Богодвид О.В., магістрант,
Сугоняк І.І., к.т.н., доцент
Державний університет «Житомирська політехніка»

АНАЛІЗ ПІДХОДІВ ОПТИМІЗАЦІЇ ANDROID-ЗАСТОСУНКІВ

Оптимізація продуктивності Android-застосунків є важливим аспектом розробки мобільного програмного забезпечення, що дозволяє зменшити споживання ресурсів, підвищити швидкодію та покращити загальний користувацький досвід [1]. У сучасних умовах високої конкуренції серед мобільних застосунків ефективність їх роботи значною мірою визначає успіх продукту.

Метою аналізу є дослідження сучасних підходів до оптимізації Android-застосунків, які сприяють підвищенню продуктивності, зниженню використання енергоресурсів та покращенню інтерфейсу для користувачів [2].

Інтеграція сучасних технологій дозволяє значно покращити процес оптимізації мобільних застосунків. Завдяки інструментам моніторингу та аналізу продуктивності розробники можуть отримувати детальну інформацію про використання ресурсів, що допомагає визначати проблемні місця в коді та алгоритма

Одним з ключових підходів є оптимізація рендерингу інтерфейсу за допомогою **Jetpack Compose**, що дозволяє спрощувати розробку UI, зменшувати кількість викликів до основного потоку та підвищувати ефективність відтворення анімацій та складних візуальних елементів [3]. Це значно покращує швидкодію застосунку та користувацький досвід. На відміну від традиційного підходу з використанням XML-макетів, Jetpack Compose використовує декларативний стиль програмування та більш ефективний механізм перемальовування, який оновлює лише компоненти, що змінилися, замість повного перемальовування всього екрану. Це особливо важливо для сучасних застосунків з динамічними інтерфейсами та складними анімаціями.

Процес профілювання [4] передбачає використання інструментів, таких як **Android Profiler** та **Apptim,** які дозволяють аналізувати завантаженість CPU, використання пам’яті та продуктивність графічного рендерингу. Завдяки цьому можна швидко виявляти вузькі місця та підвищувати загальну ефективність роботи застосунку.

Важливу роль у зниженні навантаження на систему відіграє застосування сучасних бібліотек, таких як **Retrofit** для оптимізації роботи з мережею та **Glide** для ефективного завантаження й кешування зображень [5]. Використання цих інструментів дозволяє мінімізувати затримки та підвищити продуктивність роботи застосунку.

Процес оптимізації потребує комплексного підходу, що включає різні аспекти, наприклад використання ефективних архітектурних патернів (MVVM, MVI), які дозволяють розділити логіку програми,
зменшити навантаження на основний потік та покращити тестованість коду. Також варто згадати про грамотне керування ресурсами, управління потоками, ефективне кешування даних та зменшення використання пам’яті за допомогою механізмів утилізації об'єктів. Застосування найкращих практик розробки, таких як Lazy Loading, зменшення кількості необов’язкових викликів до бази даних та застосування багатопоточності для розподілу навантаження [6].

Розробка ефективних Android-застосунків потребує впровадження сучасних методів оптимізації, які дозволяють зменшити навантаження на систему, покращити продуктивність та забезпечити плавну взаємодію користувачів із програмою. Завдяки інтеграції передових підходів та інструментів розробники можуть створювати швидкі, стабільні та енергоефективні застосунки, що відповідають сучасним вимогам ринку.

Список використаних джерел:

1. Кузьма К.Т. *Програмування мобільних пристроїв: навчальний посібник для дистанційного навчання*. Миколаїв: СПД Румянцева Г.В., 2021. URL: https://files.znu.edu.ua/files/Bibliobooks/Inshi73/0053792.pdf (дата звернення: 03.03.2025).
2. Google Android Developers. *Best Practices for Performance Optimization.*2024. URL: https://developer.android.com/topic/performance/ (дата звернення: 03.03.2025).
3. Google Android Developers. *Jetpack Compose: Modern Toolkit for UI Development.* 2024. URL: https://developer.android.com/jetpack/compose (дата звернення: 03.03.2025).
4. Android Studio User Guide. *Profiling with Android Profiler*. 2024. URL: https://developer.android.com/studio/profile/android-profiler (дата звернення: 03.03.2025).
5. Square, Inc. *Retrofit: A type-safe HTTP client for Android and Java.* 2024. URL: https://square.github.io/retrofit/ (дата звернення: 03.03.2025).
6. *Google* Android Developers. *Memory* Management Best Practices. 2024. URL: https://developer.android.com/topic/performance/memory (дата звернення: 03.03.2025).