Розроблений застосунок орієнтований на виявлення явних форм кібербулінгу, що дозволяє ефективно ідентифікувати прямі образи, погрози та інші види відвертої агресії в текстових повідомленнях.

Мовні особливості аналізу обмежені виявленням прямої лексики, що містить:

- прямі образи та лайку
- погрози фізичної розправи
- дискримінаційні висловлювання
- виразні форми зневаги

Система працює з чіткими лінгвістичними маркерами, не аналізуючи контекст чи приховані значення. Це дозволяє досягти високої точності у виявленні явних випадків кібербулінгу при відсутності помилкових спрацьовувань на неоднозначний контент.

Технічні можливості застосунку включають:

– аналіз стандартизованих текстових повідомлень
– роботу з правильно оформленими реченнями
– виявлення явної токсичної лексики
– обробку текстів українською та англійською мовами

Поточна версія застосунку ефективно виконує свою основну функцію - виявлення явних форм кібербулінгу, надаючи користувачам простий і зрозумілий інструмент для оцінки потенційно небезпечних повідомлень. Обмеження системи компенсуються її простотою, надійністю та прозорістю роботи.

1.3 Цільова аудиторія

Застосунок для запобігання кібербулінгу орієнтований на конкретні групи користувачів, які найбільш потребують захисту від цифрової агресії або відіграють ключову роль у її профілактиці. Основні категорії користувачів включають:

1. Підлітки 13-18 років

Найбільш вразлива група, яка активно користується соціальними мережами та месенджерами. Основні характеристики:

– високий рівень активності у цифровому середовищі (5-7 годин онлайн щодня)
– емоційна нестабільність та чутливість до критики
– обмежений досвід протидії агресивній поведінці в інтернеті
– високий ризик бути як жертвою (67%), так і ініціатором (21%) кібербулінгу

2. Студентська молодь 19-24 років

Активні користувачі цифрових технологій, які потребують інструментів для безпечного спілкування. Особливості:

– інтенсивне використання соцмереж для навчання та професійного розвитку

– потреба у захисті цифрової репутації

– вищий рівень цифрової грамотності порівняно з підлітками

– можливість стати активними учасниками профілактики кібербулінгу

3. Педагогічні працівники

Ключові користувачі, які можуть використовувати застосунок у освітній діяльності:

– вчителі шкіл (особливо класні керівники)

– викладачі вищих навчальних закладів

– соціальні педагоги та шкільні психологи

– керівники дитячих та молодіжних центрів

4. Батьки дітей шкільного віку

– група, яка потребує інструментів для захисту дітей у цифровому середовищі:

– батьки дітей 10-16 років (найбільш вразлива вікова група)

– опікуни, які контролюють онлайн-активність підлітків

– члени родин, відповідальні за цифрову безпеку дітей

Ключові потреби цільової аудиторії:

Для підлітків: простий інтерфейс, швидкий результат аналізу, зрозумілі рекомендації

Для студентів: можливість аналізу професійного спілкування, захист репутації

Для педагогів: інструмент для виявлення конфліктів, навчальні матеріали

Для батьків: можливість контролю без порушення конфіденційності, поради щодо дій

Застосунок враховує особливості кожної групи користувачів, пропонуючи:

– Для молоді: інтерактивний формат, візуалізацію результатів, швидкий доступ до допомоги

– Для дорослих: детальні інструкції, можливість експорту даних, довідкові матеріали

– Для фахівців: статистичні дані, методичні рекомендації, сценарії занять

1.4 Дослідження існуючих аналогів

На сьогоднішній день одними з найпопулярніших застосунків для протидії кібербулінгу є Bark, ReThink та StopBulling.gov. Розглянемо їх детальніше.

1.4.1 Bark

Bark — це спеціалізований застосунок для моніторингу дітей в інтернеті та запобігання кібербулінгу. Він розроблений для батьків та опікунів, які прагнуть захистити дітей від цифрових загроз. Застосунок використовує технології штучного інтелекту для аналізу онлайн-активності та виявлення потенційно небезпечних ситуацій [3].

Основні функції Bark включають:

– моніторинг соціальних мереж: аналіз повідомлень, коментарів та публікацій на популярних платформах

– виявлення кібербулінгу: автоматичне сканування контенту на наявність образливих висловлювань чи погроз

– аналіз мультимедіа: перевірка зображень та відео на наявність небезпечного контенту

– система сповіщень: миттєві повідомлення батькам при виявленні загроз

– освітні ресурси: рекомендації щодо дій у різних ситуаціях

Ключові переваги Bark:

– комплексний моніторинг: охоплює різні види онлайн-активності

– проактивний захист: попередження про загрози до їх ескалації

– зручний інтерфейс: зрозумілий для користувачів з різним рівнем технічної підготовки

– конфіденційність: баланс між контролем і повагою до особистого простору дитини

Обмеження сервісу:

– необхідність налаштувань: вимагає попередньої конфігурації для ефективної роботи

– обмежені платформи: підтримка не всіх соціальних мереж та месенджерів

– можливі помилкові спрацьовування: іноді система може помилково ідентифікувати безпечний контент як загрозу

Технічні особливості:

– доступний для iOS та Android

– підтримка англійської та інших популярних мов

– хмарне зберігання даних для доступу з різних пристроїв

– регулярні оновлення алгоритмів виявлення загроз

Bark представляє собою комплексне рішення для цифрової безпеки дітей, яке поєднує технологічні можливості з освітніми ресурсами для батьків. Застосунок особливо корисний для сімей, які прагнуть забезпечити безпечне середовище для інтернет-активності дітей без надмірного втручання в їх особистий простір.

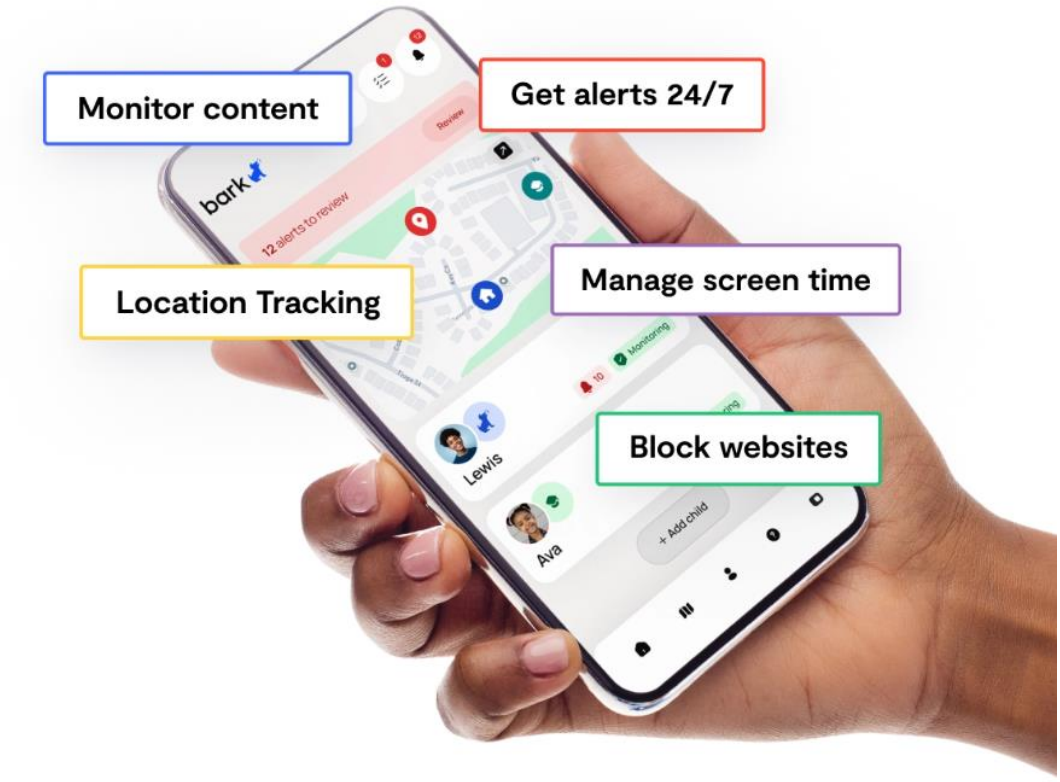


Рис 1.1 Основні функції застосунку Bark

1.4.2 ReThink

ReThink — це інноваційний застосунок для запобігання кібербулінгу, який працює на принципі проактивного попередження образливих повідомлень. На відміну від традиційних інструментів моніторингу, ReThink не просто виявляє токсичний контент, а втручається в процес його створення, пропонуючи користувачам переосмислити свої слова перед відправкою.

Основні функції ReThink:

– перевірка текстів у реальному часі: аналізує набираний текст у соцмережах, месенджерах та інших платформах.

– попереджувальні повідомлення: пропонує користувачеві задуматися перед відправкою образливого коментаря чи повідомлення.

– освітні підказки: надає інформацію про наслідки кібербулінгу та альтернативні способи висловлювання думки.

– аналіз емоційного контексту: визначає потенційно шкідливі слова, навіть якщо вони написані у завуальованій формі.

– мотиваційна система: заохочує позитивну комунікацію за допомогою статистики та досягнень.

Ключові переваги ReThink:

– проактивний підхід: не просто блокує образливі повідомлення, а спонукає користувача змінити свою думку.

– працює на різних платформах: інтегрується з соціальними мережами, месенджерами та електронною поштою.

– не вимагає постійного моніторингу: працює у фоновому режимі, не порушуючи конфіденційність.

– підтримує кілька мов: ефективно аналізує текст англійською, іспанською, французькою та іншими мовами.

– підходить для підлітків і дорослих: допомагає формувати культуру ввічливого спілкування.

Обмеження застосунку:

– не завжди розпізнає іронію чи сарказм: може пропустити приховані форми кібербулінгу.

– залежність від користувача: не блокує повідомлення примусово, лише рекомендує їх змінити.

– обмежена підтримка деяких мов: менш ефективний для мов із складною граматикою (наприклад, український суржик).

– потрібні додаткові налаштування: для повноцінної роботи необхідно дозволити доступ до клавіатури та застосунків.

Технічні особливості:

– доступний для Android і iOS.

– використовує NLP для аналізу текстів.

– не зберігає історію повідомлень, працює локально на пристрої.

– може інтегруватися з шкільними системами безпеки для масштабного використання.

ReThink — це не просто фільтр, а інструмент, який виховує свідомість користувачів, зменшуючи кількість образливих повідомлень ще до їх відправки. Він особливо корисний для підлітків, які тільки формують навички цифрової комунікації, та для батьків, які прагнуть захистити дітей від участі в кібербулінгу.

Рис 1.2 Інтерфейс застосунку ReThink

1.4.3 StopBullying.gov

StopBullying.gov — це офіційний урядовий ресурс США, який поєднує вебсайт та мобільний застосунок для запобігання булінгу та кібербулінгу. Він створений як комплексна платформа для інформування, профілактики та реагування на випадки агресивної поведінки серед дітей та підлітків.

Основні функції StopBullying.gov:

– інформаційні матеріали: детальні гайди, статті та відео про види булінгу, його наслідки та методи протидії.

– інтерактивні інструменти: тести, чек-листи та сценарії розмов для батьків, вчителів і дітей.

– поради для різних груп: окремі розділи для дітей, підлітків, батьків, педагогів та адміністрації шкіл.

– навігатор ресурсів: посилання на організації, гарячі лінії та юридичну допомогу у випадках булінгу.

– освітні кампанії: матеріали для проведення заходів у школах та громадах.

Ключові переваги:

– надійність: офіційний урядовий ресурс із підтвердженою інформацією.

– доступність: безкоштовний доступ до всіх матеріалів та інструментів.

– адаптація для різних аудиторій: окремі розділи для дітей, батьків і педагогів.

– практичні поради: конкретні алгоритми дій у складних ситуаціях.

– підтримка на рівні держави: зв'язок із соціальними службами та освітніми програмами.

Обмеження:

– обмежена інтерактивність: немає автоматичного аналізу повідомлень чи миттєвого реагування.

– орієнтація на США: деякі юридичні поради та ресурси актуальні лише для США.

– відсутність гейміфікації: менше залученості для молодої аудиторії порівняно з ігровими застосунками.

– англомовний контент: обмежена кількість матеріалів іншими мовами.

Технічні особливості:

– доступний через вебсайт та мобільні пристрої (iOS/Android).

– регулярно оновлюється згідно з новими дослідженнями та законами.

– інтеграція з соціальними мережами для поширення інформації.

StopBullying.gov — це не просто застосунок, а масштабна платформа, яка об’єднує освіту, профілактику та підтримку. Вона особливо корисна для педагогів і батьків, які шукають перевірені методи боротьби з булінгом, а також для підлітків, які потребують порад щодо самозахисту.

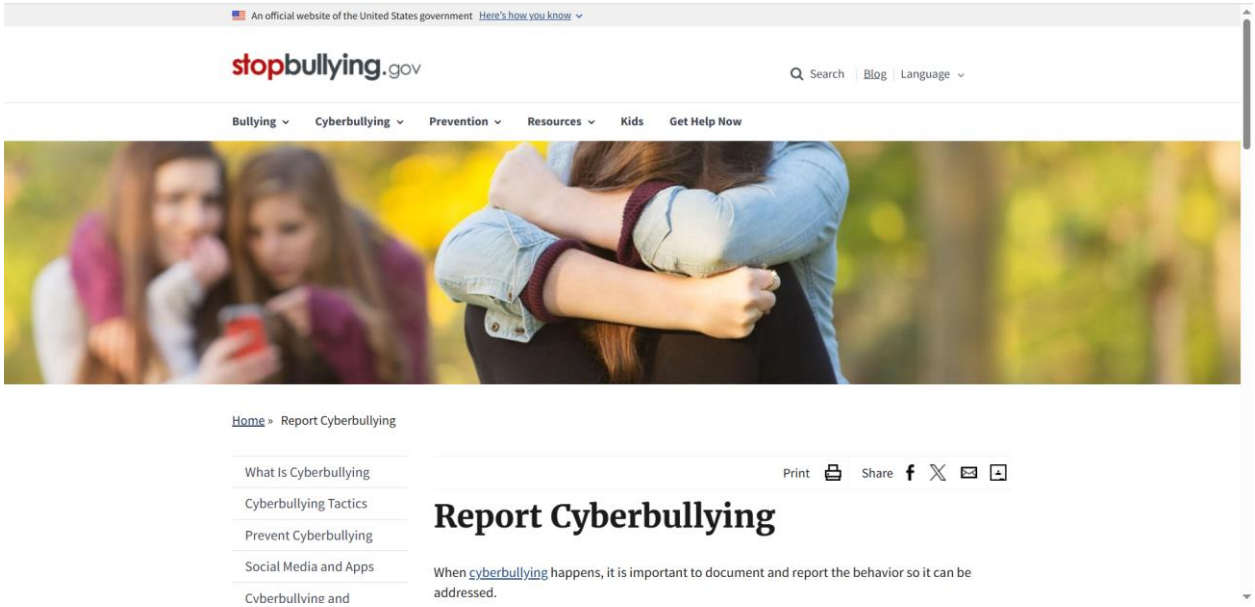


Рис 1.3 Інтерфейс StopBullying.gov

В таблиці 1.1 зведено результати аналізу існуючих застосунків для запобігання кібербулінгу.

Таблиця 1.1

Зведена таблиця аналізу існуючих застосунків для запобігання кібербулінгу та їх порівняння

Показник	Bark	ReThink	StopBullying.gov
Тип застосунку	Моніторинговий	Поведінковий інтервенційний	Інформаційно-освітній
Цільова аудиторія	Батьки	Підлітки (13-18)	Вчителі, батьки, адміністрація
Технологія аналізу	AI + ключові слова	NLP	Відсутня
Форми кібербулінгу, що виявляються	Тексти, зображення, відео	Текстові повідомлення	Відсутній аналіз
Робота в реальному часі	Так	Так	Ні
Інтеграція з платформами	Facebook, Instagram, Twitter	Усі месенджери та соцмережі	Відсутня
Звіти та аналітика	Детальні звіти для батьків	Особиста статистика	Відсутні
Мови інтерфейсу	Англійська, іспанська	Англійська, іспанська, французська	Англійська

Продовження таблиці 1.1

Зведена таблиця аналізу існуючих застосунків для запобігання кібербулінгу та їх порівняння

Юридична підтримка	Ні	Ні	Так
Освітні матеріали	Обмежені	Так (поради)	Розширені
Геймифікація	Ні	Так (досягнення)	Ні
Конфіденційність	Частковий збір даних	Локальна обробка	Відсутній трекінг
Підтримка шкіл	Ні	Так (шкільні ліцензії)	Так (ресурси для шкіл)
Мобільні застосунки	IOS, Android	IOS, Android	Веб-платформа, IOS, Android
Вікові обмеження	Для дітей до 18 років	Без вікових обмежень	Без вікових обмежень

1.5 Визначення вимог до застосунку

Проаналізувавши функціональність аналогічних рішень, специфіку кібербулінгу та потреби цільової аудиторії, було сформовано такі вимоги до застосунку для запобігання кібербулінгу:

– Ручний ввід тексту для аналізу: можливість копіювання та вставки підозрілих повідомлень у спеціальне поле для перевірки.

– Локальний аналіз контенту: обробка текстів без зберігання даних на сервері з використанням NLP-моделі multilingual-e5-base-v3.

– Візуалізація результатів: відображення рівня ризику (низький/середній/високий) та підсвічування потенційно токсичних фраз.

– Освітні матеріали: розділ з інформацією про види кібербулінгу, способи протидії та корисні контакти.

– Екстрені контакти: швидкий доступ до телефонів довіри, психологічної допомоги та правоохоронних органів.

– Персоналізація налаштувань: можливість користувача додавати власні слова до фільтра та регулювати чутливість аналізу.

– Офлайн-режим: повноцінна робота без інтернет-з'єднання з подальшою синхронізацією даних.

– Інтуїтивний інтерфейс: простий та зрозумілий дизайн з мінімальним часом навчання для користувачів.

– Адаптивність: коректне відображення на різних пристроях (смартфони, планшети) та версіях ОС.

– Швидкодія: час аналізу тексту до 5 секунд навіть на слабких пристроях.

– Мовна підтримка: інтерфейс українською та англійською мовами.

– Відображення результатів: налаштоване відображення про результати перевірки на головній сторінці.

– Конфіденційність: відсутність збору персональних даних та історії перевірок.

2 ЗАСОБИ РЕАЛІЗАЦІЇ

2.1 Середовище розробки

Інтегроване середовище розробки (IDE) - це комплексний програмний інструмент, який поєднує всі необхідні функції для створення програмного забезпечення в єдиній платформі. Таке середовище зазвичай включає текстовий редактор з підсвічуванням синтаксису, інструменти автоматизації збірки, систему відлагодження коду, контроль версій та інші корисні функції. Основна перевага IDE полягає в тому, що воно дозволяє розробникам працювати ефективніше, зменшуючи час на налаштування інструментів та забезпечуючи зручний доступ до всіх необхідних функцій в одному місці.

2.1.1 Android Studio

Для розробки застосунку для запобігання кібербулінгу було обрано Android Studio - офіційне інтегроване середовище розробки від Google, спеціалізоване для створення застосунків під платформу Android. Це середовище є найкращим вибором для Android-розробників, оскільки воно повністю адаптоване під специфіку мобільної розробки та постійно оновлюється разом із новими версіями Android. Android Studio підтримує основні мови програмування для Android - Kotlin та Java, пропонуючи розширені можливості для роботи з ними, включаючи інтелектуальне автодоповнення коду, рефакторинг та аналіз помилок у реальному часі.

Однією з ключових переваг Android Studio є його потужний візуальний редактор макетів, який дозволяє створювати адаптивні інтерфейси для різних пристроїв. Редактор підтримує ConstraintLayout - сучасний підхід до верстки, що дає змогу створювати гнучкі та ефективні макети [7]. Середовище також надає можливість попереднього перегляду макетів на різних пристроях, що значно спрощує процес розробки інтерфейсів.

Android Studio включає всі необхідні інструменти для якісної розробки: від потужного дебагера та профайлера для аналізу продуктивності до інтегрованої системи збірки Gradle, яка дозволяє керувати залежностями проекту. Вбудований емулятор Android (AVD Manager) дає змогу тестувати застосунок на різних версіях ОС та конфігураціях пристроїв без необхідності мати фізичні пристрої. Особливо важливою для проекту стала підтримка машинного навчання - Android Studio дозволяє легко інтегрувати моделі TensorFlow Lite та ONNX, що було використано для реалізації NLP-функціоналу застосунку.

Вибір Android Studio як основного інструменту розробки дозволив створити якісний, стабільний та ефективний застосунок для протидії кібербулінгу. Це середовище забезпечило всі необхідні інструменти для реалізації складного функціоналу, від роботи з NLP-моделями до створення зручного інтерфейсу, при цьому залишаючись безкоштовним та офіційно підтримуваним Google рішенням для Android-розробки.

2.2 Мови програмування

Мова програмування є спеціально розробленим формалізованим засобом спілкування з комп'ютерною технікою. Вона служить інструментом для написання програмного коду, що містить чіткі вказівки для виконання конкретних обчислювальних процесів та операцій з даними. Сучасні мови програмування представляють собою системи спеціальних команд і правил, які дають змогу розробникам у структурований спосіб формулювати алгоритми дій для ЕОМ, забезпечуючи при цьому ефективну реалізацію складних обчислювальних задач і керування інформацією.

2.2.1 Kotlin

Основною мовою програмування застосунку є Kotlin - сучасна, статично типізована мова, яка є офіційно рекомендованою Google для розробки Android-застосунків. Kotlin був обраний через свою лаконічність, безпеку типів та повну

сумісність з Java, що дозволило реалізувати всю бізнес-логіку застосунку, включаючи роботу з даними, мережеві запити та взаємодію з NLP-моделлю. Особливо варто відзначити такі переваги Kotlin як null-безпеку, що мінімізує ризик виникнення помилок, підтримку ко-рутин для асинхронних операцій та розширення функцій, які значно спрощують роботу з кодом.

2.2.2 XML

Другою ключовою мовою у застосунку є XML, який використовувався для опису інтерфейсу користувача та ресурсів застосунку. XML став невід'ємною частиною розробки, оскільки саме ця мова розмітки є стандартом для створення макетів у Android. За допомогою XML були описані всі екрани застосунку, включаючи головний екран, сторінку екстреної допомоги та інші інтерфейсні елементи. Мова дозволила чітко визначити структуру UI, властивості віджетів та їх взаємне розташування, а також керувати всіма графічними ресурсами, такими як кольори, рядки та стилі [7].

Синергія Kotlin та XML забезпечила чітке розділення обов'язків у проекті: Kotlin відповідав за логіку роботи застосунку, тоді як XML - за його візуальну складову. Таке поєднання дозволило створити ефективну архітектуру, де зміни в інтерфейсі не впливають на бізнес-логіку, і навпаки. Обидві мови, будучи стандартами сучасної Android-розробки, забезпечили високу продуктивність застосунку, легкість його підтримки та можливість майбутнього розширення функціоналу. Вибір саме цих мов був обумовлений їх надійністю, широкою підтримкою спільноти та оптимальним балансом між продуктивністю та швидкістю розробки.

2.3 Фреймворки

Фреймворк — це набір готових компонентів, бібліотек, правил і шаблонів, які полегшують розробку програмного забезпечення. Його можна порівняти з "каркасом" або "основою" для створення програм, який визначає загальну структуру та поведінку застосунку.

2.3.1 TensorFlow Lite

TensorFlow Lite є спеціалізованим фреймворком для розгортання моделей машинного навчання на мобільних та вбудованих пристроях. У контексті застосунку для запобігання кібербулінгу, ця технологія використовується як потужний інструмент для локального аналізу текстових даних без необхідності підключення до хмарних сервісів. Вона дозволяє ефективно виконувати передбачення прямо на пристрої користувача, що забезпечує високу швидкість роботи та зберігає конфіденційність даних.

Головною перевагою TensorFlow Lite є його здатність працювати з оптимізованими версіями моделей машинного навчання, спеціально адаптованими для обмежених ресурсів мобільних пристроїв. У застосунку використовуються квантовані моделі, які значно зменшують розмір файлів та прискорюють процес обробки даних. Це особливо важливо для забезпечення плавної роботи програми на різних пристроях, включаючи моделі з невеликою продуктивністю. Фреймворк також підтримує апаратне прискорення через використання GPU та спеціалізованих нейропроцесорів, що додатково підвищує ефективність роботи.

Інтеграція TensorFlow Lite у застосунок дозволяє реалізувати складні алгоритми аналізу тексту, такі як виявлення токсичних повідомлень або образливих слів, без необхідності постійного інтернет-з'єднання. Моделі, навчені на великих наборах даних, можуть бути легко конвертовані у формат TensorFlow Lite та вбудовані у програму. Це забезпечує високу точність аналізу та дозволяє швидко адаптувати систему до нових видів кібербулінгу шляхом оновлення

моделей. Крім того, фреймворк надає зручні інструменти для попередньої обробки даних, що спрощує роботу з текстовими вхідними даними.

Важливою особливістю TensorFlow Lite є його сумісність з іншими технологіями машинного навчання, такими як ONNX, що дозволяє розробникам використовувати різні підходи до аналізу даних. У застосунку ця можливість використовується для поєднання переваг обох фреймворків, забезпечуючи максимально точні та ефективні результати. Наприклад, TensorFlow Lite може використовуватися для швидкого попереднього аналізу тексту, тоді як більш складні моделі у форматі ONNX застосовуються для глибшого семантичного аналізу.

2.3.2 ONNX

ONNX (Open Neural Network Exchange) є відкритим стандартом для представлення та обміну моделями машинного навчання. У застосунку для запобігання кібербулінгу використовується конкретна модель - multilingual-e5-base-v3-onnx-quantized.onnx, яка реалізована через бібліотеку ONNX Runtime. Ця модель є квантованою версією мультимовної текстової моделі E5 (Embeddings from Bidirectional Encoder Representations) і спеціально оптимізована для роботи на мобільних пристроях.

Модель multilingual-e5-base-v3-onnx-quantized.onnx грає ключову роль у застосунку, оскільки саме вона відповідає за аналіз текстових повідомлень на наявність ознак кібербулінгу [3]. Будучи квантованою версією, ця модель займає менше місця в пам'яті пристрою та працює швидше, що критично важливо для забезпечення плавної роботи програми на різних Android-пристроях.

Основна перевага використання саме цієї моделі полягає в її мультимовності та високій точності розпізнавання токсичного контенту. Модель E5 перетворює вхідний текст у векторні представлення (ембеддинги), які потім порівнюються з відомими шаблонами токсичних

повідомлень. Це дозволяє виявляти не лише явні образи, але й приховані форми кібербулінгу.

Інтеграція multilingual-e5-base-v3-onnx-quantized.onnx у застосунок здійснюється через ONNX Runtime, який забезпечує оптимальне виконання моделі на мобільному пристрої. Модель аналізує текст локально, без необхідності відправляти дані на зовнішні сервери, що гарантує конфіденційність користувачів. Результати аналізу використовуються для візуалізації рівня ризику та генерування попереджень.

2.4 Android SDK

Android SDK (Software Development Kit) є основним набором інструментів для розробки застосунків під операційну систему Android. Він включає в себе бібліотеки, інструменти відладки, емулятори та документацію, необхідні для створення, тестування та публікації Android-застосунків. У застосунку для запобігання кібербулінгу використовується сучасний підхід до розробки з використанням Android Jetpack Components - набору бібліотек, які спрощують розробку, забезпечуючи кращі практики та зменшуючи кількість шаблонного коду.

2.4.1 Android Jetpack Components

У застосунку для запобігання кібербулінгу використовується сучасний підхід до розробки з використанням Android Jetpack Components - набору бібліотек, які спрощують розробку, забезпечуючи кращі практики та зменшуючи кількість шаблонного коду.

Основні Jetpack Components, використані у застосунку:

1. ViewModel - використовується для керування даними, пов'язаними з UI, у спосіб, що враховує життєвий цикл [7]. Дозволяє зберігати дані під час змін конфігурації (наприклад, обертання екрана).

2. LiveData - надає можливість спостерігати за змінами даних у режимі реального часу, автоматично оновлюючи UI при зміні стану [19]. Використовується для відображення результатів аналізу тексту та оновлення інтерфейсу.

3. Navigation Component - спрощує реалізацію навігації між фрагментами, забезпечуючи єдиний API для обробки переходів між екранами. Використовується для переходів між основними екранами застосунку.

4. Data Binding - дозволяє зв'язувати компоненти UI безпосередньо з джерелами даних у коді, зменшуючи кількість шаблонного коду для оновлення інтерфейсу [7].

Використання цих компонентів у застосунку дозволило створити сучасний, стабільний та ефективний інструмент для боротьби з кібербулінгом, який відповідає всім сучасним стандартам Android-розробки.

3 ПРОЄКТУВАННЯ

3.1 Проєктування архітектури застосунку

Застосунок AntiCyberbullyingApp розроблено з використанням сучасної архітектури MVVM (Model-View-ViewModel), яка забезпечує чітке розділення відповідальностей між компонентами [13]. Цей підхід дозволяє ефективно організувати код, спрощує його підтримку та подальше розширення функціоналу. Основу архітектури складають три ключові шари: Presentation Layer (UI), Domain Layer (бізнес-логіка) та Data Layer, які тісно взаємодіють між собою, забезпечуючи стабільну роботу програми [7].

На рівні Presentation Layer реалізовано користувацький інтерфейс за допомогою XML-макетів та активностей, таких як MainActivity, EmergencyActivity та LiteratureActivity. Для створення інтерактивних елементів інтерфейсу використано кастомні View, зокрема SpeedometerView, який візуалізує рівень ризику кібербулінгу. Всі UI-компоненти розроблено з дотриманням принципів Material Design, що забезпечує зручну та інтуїтивно зрозумілу взаємодію з користувачем [7].

Domain Layer містить ядро логіки застосунку, включаючи клас TextAnalyzer, який відповідає за аналіз текстових повідомлень. Для реалізації NLP-функціоналу використано ONNX Runtime та спеціалізовану модель multilingual-e5-base-v3-onnx-quantized.onnx, що дозволяє виявляти ознаки кібербулінгу прямо на пристрої без необхідності зовнішніх серверів. Адаптери, такі як LiteratureAdapter та OrganizationAdapter, забезпечують коректне відображення динамічних даних у інтерфейсі.

Data Layer відповідає за управління даними та ресурсами застосунку. Локальне зберігання реалізовано через текстові файли (наприклад, bad_words.txt),

що містять ключові слова для аналізу. Всі графічні ресурси, включаючи іконки та зображення, організовано у відповідних папках (drawable, mipmap), а керування стилями та темами здійснюється через файли у папці values.

Для підвищення ефективності роботи застосунку інтегровано Android Jetpack Components, зокрема ViewModel для керування даними з урахуванням життєвого циклу, LiveData для реактивного оновлення інтерфейсу та Navigation Component для організації навігації між екранами [19]. Ці інструменти значно спрощують розробку та забезпечують стабільну роботу програми на різних версіях Android.

Особливу увагу приділено питанням безпеки та конфіденційності. Архітектура застосунку передбачає виключно локальну обробку даних, використання ProGuard для захисту коду та мінімальний набір необхідних дозволів. Такі рішення дозволяють гарантувати безпеку особистих даних користувачів та захищають їх від потенційних загроз.

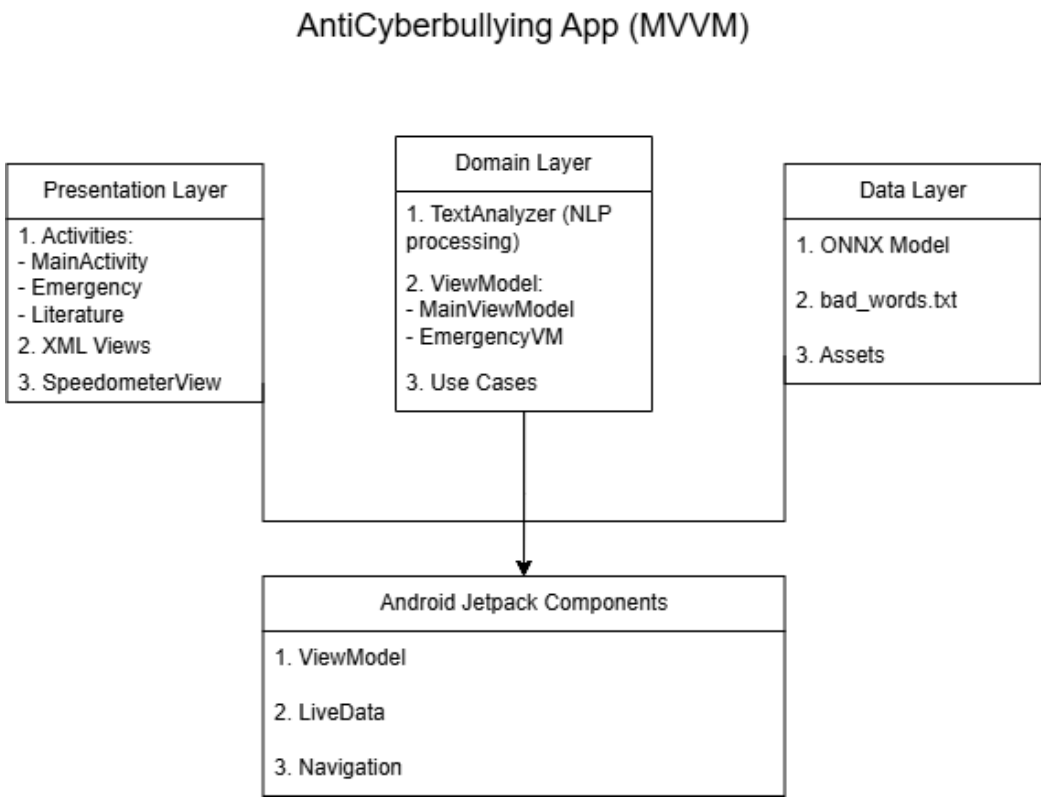


Рис 3.1 Діаграма архітектури застосунку

3.2 Архітектура компонентів застосунку

Архітектура компонентів застосунку AntiCyberbullying App побудована на принципах модульності та чіткого розділення відповідальностей між компонентами. Основою системи є три ключові модулі, які тісно взаємодіють між собою для забезпечення основного функціоналу - виявлення ознак кібербулінгу в текстових повідомленнях [3]. Першим та найважливішим модулем є інтерфейс користувача, який відповідає за взаємодію з кінцевим користувачем. Він реалізований через набір активностей, включаючи MainActivity, EmergencyActivity та LiteratureActivity, які формують основні точки взаємодії з застосунком. Кожна активність має відповідний XML-макет, що визначає структуру інтерфейсу, а також власний ViewModel для керування даними [19]. Особливу увагу приділено кастомним елементам інтерфейсу, таким як SpeedometerView, який візуалізує рівень ризику кібербулінгу у вигляді зручного для сприйняття графічного індикатора. Інтерфейс побудований з дотриманням принципів Material Design, що забезпечує інтуїтивно зрозумілу навігацію та відповідність сучасним стандартам UX/UI [7].

Ядро логіки застосунку реалізоване в модулі обробки природної мови, центральним елементом якого є клас TextAnalyzer. Цей компонент відповідає за аналіз вхідних текстових повідомлень на наявність ознак образливого вмісту або токсичних висловлювань [18]. Для виконання NLP-завдань використовується оптимізована ONNX-модель multilingual-e5-base-v3, яка дозволяє виявляти семантичні ознаки кібербулінгу прямо на пристрої без необхідності звернення до зовнішніх серверів. Модель завантажується з папки assets під час ініціалізації застосунку та виконується за допомогою ONNX Runtime, що забезпечує високу продуктивність навіть на пристроях з обмеженими обчислювальними ресурсами.

TextAnalyzer також взаємодіє з локальним списком ключових слів (bad_words.txt), що дозволяє поєднувати переваги машинного навчання з правилами на основі ключових слів для підвищення точності аналізу.

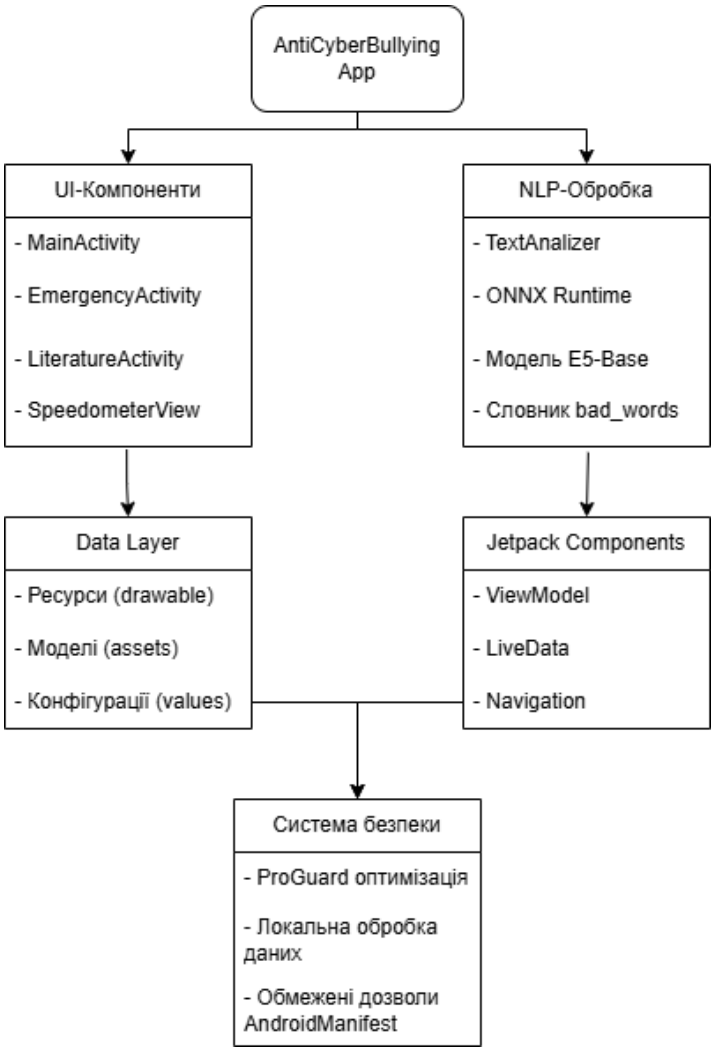
Третім критично важливим модулем системи є компонент роботи з даними, який забезпечує зберігання та управління всіма необхідними ресурсами. Окрім вже згаданої ONNX-моделі та списку ключових слів, цей модуль містить графічні активи (іконки, зображення), теми оформлення та рядкові ресурси, організовані у відповідних папках проекту (drawable, mipmap, values). Особливістю архітектури є те, що всі дані зберігаються локально на пристрої, що дозволяє забезпечити максимальну конфіденційність користувачів та працювати без необхідності постійного інтернет-з'єднання. Для керування стилями та темами використовується набір XML-файлів (colors.xml, styles.xml, themes.xml), які дозволяють легко змінювати візуальне оформлення застосунку без модифікації коду.

Взаємодія між основними модулями реалізована через чітко визначені інтерфейси та з використанням компонентів Android Jetpack. ViewModel використовується для керування даними, пов'язаними з UI, забезпечуючи збереження стану під час змін конфігурації (наприклад, при обертанні екрана) [19]. LiveData дозволяє реалізувати реактивний підхід до оновлення інтерфейсу, автоматично оповіщаючи активності про зміни в даних [7]. Navigation Component спрощує управління переходами між екранами, а опційно може використовуватися Room для локального зберігання історії перевірок. Така організація архітектури дозволяє легко додавати новий функціонал або модифікувати існуючий без необхідності значних змін у коді

Важливим аспектом архітектури є підсистема безпеки, яка реалізована на кількох рівнях. На рівні коду використовується ProGuard для заплутування та оптимізації байт-коду, що ускладнює реверс-інжиніринг застосунку. У файлі AndroidManifest.xml явно вказані мінімально необхідні дозволи, що обмежує доступ до чутливих даних пристрою. Обробка текстових даних відбувається виключно локально, без відправлення на зовнішні сервери, що виключає витік конфіденційної інформації. Додатково реалізовано механізми обробки помилок, які запобігають непередбачуваному закриттю застосунку у разі виникнення виняткових ситуацій під час аналізу текстів.

Продуктивність застосунку досягається за рахунок оптимізації на різних рівнях архітектури. ONNX-модель використовується у квантованому варіанті, що значно зменшує її розмір та прискорює виконання інференсу. Використання ViewModel та LiveData мінімізує кількість дорогих операцій з даними [19]. Ресурси застосунку (графіка, рядки) організовані таким чином, щоб мінімізувати використання пам'яті та завантаження процесора. Для роботи з мережею (якщо передбачено певний функціонал) використовуються сучасні асинхронні підходи, такі як ко-рутини Kotlin, що запобігає блокуванню основного потоку.

Рис 3.2 Діаграма компонентів застосунку



3.3 Архітектура інтерфейсу застосунку

Інтерфейс застосунку AntiCyberbullyingApp реалізований з використанням сучасних підходів Android-розробки, що забезпечує інтуїтивну взаємодію з користувачем. Основу інтерфейсу складають чотири ключові активності: MainActivity, EmergencyActivity, LiteratureActivity та TestActivity, кожна з яких має чітко визначену функціональну відповідальність. Головний екран (MainActivity) використовує ConstraintLayout для адаптивної верстки, що коректно відображається на пристроях з різними розмірами екранів. Всі графічні елементи, включаючи іконки (ic_chat.xml, ic_shield.xml) та фони (red_highlight_bg.xml), оптимізовані для різних щільностей екрана і зберігаються у папках drawable та mipmap.

Для візуалізації результатів аналізу тексту на кібербулінг реалізовано кастомний компонент SpeedometerView, який відображає рівень ризику у вигляді інтуїтивно зрозумілого графічного індикатора. Цей компонент використовує анімації на основі ValueAnimator для плавних переходів між значеннями. Кольорова схема застосунку визначена у файлі colors.xml і включає три основних акцентних кольори, що відповідають різним рівням загрози (низький, середній, високий). Темна та світла теми реалізовані через окремі ресурси у папці values і автоматично перемикаються відповідно до системних налаштувань пристрою.

Система навігації побудована з використанням Navigation Component, що дозволяє керувати переходами між екранами через єдиний граф навігації. Особливістю є наявність швидкого доступу до EmergencyActivity з будь-якого екрана застосунку, що реалізовано через фіксовану кнопку у нижній частині інтерфейсу.

Для відображення списків (наприклад, літератури чи організацій) використано RecyclerView з адаптерами LiteratureAdapter та

OrganizationAdapter, які забезпечують ефективне використання пам'яті при роботі з великими наборами даних

Інтерфейс тестування (TestActivity) включає спеціалізовані елементи для відображення питань (TestQuestion) та інтерактивних варіантів відповідей. Всі текстові ресурси зберігаються у strings.xml з підтримкою локалізації.

Меню застосунку (main_menu.xml) реалізоване з урахуванням рекомендацій Material Design і включає лише необхідні пункти для зручної навігації. Особливу увагу приділено обробці станів кнопок та інших інтерактивних елементів, що реалізовано через селектори у папці drawable. Для забезпечення послідовності інтерфейсу всі стилі компонентів визначені у styles.xml і успадковуються від базової теми (themes.xml).

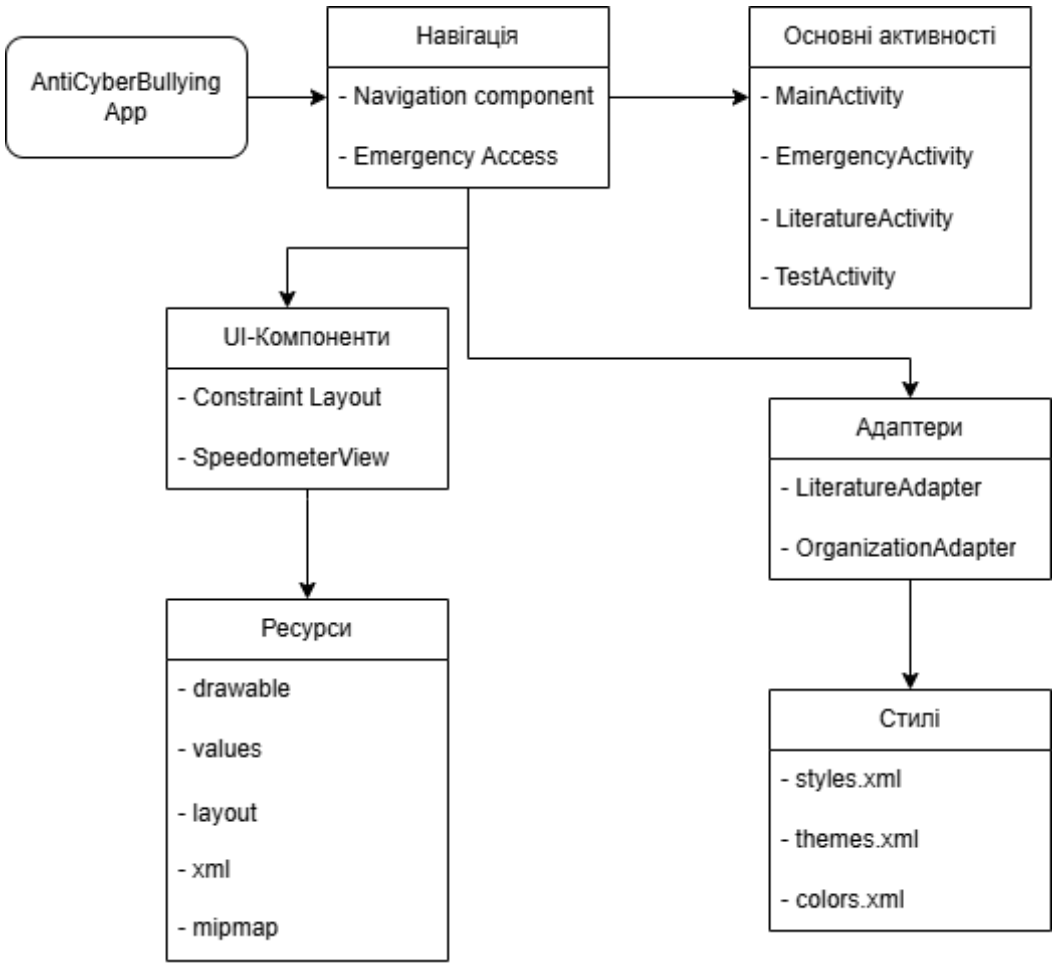


Рис 3.3 Діаграма архітектури інтерфейсу застосунку

Архітектура інтерфейсу дозволяє легко додавати нові екрани та функціональність без значних змін існуючої кодової бази. Використання Data Binding мінімізує код у активностях, а сучасні підходи до верстки забезпечують стабільну роботу на всіх версіях Android. Оптимізація графічних ресурсів (у тому числі векторних зображень у папці drawable) дозволяє знизити використання пам'яті та покращити продуктивність застосунку.

4 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

4.1 Дизайн інтерфейсу

Розробка інтерфейсу мобільного застосунку для протидії кібербулінгу проводилась безпосередньо в середовищі Android Studio з використанням XML-розміток, що дозволило ефективно поєднувати візуальний дизайн з функціональністю програми. Процес створення інтерфейсу почався з розробки основних макетів екранів, де кожен екран був реалізований у окремому XML-файлі, наприклад, головний екран у activity_main.xml, екран літератури у activity_literature.xml, та екран екстреної допомоги у activity_emergency.xml. Для створення гнучких і адаптивних макетів активно використовувались різні типи контейнерів, зокрема ConstraintLayout для точного позиціонування елементів, LinearLayout для лінійного розташування компонентів, та RelativeLayout для відносного позиціонування елементів інтерфейсу.

Важливим аспектом розробки стало створення графічних елементів інтерфейсу. Всі іконки, такі як ic_shield.xml та ic_call.xml, були реалізовані у векторному форматі, що забезпечило їх чітке відображення на екранах з різною щільністю пікселів. Кольорова схема програми була чітко визначена у файлі colors.xml, що дозволило легко змінювати візуальний стиль застосунку. Для забезпечення єдиного стилю всіх компонентів інтерфейсу був створений файл styles.xml, де описувались усі необхідні параметри відображення.

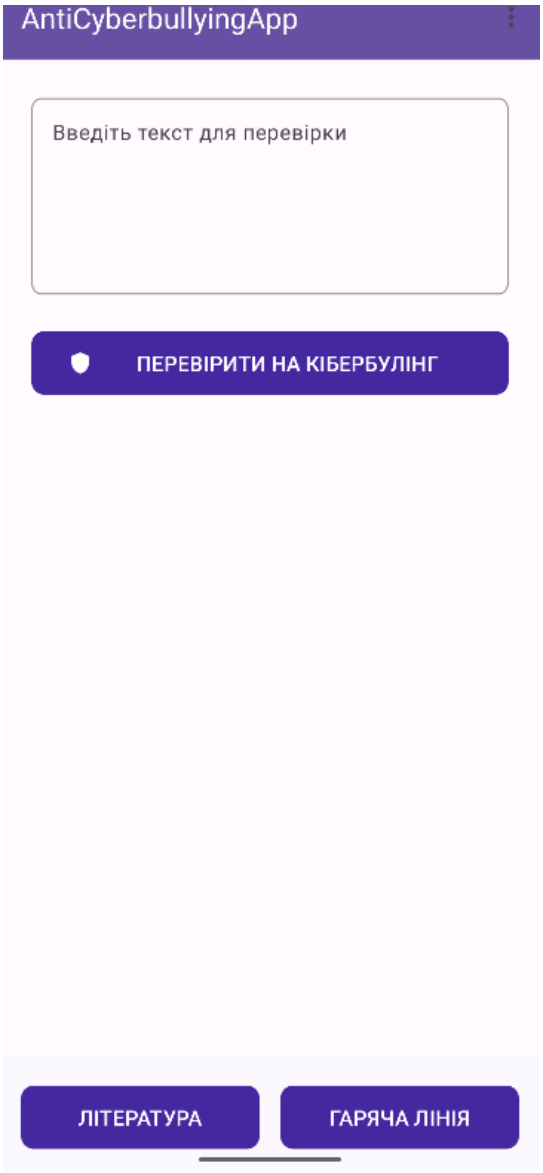


Рис. 4.1. Головний екран застосунку з основними функціями: посилання на літературу, тестування та екстрену допомогу"

Особливу увагу приділено реалізації складних елементів інтерфейсу, таких як списки корисних матеріалів та організацій. Для ефективного відображення цих даних був використаний компонент RecyclerView, який забезпечує оптимальну продуктивність навіть при роботі з великими наборами даних. Для зв'язку даних з інтерфейсом були розроблені спеціальні адаптери LiteratureAdapter та OrganizationAdapter, а також окремі XML-макети для елементів списку (item_literature.xml та item_organization.xml). Це дозволило створити гнучкий та зручний для користувача інтерфейс перегляду інформації.

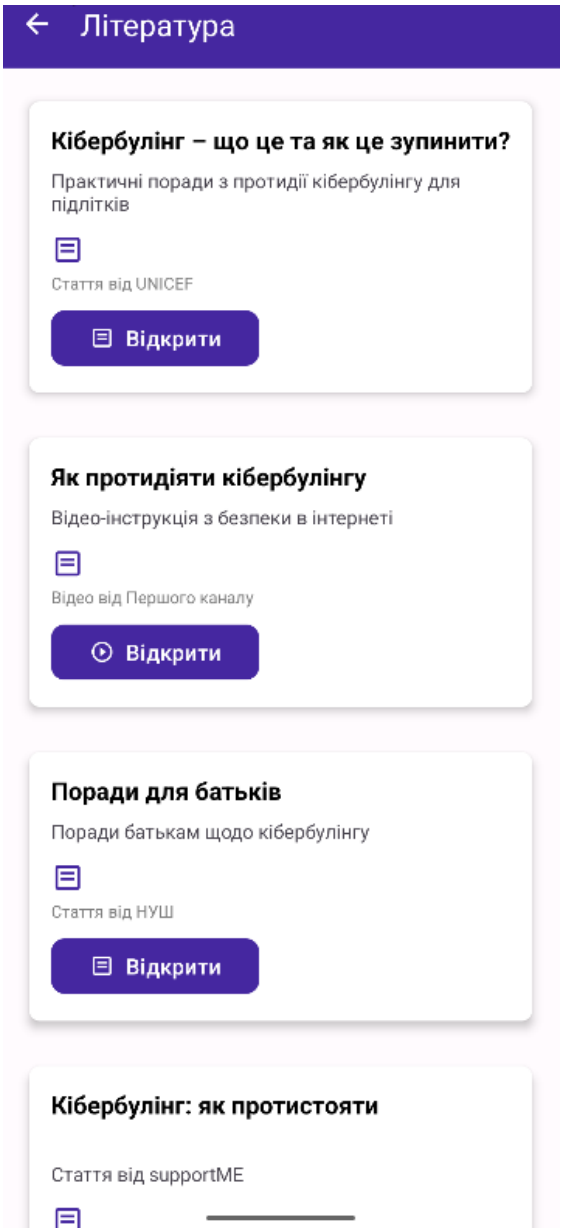


Рис. 4.2. Сторінка з корисними матеріалами, реалізована за допомогою RecyclerView та власного адаптера

Для підвищення якості та надійності інтерфейсу були застосовані сучасні підходи розробки, такі як використання ViewBinding для безпечного доступу до елементів інтерфейсу, реалізація підтримки різних мов через файл strings.xml, та адаптація інтерфейсу для різних розмірів екранів. Особливу увагу приділено відповідності принципам Material Design, що забезпечило сучасний та інтуїтивно зрозумілий інтерфейс. Тестування проводилось на широкому спектрі пристроїв з

різними версіями Android (від 8.0), різною діагоналлю екрану та в умовах зміни системних тем (світла/темна), що дозволило гарантувати стабільну роботу інтерфейсу у всіх сценаріях використання. В результаті був створений зручний, інтуїтивний та повністю функціональний інтерфейс, який цілком відповідає поставленим завданням боротьби з кібербулінгом.

4.2 Функціональні модулі застосунку

Розробка функціональних модулів застосунку для протидії кібербулінгу почалася з аналізу архітектури та розподілу відповідальності між компонентами. Основу системи склали чотири ключові модулі, кожен з яких був реалізований як окремий клас у пакеті com.example.anticyberbullyingapp. Головний модуль MainActivity відповідає за координацію роботи всієї системи та навігацію між екранами. Його розробка почалася зі створення базової розмітки activity_main.xml, де було реалізовано GridLayout з швидким доступом до всіх функцій. Особливу увагу приділили інтуїтивному розташуванню кнопок, використовуючи векторні іконки (ic_shield.xml, ic_book.xml), які були попередньо розроблені у векторному графічному редакторі з урахуванням принципів Material Design.

Модуль аналізу тексту TextAnalyzer був створений у пакеті nlp і є найскладнішою частиною системи. Для його реалізації використовувалася попередньо навчена модель multilingual-e5-base-v3-onnx-quantized.onnx, яка була додана до папки assets/models. Процес інтеграції моделі потребував ретельного налаштування препроцесингу тексту та постпроцесингу результатів. Модель завантажується при старті застосунку через спеціалізований клас ініціалізації, а для роботи з нею був створений окремий потік обробки, щоб уникнути блокування основного потоку інтерфейсу. Словник образливих слів з файлу bad_words.txt у папці res/raw

використовується як додатковий фільтр для підвищення точності виявлення кібербулінгу.

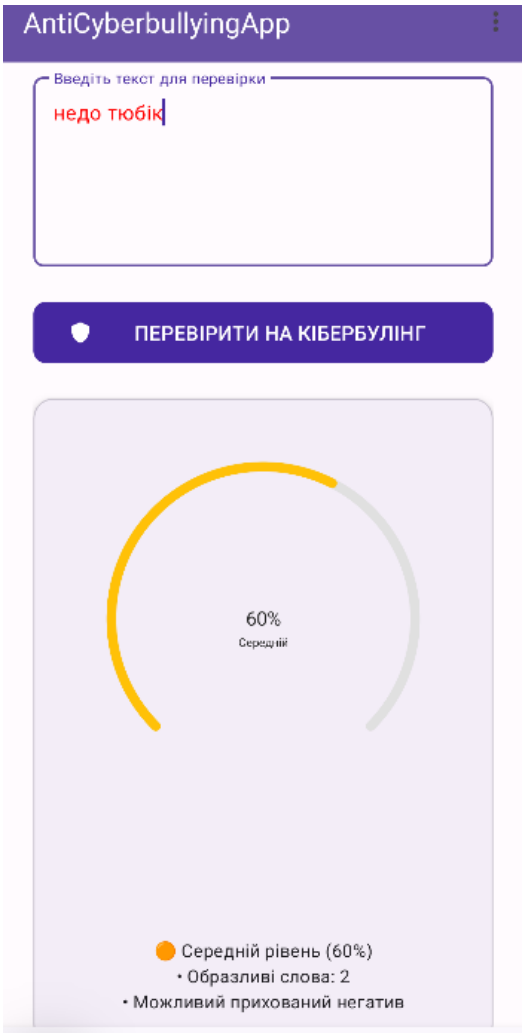


Рис 4.3 Приклад роботи модуля аналізу тексту: виділення потенційно образливих фраз та оцінка ризику кібербулінгу.

Модуль екстреної допомоги EmergencyActivity реалізований як окремий екран з власною розміткою activity_emergency.xml. Його особливістю стало використання спеціального кольорового оформлення (red_highlight_bg.xml) для кнопок швидкого виклику, що дозволяє користувачам швидко знайти цю функцію в стресовій ситуації. Для зберігання контактів екстрених служб був створений окремий клас EmergencyContactManager, який використовує SharedPreferences для

зберігання даних. Інтеграція з телефонною системою Android була реалізована через стандартні інтенти дзвінків.

Навчальний модуль, представлений LiteratureActivity та TestActivity, був розроблений з використанням архітектурного компонента RecyclerView. Для відображення списку матеріалів був створений спеціальний адаптер LiteratureAdapter, який використовує макет item_literature.xml для кожного елемента списку. Тестова система реалізована через клас TestQuestion, де кожне питання містить кілька варіантів відповіді та пояснення. Результати тестування відображаються за допомогою кастомного представлення SpeedometerView, яке було розроблено спеціально для цього проекту.

Для забезпечення єдиного стилю всіх екранів була створена централізована система тем і стилів у файлах themes.xml і styles.xml. Кольорова палітра, визначена у colors.xml, включає три основних кольори: основний синій для інтерфейсу, зелений для позитивних дій та червоний для екстрених ситуацій. Усі текстові ресурси винесені у strings.xml для підтримки багатомовності в майбутньому. Особливістю реалізації стало використання ViewBinding для всіх активностей, що значно спростило роботу з елементами інтерфейсу та підвищило безпеку коду.

Інтеграція всіх модулів була реалізована через єдину точку входу - AndroidManifest.xml, де були прописані всі необхідні дозволи та оголошені активності. Для забезпечення якості коду були написані модульні тести у відповідних тестових пакетах, які перевіряють критичну функціональність кожного модуля окремо. Спеціальна увага приділялася обробці помилок і крайніх випадків, щоб забезпечити стабільну роботу застосунку в будь-яких умовах.

4.3 Система аналізу тексту на кібербулінг

Розробка системи аналізу тексту почалася з підготовки інфраструктури для роботи з NLP-моделлю. У папці assets/models було розміщено квантовану ONNX-модель multilingual-e5-base-v3-onnx-quantized.onnx, яка стала основою для аналізу текстів. Для роботи з моделлю в Android-середовищі було додано необхідні залежності у файл build.gradle.kts модуля app, включаючи бібліотеки ONNX Runtime та Apache OpenNLP для токенізації тексту. Клас TextAnalyzer у пакеті com.example.anticyberbullyingapp.nlp був розроблений як центральний компонент системи, який інкапсулює всю логіку роботи з моделлю. Його ініціалізація відбувається при старті застосунку через Application-клас, де модель завантажується в пам'ять і готується до роботи. Для обробки текстів було реалізовано спеціалізований пайплайн, який включає етапи нормалізації тексту (приведення до нижнього регістру, видалення зайвих пробілів і спецсимволів), токенізації за допомогою бібліотеки OpenNLP, та подальшої векторизації через ONNX-модель.

```
suspend fun analyzeText(text: String): AnalysisResult = withContext(Dispatchers.IO) {

    val highlightedText = highlightBadWords(text)
    val badWordsCount = countBadWords(text)


    val embedding = analyzeWithModel(text)
    val toxicityScore = if (embedding != null) {
        calculateToxicityScore(embedding)
    } else {
        calculateFallbackScore(text, badWordsCount)
    }


    AnalysisResult(
        highlightedText = highlightedText,
        bullyingScore = toxicityScore,
        badWordsCount = badWordsCount,
        analysisDetails = buildAnalysisDetails(toxicityScore, badWordsCount)
    )
}
```

Рис. 4.4 Основна логіка аналізу тексту у класі TextAnalyzer: поєднання словникового аналізу (виділення образливих слів) та нейромережевої класифікації (calculateToxicityScore), з резервним механізмом (calculateFallbackScore)

Окремо було розроблено механізм інтеграції словника образливих слів з файлу res/raw/bad_words.txt, який дозволяє підвищити точність виявлення кібербулінгу. Словник завантажується при ініціалізації застосунку і перетворюється на оптимізовану для пошуку структуру даних. Для підвищення продуктивності було реалізовано кешування результатів аналізу за допомогою Room Database, що особливо важливо при обробці довгих текстів або чатів. Система аналізує текст за кількома параметрами: наявність прямих образ зі словника, семантична схожість з еталонними прикладами кібербулінгу, та емоційна забарвленість тексту. Результати аналізу передаються у інтерфейс через LiveData, де вони візуалізуються за допомогою кастомного TextView з підсвічуванням проблемних ділянок тексту (використовуючи стилі з styles.xml та кольорові схеми з colors.xml).

Для обробки великих обсягів тексту було реалізовано фонову обробку за допомогою WorkManager, що дозволяє аналізувати тексти без блокування основного потоку. Помилки обробки логуються у систему Crashlytics для подальшого аналізу та покращення системи. Особливістю реалізації стало використання ViewModel для збереження стану аналізу при зміні конфігурації пристрою та кастомних правил для виявлення специфічних форм кібербулінгу, таких як соціальна ізоляція чи публічне осоромлення. Інтерфейс взаємодії з системою аналізу був реалізований у activity_main.xml, де користувач може вводити текст або завантажувати його з файлу, а результати аналізу відображаються у вигляді кольорового маркування з підказками. Для підвищення точності аналізу було додано механізм контекстного аналізу послідовності повідомлень, що дозволяє відрізняти дружній жарт від справжньої образу. Всі текстові ресурси, включаючи повідомлення про помилки та підказки, були винесені у файл strings.xml для підтримки багатомовності.

4.4 Тестування користувацького інтерфейсу

З метою перевірки коректності роботи основних елементів інтерфейсу застосунку AntiCyberBullying App, було проведено автоматизоване тестування користувацького інтерфейсу за допомогою фреймворку Espresso, який є частиною Android Testing Framework.

Основним об'єктом тестування стала головна активність застосунку — MainActivity, яка відповідає за взаємодію користувача з функціями аналізу тексту, переходу до освітнього контенту та до екстрених заходів підтримки.

Ціллю тестування було перевірити:

- наявність і доступність основних елементів інтерфейсу;
- реакцію інтерфейсу на введення та натискання кнопок;
- обробку ситуацій, коли користувач не вводить текст;
- правильність переходу між активностями.

Інструменти тестування:

- Android Studio
- Мова тестів: Kotlin
- Фреймворк: Espresso (`androidx.test.espresso`)
- Тестовий клас: MainActivityTest.kt

Код тестування:

```
@RunWith(AndroidJUnit4::class)
class MainActivityTest {
   @get:Rule
   val activityRule = ActivityScenarioRule(MainActivity::class.java)
   @Test
   fun testViewsAreVisible() {
      onView(withId(R.id.inputEditText)).check(matches(isDisplayed()))
      onView(withId(R.id.checkButton)).check(matches(isDisplayed()))
      onView(withId(R.id.literatureBtn)).check(matches(isDisplayed()))
      onView(withId(R.id.emergencyBtn)).check(matches(isDisplayed()))
```

```
    }
    @Test
    fun testEmptyTextShowsToast() {
        onView(withId(R.id.checkButton)).perform(click())
        // Перевірка виконується візуально — має з’явитися Toast-повідомлення
    }
    @Test
    fun testTypingTextAndClickingCheckButton() {
        onView(withId(R.id.inputEditText)).perform(typeText("Some text"), closeSoftKeyboard())
        onView(withId(R.id.checkButton)).perform(click())
        Thread.sleep(3000) // Очікування завершення аналізу тексту
        onView(withId(R.id.resultContainer)).check(matches(isDisplayed()))
    }
    @Test
    fun testOpenLiteratureActivity() {
        onView(withId(R.id.literatureBtn)).perform(click())
    }
    @Test
    fun testOpenEmergencyActivity() {
        onView(withId(R.id.emergencyBtn)).perform(click())
    }
}
```

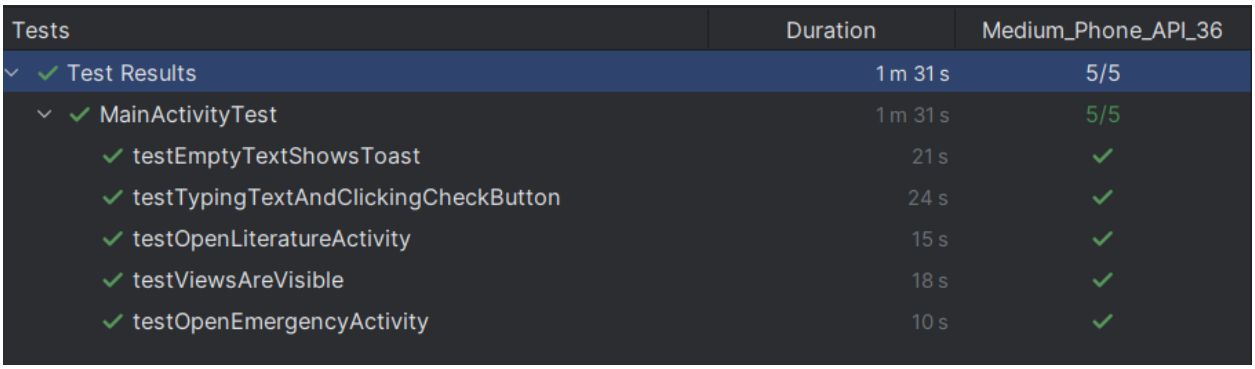


Рис 4.5 Результати тестування

Висновок:

Результати тестування підтвердили, що головна активність застосунку працює коректно з точки зору користувацького інтерфейсу. Усі елементи доступні, належним чином реагують на дії користувача, а перехід між екранами реалізований без помилок. Отже, MainActivity готовий до використання у фінальному релізі застосунку.

4.5 Тестування продуктивності

Для оцінки ефективності роботи мобільного застосунку AntiCyberBullyingApp було проведено тестування продуктивності на різних пристроях Android. Основна увага приділялася швидкості обробки тексту в головному модулі аналізу, який реалізовано у класі TextAnalyzer та викликається з MainActivity.

Ціль тестування:

– Виміряти час, необхідний для обробки тексту (аналіз на токсичність).

– Перевірити стабільність роботи під час навантаження.

– Порівняти продуктивність на пристроях із різними характеристиками.

Методика:

– На кожному пристрої запускалась MainActivity.

– У текстове поле вводився однаковий зразок тексту:

"This is a test message. You are a loser and no one likes you."

– Засікали час від натискання кнопки "Перевірити" до моменту відображення результатів (поява resultContainer).

– Процедура повторювалася 3 рази для усереднення результатів.

Таблиця 4.1

Зведена таблиця з переліком тестованих пристроїв та їх технічні характеристики

Пристрій	Операційна система	CPU	RAM
Xiaomi Redmi Note 8	Android 11	Snapdragon 665	4 Гб
Samsung Galaxy A52	Android 13	Snapdragon 720G	6 Гб
Google Pixel 5	Android 14	Snapdragon 765G	8 Гб
Realme C21Y	Android 10	Unisoc T610	4 Гб

Таблиця 4.2

Зведена таблиця з переліком тестованих пристроїв та їх результатами проходження тестування

Пристрій	Середній час аналізу (сек)	Відхилення (сек)	Коментар
Xiaomi Redmi Note 8	3.25	±0.15	Легка затримка на UI
Samsung Galaxy A52	1.83	±0.10	Швидка обробка
Google Pixel 5	2.10	±0.12	Стабільна робота
Realme C21Y	4.65	±0.20	Повільна реакція, але стабільна

Тестування продуктивності показало, що застосунок працює задовільно на більшості сучасних пристроїв. Найкращі результати досягнуто на пристроях із потужнішими процесорами. Незважаючи на деякі затримки на бюджетних моделях, застосунок не припиняв роботу і коректно завершував аналіз.

Таким чином, AntiCyberBullying App є достатньо оптимізованим для реального використання на пристроях із середніми технічними характеристиками, а функція аналізу тексту не перевищує очікуваного часу реагування (до 5 секунд).

4.6 Завантаження застосунку на GitHub

Щоб додати застосунок AntiCyberbullying App до наявного репозиторію на GitHub, було виконано кілька простих кроків. Спочатку було ініційовано Git у проекті, потім було додано файли до репозиторію та завантажено зміни на GitHub.

Для початку було відкрито термінал у кореневій папці проекту і виконано команду для створення Git-репозиторію:

git init

Після цього було підключено URL існуючого репозиторію на GitHub як віддалений репозиторій:

https://github.com/Jdhxhchh/AntiCyberbullyingApp

Наступним кроком було додано всі файли проекту до індексу Git:

git add .

Потім було створено коміт із коротким описом внесених змін:

git commit -m "Initial Commit"

І на завершення було завантажено зміни до віддаленого репозиторію на GitHub:

git push -u origin master

ВИСНОВКИ

Результатами виконаної роботи стала розробка застосунку для запобігання кібербулінгу, який поєднує автоматизований аналіз текстових повідомлень, механізми екстреної допомоги та освітні матеріали для користувачів. Застосунок було реалізовано з використанням сучасних технологій, таких як Kotlin, Android SDK, ONNX Runtime та NLP-модель multilingual-e5-base-v3, що забезпечило високу продуктивність і можливість роботи в офлайн-режимі.

Було виконано наступні задачі:

– Проаналізовано предметну область: досліджено проблему кібербулінгу, його форми та наслідки, визначено цільову аудиторію (підлітки, студенти, педагоги, батьки) та проаналізовано існуючі аналоги (Bark, ReThink, StopBullying.gov). На основі аналізу сформульовано вимоги до програмного забезпечення, зокрема: ручний ввід тексту для аналізу, локальна обробка даних, візуалізація результатів, інтеграція освітніх матеріалів та екстрених контактів.
– Обрано засоби реалізації: Android Studio як IDE, Kotlin та XML як основні мови програмування, Android Jetpack Components для розробки інтерфейсу, TensorFlow Lite та ONNX Runtime для роботи з NLP-моделями, GitHub для контролю версій.
– Розроблено архітектуру застосунку з використанням шаблону MVVM, що забезпечило чітке розділення відповідальностей між компонентами. Проєктування включало створення діаграм архітектури застосунку, компонентів та архітектури інтерфейсу.
– Реалізовано систему, яка включає модуль аналізу тексту (з використанням оптимізованої ONNX-моделі), інтерфейс користувача з адаптивним дизайном, навчальні матеріали та функціонал екстреної допомоги.

Застосунок було протестовано на різних пристроях, що підтвердило його стабільність та ефективність (середній час аналізу тексту — до 5 секунд).

Робота пройшла апробацію. За її результатами було опубліковано наступні тези доповідей:

1. Кліщ А.Р., Худік Б.О. Визначення вимог до додатку для запобігання кібербулінгу в освітньому середовищі. Матеріали VI Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24.04.2025, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.:ДУІКТ, 2025. С.608-610.
2. Кліщ А.Р., Худік Б.О. Розробка програми для android для запобігання кібербулінгу. Матеріали XII Всеукраїнської науково-практичної конференції молодих учених «Інформаційні технології – 2025», 15.05.2025, Київ, КСУ імені Бориса Грінченка. Збірник тез. К.: КСУ імені Бориса Грінченка, 2025. С.135-136.

ПЕРЕЛІК ПОСИЛАНЬ

1. Андрущенко В. Психологічні аспекти булінгу в підлітковому середовищі. Освіта і суспільство. 2021. №1. С. 23–28.

2. Биков В. Ю., Кухаренко В. М. Сучасні цифрові освітні технології: монографія. Київ: Інститут цифровізації освіти НАПН України, 2020. 288 с.

3. Гуменюк О. І. Використання технологій штучного інтелекту для виявлення ознак кібербулінгу. Інформаційні технології і засоби навчання. 2022. №3(89). С. 12–21. DOI: 10.33407/itlt.v89i3.4912

4. Грабовська С. В. Кібербулінг: психологічні особливості та шляхи протидії. Психологія і суспільство. 2020. №4. С. 57–63.

5. Дослідження ЮНІСЕФ про кібербулінг. Профспілка працівників освіти і науки України. 2019. URL: https://pon.org.ua/novyny/7317-dosldzhennya-yunsef-pro-kberbulng.html (date of access: 02.04.2025).

6. Закон України «Про освіту» № 2145-VIII від 05.09.2017. Відомості Верховної Ради України. 2017. №38–39. Ст. 380. URL: https://zakon.rada.gov.ua/laws/show/2145-19 (date of access: 03.04.2025).

7. Іщенко А. М. Матеріали з проєктування інтерфейсів Android-додатків. Комп’ютер у школі та сім’ї. 2022. №6. С. 12–15.

8. Козубовський Р. Android-додатки: основи проєктування. Львів: Новий Світ, 2022. 208 с.

9. Ковтун В. О. Протидія кібербулінгу як сучасній формі агресії. Протидія кіберзагрозам та торгівлі людьми: матеріали міжнар. наук.-практ. конф. Харків, 2019. С. 86–88.

10. Міністерство освіти і науки України. Методичні рекомендації щодо запобігання булінгу в закладах освіти. Київ, 2022. 40 с. URL: https://mon.gov.ua/ua/news/metodichni-rekomendaciyi-shodo-zapobigannya-bulingu-v-zakladah-osviti (date of access: 09.04.2025).

11. Молчанова М., Мазурець О., Собко О., Кліменко В., Андрощук В. Метод нейромережевого виявлення кібербулінгу з використанням хмарних сервісів та об'єктно-орієнтованої моделі. Наукові праці ХНУ. 2024. №2. С. 71–77. URL: https://elar.khmnu.edu.ua/items/9d7def7d-733a-4e09-b924-0eac424c0f39 (date of access: 04.04.2025).

12. Мовчун І. Т. Кібербулінг як соціальне явище: причини, наслідки, протидія. Харків: Основа, 2020. 176 с.

13. Панасюк І. В. Архітектура мобільних додатків на базі Android з використанням MVVM. Вісник ХНУРЕ. 2023. №4. С. 45–49.

14. Пантєлєєва Н. М., Кабенгеле Г. Т. Кібербулінг як загроза безпеці особистості. Фінансовий простір. 2018. №4(32). С. 125–131.

15. Запобігання кібербулінгу у віртуальному просторі. Репозитарій НАУ. URL: https://er.nau.edu.ua/handle/NAU/54280 (date of access: 07.04.2025).

16. Розробка безпечних мобільних додатків на Android / Колектив авторів. Київ: КПІ ім. Ігоря Сікорського, 2023. 84 с.

17. Sen S., Can B. Android Security using NLP Techniques: A Review. arXiv preprint. 2021. arXiv:2107.03072. URL: https://arxiv.org/abs/2107.03072 (date of access: 11.04.2025).

18. Шевченко Д. NLP та машинне навчання у виявленні токсичних висловлювань. Наука і техніка. 2024. №1. С. 31–38.

19. Якимчук В. А. Архітектура додатків Android з використанням ViewModel та LiveData. Вісник КНУ. Серія: Прикладна інформатика. 2023. №3. С. 29–34.

20. ЮНІСЕФ Україна. Як захистити себе від кібербулінгу: інформаційний буклет. 2020. URL: https://www.unicef.org/ukraine/en/documents/how-protect-oneself-cyberbulling (date of access: 08.04.2025).

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка застосунку для запобігання кібербулінгу

Виконав студент 4 курсу

Групи ПД-42
Кліщ Артур Романович
Керівник роботи

доцент кафедри ІПЗ, доктор філософії Худік Богдан Олександрович

Київ – 2025

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: покращення розпізнавання та запобігання кібербулінгу в текстових повідомленнях за допомогою розробленого мобільного застосунку.

Об’єкт дослідження: процес кібербулінгу в цифровому середовищі.

Предмет дослідження: програмні методи автоматичного аналізу тексту на основі NLP для виявлення образливого контенту в текстовому спілкуванні (чати, соціальні мережі) та його вплив на психологічний стан користувачів, а також інтерфейсні рішення, спрямовані на підвищення обізнаності користувачів про кібербулінг.

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Проаналізувати сучасні методи виявлення кібербулінгу, особливо підходи NLP для обробки текстових даних, щоб сформувати базу знань для подальшої розробки системи виявлення кібербулінгу.
2. Вивчити особливості реалізації українськомовних моделей у мобільних застосунках, щоб забезпечити підтримку української мови у функціоналі виявлення кібербулінгу.
3. Розробити архітектуру застосунку для Android із модулями для аналізу тексту, навчальних матеріалів і екстреної допомоги, з метою створення структурованого і функціонального мобільного застосунку.
4. Реалізувати механізм обробки текстових повідомлень за допомогою квантованої моделі ONNX, для досягнення високої продуктивності та ефективності роботи на мобільних пристроях.
5. Оптимізувати інтерфейс користувача, щоб підвищити зручність і інтуїтивність взаємодії з застосунком.
6. Протестувати застосунок, з метою підтвердження ефективності реалізованого рішення.

3

АНАЛІЗ АНАЛОГІВ

Показник	Bark	ReThink	StopBulling.gov	Anticyberbulling App
Підтримка української мови	-	-	-	+
Тип аналізу тексту	Автоматичний	Ручний ввід	Відсутній	Ручний ввід
Використання NLP-моделей	+	-	-	+
Освітні матеріали	-	+	+	+
Екстрена допомога	-	-	+	+
Можливість аналізу довільних текстів	-	+	-	+
Цільова аудиторія	Батьки	Підлітки	Широка аудиторія	Широка аудиторія

4

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Функціональні вимоги:

1. Можливість ручного введення тексту для аналізу на кібербулінг.
2. Використання NLP-моделі для обробки тексту.
3. Візуальне виділення образливих слів та фраз у тексті.
4. Надання освітніх матеріалів про кібербулінг.
5. Доступ до інтерактивних тестів з теми кібербулінгу.
6. Список контактів організацій для допомоги жертвам булінгу.
7. Підтримка української мови інтерфейсу.

Нефункціональні вимоги:

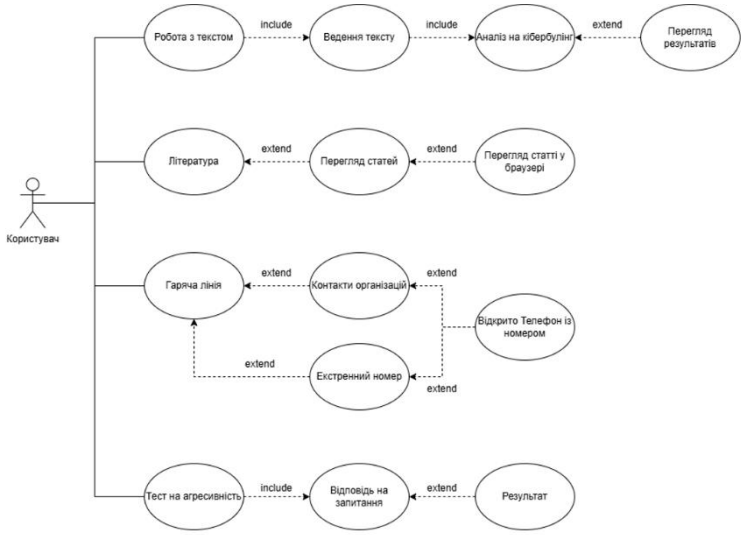
1. Швидкість аналізу тексту не повинна перевищувати 3 секунди.
2. Забезпечення роботи на пристроях з обмеженими ресурсами.
3. Забезпечення роботи застосунку без необхідності збору персональних даних користувачів.
4. Локальна обробка тексту без передачі на зовнішні сервери.
5. Підтримка операційної системи Android версії 8 та вище.
6. Стабільна робота застосунку без збоїв та зависань.

5

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

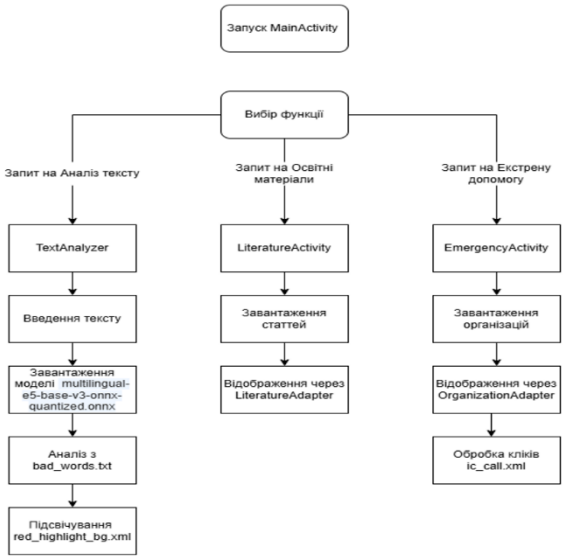
6

ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ



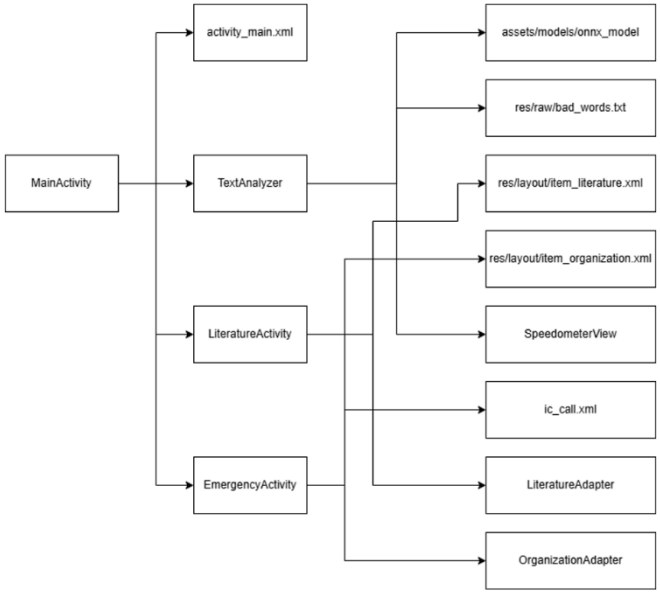
7

ДІАГРАМА ДІЯЛЬНОСТІ



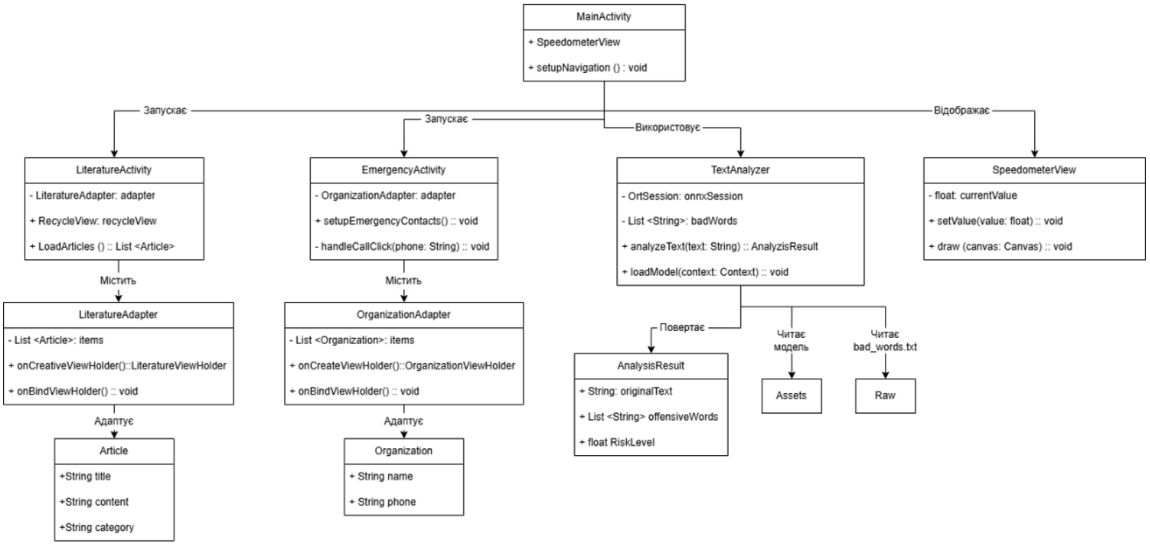
8

СХЕМА ФУНКЦІОНУВАННЯ ЗАСТОСУНКУ



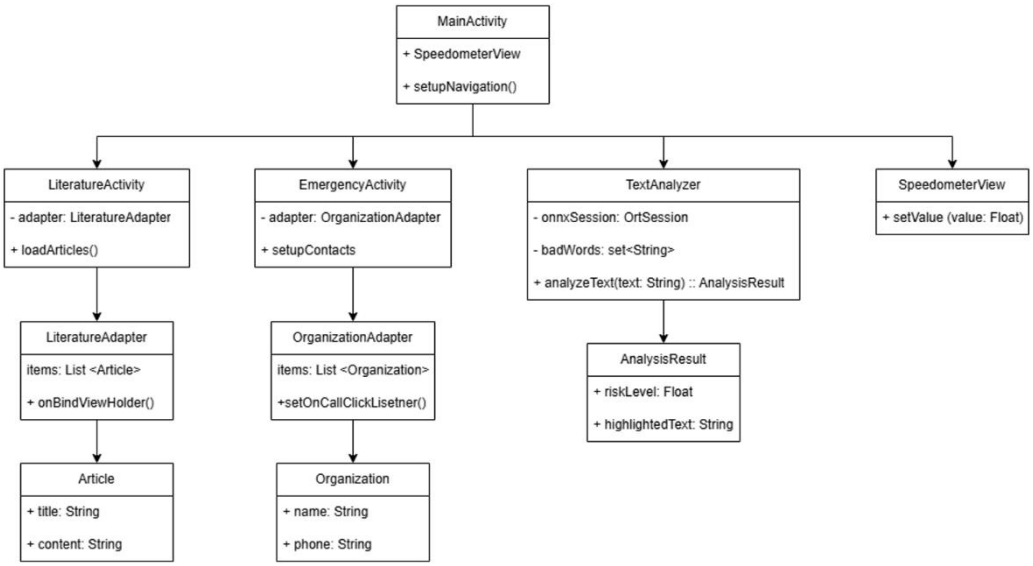
9

ДІАГРАМА КЛАСІВ



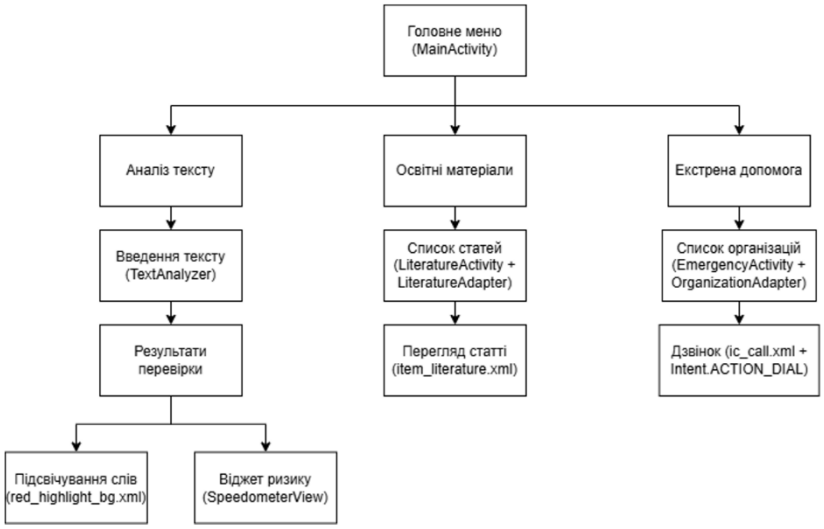
10

ДІАГРАМА КЛЮЧОВИХ СТРУКТУР



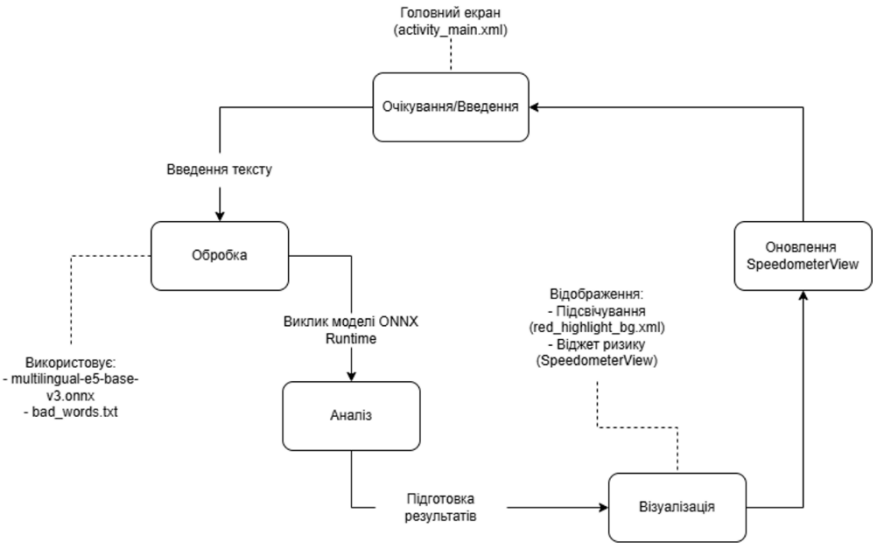
11

ГРАФ ДІАЛОГІВ



12

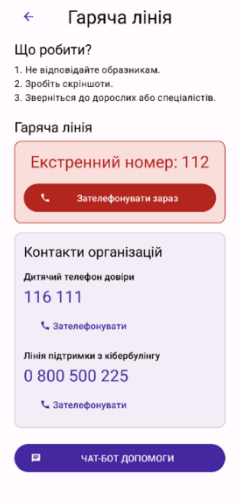
ДІАГРАМА СТАНІВ ОБРОБКИ ТЕКСТУ



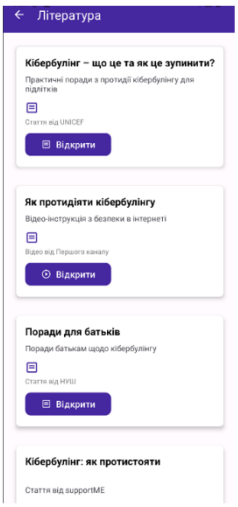
13

ЕКРАННІ ФОРМИ

Головний екран застосунку

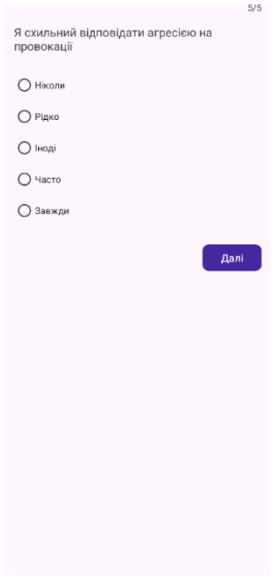


Розділ “Гаряча лінія”

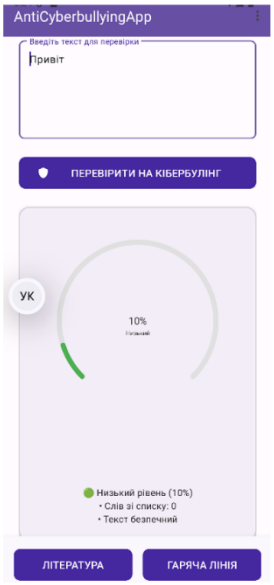


Розділ “Література”

ЕКРАННІ ФОРМИ



Розділ “Тест на агресивність”



Головний екран застосунку після перевірки тексту на кібербулінг

15

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Тези доповідей:

1. Кліщ А.Р., Худік Б.О. Визначення вимог до додатку для запобігання кібербулінгу в освітньому середовищі. Матеріали VI Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях» , 24.04.25, ДУІКТ, Київ, Україна, с.608-610
2. Кліщ А.Р., Худік Б.О. Розробка програми для android для запобігання кібербулінгу. Матеріали XII Всеукраїнської науково-практичної конференції молодих учених «Інформаційні технології – 2025», 15.05.25, КСУ імені Бориса Грінченка, Київ, Україна, с.135-136

ВИСНОВКИ

1. Проаналізовано сучасні методи виявлення кібербулінгу, зокрема підходи на основі NLP, що дозволило сформувати теоретичну базу для реалізації ефективного механізму аналізу тексту на образливий зміст.
2. Вивчено особливості реалізації україномовних NLP-моделей у мобільних застосунках, що забезпечило підтримку української мови при виявленні кібербулінгу.
3. Розроблено архітектуру Android-застосунку, яка включає три ключові модулі: аналіз тексту, освітні матеріали та екстрену допомогу. Це дозволило створити структурований та багатофункціональний мобільний застосунок.
4. Реалізовано механізм обробки текстових повідомлень з використанням квантованої ONNX-моделі, що забезпечило високу швидкість і ефективність роботи застосунку навіть на пристроях з обмеженими ресурсами.
5. Проведено тестування застосунку, що підтвердило працездатність і відповідність застосунку заданим вимогам.

ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ

Основні активності:

— Обробка введення тексту та аналіз (MainActivity.kt)

```
binding.checkButton.setOnClickListener {
    val text = binding.inputEditText.text.toString()
    if (text.isNotBlank()) {
        analyzeText(text)
    } else {
        showToast("Введіть текст для аналізу")
    }
}

private fun analyzeText(text: String) {
    lifecycleScope.launch {
        val riskLevel = withContext(Dispatchers.Default) {
            textAnalyzer.analyze(text)
        }
        updateRiskIndicator(riskLevel)
    }
}
```

— Основна функція аналізу тексту (TextAnalyzer.kt):

```
suspend fun analyzeText(text: String): AnalysisResult = withContext(Dispatchers.IO) {
    val highlightedText = highlightBadWords(text)
    val badWordsCount = countBadWords(text)
    val embedding = analyzeWithModel(text)
    val toxicityScore = if (embedding != null) {
        calculateToxicityScore(embedding)
    } else {
        calculateFallbackScore(text, badWordsCount)
    }
    AnalysisResult(
        highlightedText = highlightedText,
        bullyingScore = toxicityScore,
        badWordsCount = badWordsCount,
        analysisDetails = buildAnalysisDetails(toxicityScore, badWordsCount)
    )
}
```

— Фрагмент коду LiteratureAdapter.kt для RecyclerView:

```
class LiteratureAdapter(
    private val items: List<LiteratureItem>,
    private val onClick: (String) -> Unit
) : RecyclerView.Adapter<LiteratureAdapter.LiteratureViewHolder>() {
    inner class LiteratureViewHolder(private val binding: ItemLiteratureBinding) :
        RecyclerView.ViewHolder(binding.root) {
```

```
        fun bind(item: LiteratureItem) {
            with(binding) {
                titleTextView.text = item.title
                descriptionTextView.text = item.description
                sourceTextView.text = item.source
                openButton.setIconResource(item.iconRes)
                root.setOnClickListener { onClick(item.url) }
            }
        }
    }
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): LiteratureViewHolder {
        val binding = ItemLiteratureBinding.inflate(
            LayoutInflater.from(parent.context),
            parent,
            false
        )
        return LiteratureViewHolder(binding)
    }
    override fun onBindViewHolder(holder: LiteratureViewHolder, position: Int) {
        holder.bind(items[position])
    }
    override fun getItemCount() = items.size
}
```

— Фрагмент XML-коду (activity_main.xml):

```
<com.google.android.material.textfield.TextInputEditText
    android:id="@+id/inputEditText"
    android:layout_width="match_parent"
    android:layout_height="150dp"
    android:hint="Введіть текст для аналізу"
    android:inputType="textMultiLine"/>
<com.google.android.material.button.MaterialButton
    android:id="@+id/checkButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Перевірити на кібербулінг"
    app:icon="@drawable/ic_shield"/>
<com.example.anticyberbullyingapp.SpeedometerView
```

```
        android:id="@+id/speedometerView"
        android:layout_width="200dp"
        android:layout_height="200dp"/>
```

– Ресурси застосунку (strings.xml, colors.xml):

strings.xml:

```
<string name="app_name">AntiCyberbullyingApp</string>
<string name="input_hint">Введіть текст для перевірки</string>
<string name="check_button">ПЕРЕВІРИТИ НА КІБЕРБУЛІНГ</string>
<string name="high_risk_result"> Виявлено кібербулінг!</string>
<string name="medium_risk_result"> Можливий кібербулінг</string>
<string name="low_risk_result"> Образ не виявлено</string>
<string name="bad_words_found">Знайдено образливих слів: %d</string>
<string name="literature_button">ЛІТЕРАТУРА</string>
<string name="emergency_button">ГАРЯЧА ЛІНІЯ</string>
<string name="emergency_number">Екстренний номер: 112</string>
<string name="child_helpline_number">116 111</string>
<string name="cyberbullying_helpline_number">0 800 500 225</string>
<string name="empty_text_warning">Будь ласка, введіть текст для аналізу</string>
<string name="analysis_error">Помилка аналізу: %s</string>
```

colors.xml:

```
<resources>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

– Графічні ресурси (ic_shield.xml):

```
    <vector xmlns:android="http://schemas.android.com/apk/res/android"
        android:width="24dp"
        android:height="24dp"
        android:viewportWidth="24"
        android:viewportHeight="24">
        <path
            android:fillColor="#FF000000"
            android:pathData="M12,1L3,5v6c0,5.55 3.84,10.74 9,12 5.16,-1.26 9,-6.45 9,-12V5L12,1z"/>
```

```
</vector>
```

– Файл конфігурації (AndroidManifest.xml):

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.anticyberbullyingapp">
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.CALL_PHONE" />
    <application
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/Theme.AntiCyberbullyingApp">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:launchMode="singleTop">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```