Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Інфокомунікацій*
(повна назва)
Кафедра *Інформаційно-мережної інженерії*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів просування мобільних застосунків

(тема)

Виконав:

студент *2* курсу, групи *ІМІм-22-3*

Попов І.К.
(прізвище, ініціали)

Спеціальність *172 «Телекомунікації та радіотехніка»*
(код і повна назва спеціальності)

Тип програми *освітньо-наукова*
(освітньо-професійна або освітньо-наукова)

Освітня програма *«Інформаційно-мережна інженерія»*
(повна назва освітньої програми)

Керівник *доц. Омельченко А.В.*
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *Безрук В.М.*
(підпис) (прізвище, ініціали)

2024 р.

Не містить відомостей заборонених до відкритого публікування.

Студент / Попов І.К. /

Керівник / Омельченко А.В. /

Харківський національний університет радіоелектроніки

Факультет *Інфокомунікацій*

Кафедра *Інформаційно-мережної інженерії*

Рівень вищої освіти *другий (магістерський)*

Спеціальність *172 «Телекомунікації та радіотехніка»*
(код і повна назва)

Тип програми *освітньо-наукова*

Освітня програма *«Інформаційно-мережна інженерія»*
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Попову Іллі Костянтиновичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження методів просування мобільних застосунків*

затверджена наказом університету від 18 березня 2024 р. № 232 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 15 червня 2024 р.

3. Вихідні дані до роботи *Створення сучасного андроїд застосунка. Аналіз операційної системи андроїд, аналіз засобів розробки, основні етапи планування застосунків, застосування мережних технологій при проєктуванні застосунку сканування трафіку у мережі. Аналіз методів просування мобільних застосунків та необхідних інструментів, що необхідні для моніторингу та оптимізації мобільного застосунку. Створення плану просування розробленого мобільного застосунку.*

4. Перелік питань, що потрібно опрацювати в роботі
Вступ
1. Теоретичні основи створення мобільного застосунку
2. Проєктування застосунку на базі Андроїд
3. Розробка мобільного застосунку
4. Розробка плану просування мобільного застосунку
Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) *назва, мета і актуальність кваліфікаційної роботи; етапи створення мобільного застосунку; архітектура андроїд системи; структура основного модуля; Головний екран Android Studio; основна архітектура застосунка; розробка головного мережного модулю та реалізація VPN; розбір основного пакету VPN і класів в ньому; клас VpnServiceImpl.java; розробка екранів, користувальницький інтерфейс; головний екран з позначеннями; створення XML уявлення; можливості застосунка. GitHub репозиторій; Firebase App Distribution репозиторій; методи просування мобільного застосунку; ключові фактори, що впливають позиції програм Google Play; інструменти моніторингу відвідуваності мобільного застосунку; план просування мобільного застосунку в Інтернеті; висновки*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	*Ознайомлення із завданням. Уточнення ТЗ.*	*18.03.24*	*виконано*
2	*Підбір літератури за темою роботи.*	*19.03-01.04.24*	*виконано*
3	*Теоретичні основи створення мобільного застосунку*	*02.04-20.04.24*	*виконано*
4	*Проєктування застосунку на базі Андроїд*	*21.04-30.05.24*	*виконано*
5	*Розробка мобільного застосунку*	*31.05-10.06.24*	*виконано*
6	*Розробка плану просування мобільного застосунку*	*11.06-13.06.24*	*виконано*
7	*Оформлення презентаційного матеріалу, підготовка до захисту в ЕК*	*14.06-18.06.24*	*виконано*

Дата видачі завдання _18_ ___березня__ 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Омельченко А.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка 89 с., 40 рис., 13 джерела, 5 додатків.

Об'єкт дослідження – мобільний застосунок.

Мета роботи – розробка та просування мобільного застосунку для сканування трафіку у мережі.

Розглянуті такі технології: Android, Java, Kotlin. Проведено тестування застосунку, проведений аналіз архітектури та оптимізації програми. Створено застосунок для моніторингу запитів і аналізу пакетів за допомогою Android Studio та Android ОС. Проаналізовано методи просування мобільних застосунків, та фактори що впливають на позиції програми. Досліджено базові інструменти моніторингу відвідуваності та збору статистики по мобільним застосункам. Створено план просування мобільного застосунку в Інтернеті.

АНДРОЇД, МЕРЕЖА, ПЕРЕХОПЛЕННЯ ПАКЕТІВ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ОПЕРАЦІЙНІ СИСТЕМИ, МОНІТОРИНГ ВІДВІДУВАНОСТІ, GOOGLE ANALYTICS.

THE ABSTRACT

Explanatory slip 89 p., 40 fig., 13 sources, 5 attach.

Object of research - mobile application.

The purpose of the work - development and promotion of a mobile application for scanning network traffic.

The following technologies are considered: Android, Java, Kotlin. Testing of the application was carried out, analysis of the architecture and optimization of the program was carried out. An application was created for monitoring requests and analyzing packages using Android Studio and Android OS. The methods of promoting mobile applications and the factors affecting the position of the program are analyzed. The basic tools for monitoring visits and collecting statistics on mobile applications have been studied. A plan for promoting the mobile application on the Internet has been created.

ANDROID, NETWORK, PACKET INTERCEPTION, SOFTWARE, OPERATING SYSTEMS, TRAFFIC MONITORING, GOOGLE ANALYTICS.

ЗМІСТ

С.

ПЕРЕЛІК СКОРОЧЕНЬ

IP	Internet Protocol	Протокол мережного рівня стека TCP/IP
OS	Operating System	Комплекс взаємопов'язаних програм, призначених для управління ресурсами комп'ютера
TCP/IP	Transmission Control Protocol	Мережева модель передачі даних
VPN	Virtual Private Network	Віртуальна приватна мережа
HTTP	HyperText Transfer Protocol	Протокол передачі гіпертексту
HTTPS	HyperText Transfer Protocol Secure	Розширення протоколу HTTP для підтримки шифрування
FTP	File Transfer Protocol	Протокол передачі файлів
UDP	User Datagram Protocol	Протокол призначених для користувача датаграм
DATAGRAM		Мінімальний блок інформації в протоколі UDP
FRAME		Мінімальна одиниця інформація, в мережі
MAC	Media Access Control	Унікальний ідентифікатор для кожного мережевого пристрою (мережевої карти)

ВСТУП

На сьогодні дуже важко представити сучасне життя без використання нової комп’ютерної техніки. Із допомогою комп'ютерів з'являються різні програми для перехоплення та аналізу трафіка, більшість з яких є досить професійні інструменти.

Раніше, напочатку 90х років, найчастіше використовувались різні засоби, щоб перехоплювати та аналізувати пакети. Також не було багато засобів від атак зловмисників. У більшості атаки було спрямовано, щоб отримати логін та пароль користувача. Тільки пізніше вдалося розробити певні засоби для захисту даних у мережі, розроблено та застосовано наступне: криптографію, алгоритми, більш складні інфраструктури, протоколи з передачі даних та багато іншого. Кібербезпека стала чи однією із найважливіших галузей у комп'ютерній індустрії.

Актуальність обраної теми обумовлюється тим, що кількість користувачів мобільними телефонами на операційних системах Android, iOS та WindowsPhone зростає з кожним днем. Люди розуміють, що за допомогою смартфона вони отримують доступ до необмеженої інформації: можуть вести бухгалтерію, розважатися, переглядаючи медіаконтент, встановлювати корисні програми та ігри, а також планувати свою відпустку. За допомогою смартфону також можна робити перехоплення трафіку, та й не тільки у локальних мережах, а й за їх межами.

У даній кваліфікаційній роботі досліджується процес розробки та просування мобільного застосунку.

Для досягнення поставленої мети необхідно вирішити низку завдань:

- розглянути роль сучасних мобільних програм у процесі комунікації з цільовою аудиторією;

- дослідити основні організаційні, економічні та маркетингові особливості розробки мобільного застосунку;

- дати обґрунтування створенню мобільного застосунку та вивчити етапи його створення;

- розробити мобільний застосунок;

- проаналізувати механізми просування мобільного застосунку для збільшення кількості його потенційних відвідувачів.

При виконанні роботи були використані емпіричні та кількісні методи: аналіз, синтез, класифікація, збирання інформації, вивчення статистичних даних та документації, порівняння.

1 ТЕОРЕТИЧНІ ОСНОВИ СТВОРЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

1.1 Аналіз ринку мобільних застосунків

У наш час спостерігається значний прогрес у розвитку мобільних технологій: варто лише відзначити, наскільки популярними стали планшетні комп'ютери та смартфони, вони практично витіснили ноутбуки. Дедалі більше користувачів надають перевагу повноцінному виходу в Інтернет через мобільні пристрої. Багато великих сайтів зосереджуються на розвитку не повної версії, а розрахованої на мобільний вигляд.

Мобільний застосунок - це компонент, що встановлюється на мобільний пристрій, що підключається до мобільного сервера і управляє інтерфейсом користувача і бізнес-логікою пристрою [3].

Сегмент ринку програм для мобільних пристроїв був повноцінно сформований у 2008 році. Новатором, який запустив нову модель поширення програм, стала компанія Apple. Через деякий час на даний ринок вийшла компанія Google, ставши для Apple серйозним конкурентом. Оператори стільникового зв'язку, які спочатку розповсюджували мобільний контент за моделлю VAS (Value Added Services — це набір додаткових послуг у мережах мобільного зв'язку, до яких належать всі послуги стільникового зв'язку, крім передачі голосу). До цих сервісів належать: MMS-повідомлення, різні контентні послуги (тематичні розсилки, заміна гудків очікування мелодіями тощо), послуги передачі даних (за технологією GPRS або 3G), LBS-послуги (визначення місцезнаходження абонента за місцезнаходженням телефону в мережі), послуги віддаленого захищеного доступу до ресурсів корпоративної мережі тощо). Вони перетворили систему своїх дій на ринку програм для мобільних пристроїв з приходом на нього нових платформ і зараз активно інтегруються в нову модель поширення контенту [4].

Сам собою ринок мобільних додатків, як і будь-який ринок, характеризується двома фундаментальними показниками: «попитом» і «пропозицією». Якщо вірити даним консалтингових агенцій, зокрема J'son & Partners Consulting, то обсяг ринку мобільних додатків до кінця 2024 року зросте на 67 % порівняно з 2023 і тенденція зростання збережеться, що

пояснюється постійно зростаючою популярністю мобільних технологій та активним їх просуванням з боку магазинів застосунків та лідерів IT індустрії. З погляду пропозиції ринок мобільних застосунків сильно сегментований на великих розробників, які отримують близько 80% всіх замовлень, внаслідок чого вони сильно перевантажені, і невеликі компанії, які отримують малу частку замовлень.

Український мобільний ринок відносно молодий, у той же час він показує стабільну динаміку зростання, що підтримується стрімким розповсюдженням мобільного Інтернету. Таким чином, для розробника існує багато можливостей зайняти прибуткову нішу.

У процесі планування розробки, впровадження та просування на мобільний ринок програми необхідно враховувати такий фактор як драйвер ринку. Це перетворюючі чинники, що впливають стан ринку під час переходу його з одного стану до іншого. Іншими словами, драйвер ринку – це причина зміни його стану [3].

Позначимо основних драйверів ринку в Україні:

- збільшення кількості замовників. На ринку мобільної розробки у 2015 році відбувся перехід від ринку пропозиції до ринку попиту. Багато замовників почали звертати увагу на мобільний сегмент, як сегмент масової комунікації;
- зростання мобільного споживання. Зростають продажі смартфонів, планшетів, зростає споживаний мобільний трафік;
- зростання мобільної реклами. Бюджети мобільної реклами закономірно зростають слідом за зростанням кількості мобільних застосунків та їх конкуренції за користувача;
- стимулювання цього ринку з боку власників платформ Google, Apple, Microsoft стимулюють розробників локальних ринків шляхом проведення конкурсів, створення більш вигідних умов співпраці. Мета власників платформ - максимальна кількість програм для своєї платформи та отримання більш конкретного сервісу для користувача.

З важливих трендів українського ринку мобільних застосунків можна назвати такі.

1. Зростання сегмента бізнес-застосунків у рітейлі. Наприклад, використання мобільних програм у категорії Lifestyle and shopping у 2014 році

збільшилося на 174% — це більше, ніж у всіх інших категоріях. У міру того, як рітейлери відкривають численні можливості мобільного каналу для свого бізнесу, зростання в сегменті шопінг-застосунків продовжиться і в Україні, і у світі.

2. Домінування кастомних рішень. Пристрої, що працюють під управлінням системи «Андроїд», вважаються одними з найбільш гнучких в конфігуруванні. Рішення від "Майкрософт" і Apple серйозно поступаються в цьому аспекті. Безперечною перевагою систем від "Гугл" перед iOS та WinPhone є можливість відносно простої заміни початкової операційної системи на альтернативне (кастомне) рішення. Кастомне рішення (прошивка) - це своєрідно змінені або модифіковані прошивки конкретним користувачем або групою користувачів.

3. Мультиканальність та інтеграція.

Мультиканальність - це маркетингова стратегія, яка надає клієнту можливість придбати товар або послугу через різні точки входу, які не взаємодіють між собою. У контексті мультиканальності завдання рітейлера — забезпечити якомога більшу кількість каналів продажів, щоб охопити максимальну аудиторію споживачів. Наприклад, рітейлер може бути представлений у великих торгових центрах, може торгувати через інтернет-магазин або мобільний застосунок [5].

Мобільні програми зручніші у використанні - і дизайн у них нерідко краще, ніж у мобільних сайтів. Інтеграція мобільних платформ з сайтом організації є одним із шляхів доступу користувача до застосунку. Такий підхід забезпечить більш високий рівень сервісу, який надає аудиторія на смартфонах або планшетах. Мобільний канал стає одним із каналів комунікації зі споживачем та інтегруються зі сторонніми сервісами: банки, стільникові оператори, державні установи та багато іншого.

У результаті зростання конкуренції покращується якість послуг, а ціни поступово знижуються. Більше того, мобільні оператори відтепер за законом зобов'язані надавати покриття у населених пунктах із кількістю населення 1000 і більше осіб для того, щоб отримати та зберегти свої ліцензії.

Згідно зі статистикою, ще десять років тому відсоток користувачів інтернетом через мобільні пристрої не перевищував 10%, а на кінець 2015 року частка користувачів мобільних пристроїв в Інтернеті перевищила 65%.

Нещодавно було проведено дослідження, звіти якого стверджують той факт, що у 2020 році покриття мобільного Інтернету покриє 90% всієї земної кулі. Відчутний стрибок популярності мобільного Інтернету, звичайно, можна пов'язати з портативністю пристроїв для з'єднання та розвитком мобільних типів мереж .

Можна виділити два види перешкод для зростання ринку мобільних застосунків – це тактичні та стратегічні. До тактичних належить загальне зниження фінансування різні аспекти діяльності підприємств. До стратегічних – відсутність у бізнесу, не пов'язаного з IT чи інформаційною сферою розуміння навіщо взагалі потрібні мобільні програми та яку можна отримати з них користь.

З погляду виконавця чинником, що перешкоджає зростанню, є гострий дефіцит кваліфікованих кадрів, що лише посилюється відсутністю фундаментальної підготовки таких фахівців вищими навчальними закладами нашої країни.

За даними за січень-березень 2023 року, аудиторія української мережі Інтернет склала 66% населення у віці 12-64 років у містах та малих населених пунктах. Загальний приріст аудиторії протягом року становив 5%. При цьому припинилося зростання аудиторії за рахунок користувачів десктопного Інтернету; більше того, порівняння динаміки 1 кварталу 2023 року з аналогічним періодом 2022 року взагалі демонструє деяке зниження кількості людей, що виходять в Інтернет саме зі стаціонарних комп'ютерів і ноутбуків.

Приріст загальної аудиторії Інтернету в Україні за минулий рік було забезпечено, переважно, з допомогою активного зростання мобільного Інтернету. Так, за рік частка ексклюзивних мобільних користувачів, які не виходять в Інтернет зі стаціонарних комп'ютерів та ноутбуків, в Україні зросла на 90%. Що дозволяє зробити висновок про перспективність входження на ринок мобільних пристроїв та ставить під сумнів доцільність розробки дзастосунків для стаціонарних персональних комп'ютерів.

1.2 Мобільний застосунок як новий канал комунікації з цільовою аудиторією

Канали комунікації представляють собою засоби, за допомогою яких

суб'єкт комунікації передає повідомлення цільовій аудиторії. Канали комунікації включають особисте спілкування, засоби інформації, зовнішню інформацію, громадські заходи. Канали комунікації можуть бути інституційно оформлені (наприклад, будь-які ЗМІ), а можуть бути і не оформлені (усні комунікації).

Канали комунікації бувають прямими, непрямими, офіційними та неофіційними, особистими та не особистими. Прямі канали комунікації дозволяють передати інформацію безпосередньо від інформатора до інформованої особи, непрямі канали комунікації здійснюють опосередковану передачу інформації, офіційні канали комунікації надають можливість здійснювати інформаційну взаємодію з офіційними органами, а неофіційні канали комунікації оперують неперевіреною, неофіційною інформацією за допомогою. Найефективнішими є особисті канали комунікації, що забезпечують взаємодію кількох людей у процесі спілкування як безпосередньо, так і телефоном, Інтернетом, шляхом листування, під час прямого телевізійного ефіру. Неособисті канали комунікації можуть бути задіяні в умовах відсутності безпосереднього зворотного зв'язку та особистого контакту [6].

Ефективність кампаній мобільного маркетингу залежить від правильно обраного формату комунікації та цільової аудиторії, активності в анонсуванні кампанії (для цього використовують переважно традиційні канали – ТБ, Інтернет тощо) та продуманої мотивації до участі в ній.

Мобільний телефон дозволяє звертатись особисто до кожного споживача. А продумана механіка інтерактивної взаємодії з потенційними покупцями за допомогою широкого арсеналу інструментів мобільного маркетингу допомагає вибудовувати стосунки з клієнтом та в перспективі робити його лояльним до бренду.

Будучи частиною системи просування бренду, мобільний маркетинг використовує найбільш особистий канал комунікації: мобільний телефон завжди під рукою, а всі надіслані повідомлення так чи інакше трапляються на очі його власнику. Крім того, останнє має можливість відповісти на них (або відмовитися від спілкування на свій розсуд).

Переваги мобільних програм як нового каналу комунікації з цільовою аудиторією складно заперечувати. Щороку кількість бажаючих підключитися

до мобільного Інтернету стрімко зростає. Найголовніший плюс, який отримує абсолютно кожен потенційний користувач – можливість безмежного доступу до всесвітнього павутиння абсолютно скрізь. Якість та територіальне покриття мобільного інтернету збільшується з кожним роком. Сьогодні мешканці навіть найвіддаленіших куточків мають можливість отримати доступ до 4G Інтернету. Більше того, для багатьох із них наявність бездротового Інтернету є єдиним способом виходу до мережі. Для підприємця, що йде в ногу з часом, – кожен користувач мобільного Інтернету є потенційним клієнтом.

Останнім часом мобільний маркетинг використовують і при реалізації концепції купівельного маркетингу (shoppermarketing): раз мобільний телефон у людини завжди з собою, у тому числі й тоді, коли він знаходиться в магазині та здійснює покупку, компанія може вигадати механіку, яка змусить споживача спілкуватися з нею у місцях продажу. Так, американська мережа Kroger запустила програму розповсюдження мобільних купонів серед власників фірмових карток. Система дозволяє покупцям вибрати зі списку ряд доступних купонів на знижку, які вони одержують на свій телефон, та додати їх на свій клієнтський рахунок. Унікальність проєкту полягає в тому, що в ньому задіяні дисконтні картки магазину. Приймаючи мобільні купони, споживач як би прив'язує до карти свій мобільний телефон.

При цьому знижка автоматично активуватиметься, коли покупець прийде в магазин і пред'явить дисконтну картку на касі.

Головним недоліком мобільних програм можна вважати те, що користувачі мобільних пристроїв втрачають можливість оцінити повною мірою зміни інтернет-сайтів. Так як всі глобальні зміни і модифікації зазвичай стосуються повного виду сайту, поки що дуже важко реалізувати щось кардинальне на мобільній платформі, до того ж даному прогресу заважають поки що недостатньо високі швидкості інтернет з'єднань для мобільних пристроїв.

Таким чином, до основних переваг мобільних застосунків як каналу комунікації з аудиторією належать такі фактори:

- клієнт сам знаходить необхідну інформацію;
- увага та залучення клієнта;
- інтерактивність взаємодії;
- потенціал прямого продажу;

- гнучкість зв'язку з клієнтом;
- широке охоплення населення.

До недоліків мобільних застосунків як каналу комунікації з аудиторією належать:

- проблеми зв'язку;
- незначні технологічні обмеження;
- слабкий контроль та система безпеки;
- складність запровадження інновацій.

Як висновок можна відзначити, що на сьогоднішній день, коли мова заходить про канали рекламної комунікації, більшість власників бізнесу роблять акцент на наступних її видах: рекламні блоки та оголошення в місцевих газетах, статті в журналах, реклама на радіо, рекламний ролик на телебаченні, банерна та контекстна реклама в Інтернеті, забуваючи при цьому про те, що технічний прогрес не стоїть на місці і до існуючих «традиційних» каналів регулярно додаються нові, які за своєю ефективністю ні в чому не поступаються, а деяких випадках і перевершують їх. До однієї з найефективніших інновацій у сфері комунікації з цільовою аудиторією належать мобільні програми.

Мобільні застосунки дають можливість встановити довгострокову комунікацію зі споживачем: після встановлення вона завжди буде присутня у мобільному телефоні споживача, що дозволить йому частіше взаємодіяти з ритейлером, отримувати доступ до інформації та контенту.

Мобільний застосунок може стати інструментом збору інформації про споживача, за його допомогою можна не тільки стежити за активністю споживача, але і вступати з ним в інтерактивний діалог.

1.3 Вибір формату мобільного застосунку

Очевидно, що перше, про що варто задуматися — це мета, заради якої створюється мобільна система для взаємодії з клієнтами, та коло завдань, які планується з її допомогою вирішити. Наприклад, оповістити клієнтів про розташування найближчих офісів або точок продажів, донести інформацію про вигідні пропозиції, забезпечити можливість придбання продуктів або послуг через мобільний додаток, налагодити канал для зворотного зв'язку з клієнтами,

сформувати спільноту клієнтів і надати їм можливість спілкуватися між собою, і так далі. Від вибору цілей і завдань сильно залежатиме і архітектура мобільної системи, і її функціонал, і зовнішній вигляд мобільного застосунку, і підходи до забезпечення інформаційної безпеки та багато іншого.

Також необхідно чітко визначити коло клієнтів, для яких створюється мобільна система, оскільки особливості поведінки її аудиторії, сприйняття нею і зовнішнього вигляду, і набору функцій, і швидкодії та інших параметрів будуть різними в різних демографічних та соціальних груп. Без урахування цих особливостей навряд чи мобільна система дасть той ефект, на який розраховували її бізнес-замовники.

Займаючись такою серйозною справою, як розробка мобільних застосунків, також доводиться враховувати факт поширеності тих чи інших операційних систем, на яких вони працюють.

Технічне здійснення ідеї починається після виявлення найголовніших завдань. Існує кілька технологій створення застосунків, кожна з яких відрізняється швидкістю роботи, адаптацією під певну платформу та ін. Популярність платформи зумовлює її вибір. У цьому випадку всі роботи можна виконати в стислий термін, підібравши фахівців не тільки для створення, але й для обслуговування сервісу. Цей фактор визначає і зручність користування, зрозумілість, простоту, а також періодичність оновлень платформи. Дизайн, швидкість роботи та взаємодію з іншими ОС мобільних пристроїв доповнюють загальні вимоги.

На сьогоднішній день створення мобільних застосунків стало одним із найприбутковіших видів бізнесу. Причина цього проста: мобільних пристроїв стає дедалі більше. З'являються нові технології, а старі невпинно вдосконалюються. У зв'язку з цим дуже актуально відзначити, більшість користувачів бажають як користуватися вже готовими застосунками, а й створювати свої. Три фірми – Apple, Google та Microsoft – є трьома «законодавцями моди» у світі мобільних пристроїв. А значить, і у світі застосунків також.

У зв'язку з цим створення мобільних програм, які зможуть успішно працювати під Windows, добре документовано і не становить особливих труднощів. Багато планшетні комп'ютери та смартфони використовують цю популярну операційну систему. Головна перевага таких застосунків – їхня

універсальність. Зазвичай застосунок для мобільного пристрою або запускається і на звичайних комп'ютерах і ноутбуках, або має «старшу» версію для несенсорних пристроїв. Головний мінус таких застосунків – визнана навіть компанією Microsoft порівняно низька продуктивність з погляду графіки і звуку [7].

Операційна система iOS – одна із закритих «екосистем», яка замислювалася як самодостатня та повністю незалежна від будь-яких зовнішніх джерел. Те саме стосується розробки мобільних застосунків під iOS. Ця концепція передбачає можливість створити застосунок, який буде повністю автономним і не потребуватиме «підтримки» сторонніх розробників [7].

Вже давно зазначено, що створення мобільних застосунків під Android стало одним із основних для всіх, хто бажає якнайшвидше заявити про себе на ринку мобільних пристроїв та застосунків для них. Адже «андроїди» поширені досить широко: під ними працюють планшети та смартфони від різних виробників, причому як найбільших світових компаній, так і невеликих фірм, які продають свою продукцію в окремо взятих країнах. Головна перевага таких застосунків – це швидкість та універсальність. Немає необхідності розробляти окремі версії під кожний пристрій чи кожну специфікацію. Регулярні оновлення, які виходять для ОС Android, уможливлюють використання нових технологій. У той же час, головним мінусом таких застосунків вважається їхня нестабільність. На жаль, на даний момент мало застосунків для «андроїда» можуть похвалитися відсутністю багів та «дір» у системі безпеки.

Враховуючи сучасні тенденції та аналізуючи сучасний IT ринок, продукцію різних виробників, можна з упевненістю сказати, що більшість моделей побудована на операційній системі Android (HTC Sensation, знаменитий Samsung Galaxy S, Sony Ericsson Xperia, Motorola Droid). На відміну від iPhone та Windwos Phone, вона має відкритий код, внаслідок чого може встановлюватися на різні моделі телефонів. Головна перевага телефонів і планшетів на базі Android - це можливість встановлювати безліч програм як інформаційного, так і розважального характеру [7].

Розробка програми для Android – це технологія, яка потребує великих зусиль та тривалого часу для розробки спеціалізованих кодів та ідеї загалом. Не варто думати, що застосунки мають виключно розважальний характер. Сьогодні технології розвиваються стрімко та важливо йти в ногу з прогресом.

Сучасна Android розробка вимагає використання спеціальних технологій та великих зусиль, чіткої та злагодженої роботи фахівців, адже програми створюються не тільки розважальні, часто – вони ділові чи інформаційні, що потребує особливо серйозного підходу до питання.

Існує кілька способів створити програму на Android. Його можна розробити з нуля, можна замовити у студії або створити, використовуючи готові конструктори. Перші два способи є витратними за часом та засобами, тоді як третій передбачає наявність спеціального програмного забезпечення. Розповсюдженням такого займається AppGlobal. Відгуки їхніх клієнтів свідчать, що компанія пропонує ліцензії до спеціального конструктора, який дозволяє суттєво полегшити та прискорити процес розробки програми для бізнесу. Завдяки можливостям платформи, кожен може відкрити власну справу, присвячену створенню програм на замовлення. Таким чином, існує можливість пропонувати підприємцям та компаніям зі свого регіону та за його межами якісні мобільні програми без додаткового найму програмістів або вивчення основ створення програм [8].

У підсумку коротко перерахуємо основні переваги мобільної платформи Android, якій надалі буде віддано перевагу:

1) відкритість ОС Android. Android є повністю відкритою ОС, що дозволяє вільно вести розробку. З практичного боку це забезпечує більшу доступність різних застосунків та ігор. На відміну від продуктів, які пропонуються в AppStore, вони можуть поширюватися повсюдно через мережі, Google і контент-провайдерів;

2) інтегрованість із онлайн-сервісами Google. Вибравши ОС Android, ви отримуєте повну підтримку таких сервісів, як Gmail, Документи Google, Календар, Карти і т. д. Надалі компанія Google планує покращувати та розширювати набір своїх сервісів;

3) Android Market - чудова база програмного забезпечення. Незважаючи на те, що база програмного забезпечення AppStore є найбільшою та впорядкованою, незабаром ситуація може різко змінитися на користь Android Market. Ця перспектива стає можливою завдяки більш простому та толерантному розміщенню софту на Android Market, а також більш простому алгоритму встановлення програм;

4) підтримка різних форматів. Більшість користувачів розчарувалися в

мобільній продукції Apple тільки через низку обмежень, що вводяться компанією. Android надає можливість підтримки великої кількості форматів, що, безумовно, відіграє велику роль у збільшенні шанувальників цієї операційної системи;

5) можливість підтримки Flash. Сьогодні багато веб-сайтів використовують Flash, і до цього потрібно пристосовуватись. Компанія Apple відмовилася від ідеї підтримки цієї технології, Android же, у свою чергу, розробив версію Flash-плеєра;

6) віджети для швидкого доступу до функцій пристрою. В Android OS активно використовуються такі віджети, як Twitter Widget, People Widget, Messages Widget. Використання віджетів надає більшу функціональність на відміну від іконок з літерним позначенням на робочому столі в iOS;

7) можливість зберігання персональних даних у мережі Інтернет. Зберігання більшості персональних даних можна організувати в Інтернеті, що позбавить користувачів від частої синхронізації мобільного пристрою з комп'ютером. Це, у свою чергу, може стати в нагоді при втраті самого пристрою або локальних даних з нього;

8) спрощене оновлення системи. Для того щоб оновити прошивку пристрою, немає необхідності синхронізувати девайс з комп'ютером. Оновлення тепер завантажуються безпосередньо на пристрій;

9) невисока ціна на пристрої з ОС Android. Незважаючи на те, що платформа існує порівняно недавно, на ринку вже сформувався широкий вибір економічних та функціональних пристроїв за помірними цінами;

10) використання великої кількості Android-девайсів. На платформі Android з кожним днем з'являється все більше і більше гаджетів, включаючи планшети, електронні книги та нетбуки. У майбутньому це сприятиме розширенню функціоналу та гнучкості даної операційної системи;

11) широкий вибір моделей пристроїв у різній ціновій категорії. Компанія Apple постачає на ринок досить обмежену низку моделей пристроїв. Пристрої на платформі Android представлені як дуже дешевими, так і досить дорогими моделями, що, безумовно, є важливою перевагою;

12) наявність реальної чи віртуальної клавіатури. На відміну від iPhone, у пристроях на платформі Android існує альтернатива вибору смартфона лише з екранною або фізичною повною клавіатурою QWERTY;

13) можливість коригування інтерфейсу Android. Виробники, орієнтуючись на свій смак, можуть впровадити в Android різні версії інтерфейсів користувача. Наприклад, HTC Sense на Hero та Tattoo;

14) можливість максимально настроїти пристрій під себе. В Android OS доступний широкий ряд можливостей щодо створення та налаштування кількох робочих столів. Також можна настроїти реакцію пристрою на різні події [8].

Виходячи з перерахованих переваг платформи Android, можна зробити висновок про те, що найбільш оптимальною платформою для розробки мобільного застосунку у співвідношенні ціна/ефективність є саме вона.

Перейдемо до практичних аспектів розробки мобільного застосунку.

1.4 Організаційні, економічні та маркетингові аспекти створення мобільного застосунку

Маркетинг за допомогою мобільних застосунків є новим каналом комунікації, але таким що швидко розвивається, в порівнянні з іншими каналами, такими як мережа Інтернет, ТБ, зовнішня реклама, і він істотно дешевше на сьогоднішній день. При цьому мобільний маркетинг вже досяг тієї точки розвитку, після якої він може конкурувати з будь-яким іншим каналом за основним критерієм - обсягом аудиторії.

Існує дві основні причини розвитку власного бізнесу у напрямку ринку мобільних застосунків:

- по-перше, вже сьогодні з магазинів, що пропонують застосунки та ігри, було викуплено понад мільйон розробок. Тобто є можливість розмістити продукт чи послугу на ресурсах AppStore, Android Market та Windows Marketplace. При цьому люди, які були зацікавлені товаром або залишилися задоволеними якістю наданої послуги, ще й порекомендують його іншим;

- по-друге - розробка застосунків для мобільних пристроїв – це крок уперед у розвитку та просуванні фірми. Якщо людям сподобається програма, вони звернуть увагу і на інші послуги та продукти в арсеналі компанії. Плюс до цього – популярність. Якщо зробити рекламу мобільному застосунку, надалі, вона вже сама по собі найкраще розповідатиме про діяльність фірми.

Сьогодні вже не важливо, чи ви хочете створити застосунок для Android або іншої платформи, – користувачі хочуть доступний контент, на який не

потрібно витрачатися. Якщо компанії важлива увага споживачів, варто задумати про те, як зробити застосунок безкоштовним.

Як загальні рекомендації для входу на український ринок мобільних застосунків можна виділити кілька пунктів:

1) популярні та перспективні категорії застосунків. Більшість мобільних застосунків та сервісів, доступних зараз на українському ринку, сконцентровані у таких сегментах: соціальні мережі, геолокація, ігри, музика, новини, контент-проєкти та транспортні послуги. У той же час спостерігається нестача муніципальних та туристичних мобільних сервісів, спостерігається «застій» у сегменті повноцінного онлайн-рітейлу, шопінгу та вузькоспеціалізованих застосунків;

2) локалізація. Вхід на український ринок майже завжди має на увазі локалізацію. Більшість користувачів взагалі не встановлює або відразу видаляє в цілому корисний, але не локалізований застосунок просто тому, що не хоче розбиратися в його малозрозумілому інтерфейсі. Соціологічне дослідження, міжнародної аналітичної компанії Common Sense Advisory, проведене в 2014 р. у низці країн, для яких англійська не є державною (у тому числі і в Україні), показало, що 87% споживачів, які не знають англійської, не купують товари і не користуються послугами англомовних веб-сайтів;

3) просування у місцевих соціальних мережах. Огляди та прес-релізи, що публікуються популярними українськомовними технологічними блогами та веб-сайтами, зазвичай дають результат у вигляді швидкого приросту завантажень, проте не можуть стати джерелом стабільного постійного трафіку. Це непогана тактика для первинного запуску програми, але згодом необхідно стимулювати інтерес аудиторії у вигляді соціальних мереж. Місцеві соціальні мережі пропонують хороші можливості для просування застосунків за допомогою реклами в стрічках новин, групах користувачів та відкритих форумах. Цікавий контент (смішне відео, демотиватори, корисні підказки), який може стати вірусним і дати великий відгук, для найкращого ефекту має бути українською.

Говорячи про організаційні моменти створення мобільного застосунку, варто згадати, що головний етап – це створення ескізів та побудова алгоритму роботи програми. Тобто який екран буде за тим чи іншим натисканням клавіші на дисплеї. Після того, як алгоритм створено і його повністю опрацьовано, за

ескізами готуються тестові скріншоти дисплеїв. І вже за ними програма пишеться, оптимізується.

Слід враховувати, що часто, після створення програми для мобільної платформи, її необхідно буде в майбутньому оновлювати. Найпростіший спосіб при цьому користуватися вже існуючими маркетами застосунків і через них проводити оновлення. Тому створений застосунок повинен підтримувати автооновлення.

Якісне створення мобільного застосунку також передбачає виконання численних тестів перед випуском програми у користування. Тестування дозволяє знайти всі невраховані помилки, пропущені ще на етапі побудови алгоритму, і швидко їх виправити. До речі, вартість створення мобільного застосунку на даний момент є досить низькою через високу конкуренцію на споживчому ринку. Але це не означає, що створення високоякісної багаторівневої програми вимагатиме мінімальних фінансових вкладень.

1.5 Етапи створення мобільного застосунку

Розглянемо ключові етапи створення мобільного застосунку.

Перший етап створення – це складання технічного завдання. Технічне завдання (Також - техзавдання, ТЗ) - технічний документ (специфікація), що обумовлює набір вимог до системи та затверджений як замовником/користувачем, так і виконавцем/виробником системи. Така специфікація може містити системні вимоги та вимоги до тестування [9].

Необхідно також скласти план завдань. Він має бути докладно описаний, щоб не переробляти його знову. Адже упустивши навіть незначну деталь, може порушитися вся будова мобільного застосунку. Функціонал програми – основа всієї роботи.

Другий етап розробки мобільного застосунку - це проєктування UI/UX. User Experience Design у перекладі означає «досвід взаємодії» і включає різні UX-компоненти: інформаційну архітектуру, проєктування взаємодії, графічний дизайн і контент. На даному етапі реалізуються всі роботи, описані в технічному завданні. Створюється графічна мапа взаємодії між екранами. Також на цьому етапі роботи треба визначити, яким чином працюватиме застосунок, і як проходитиме робота користувача з ним [9].

Третій етап: робота з дизайном. Дизайн створюється на основі побажань клієнта та націлений на цільову аудиторію. Спочатку опрацьовується дизайн 1-3 сторінок, який закладає всю основу програми. За побажанням клієнта може бути створено кілька стилів дизайну, оцінивши які замовник визначить найкращий варіант [9].

Четвертий етап: технологія. Фахівці зі статичного стану переводять застосунок до інтерактивної моделі. Цього можна досягти за допомогою верстки. Закінчений варіант роботи відправляється клієнту, щоб він оцінив результат роботи і висловив свою думку.

Кілька днів мобільний застосунок активно тестується. Якщо виявляються якісь помилки, то програму допрацьовують. Доробка займає приблизно половину всього часу, який було витрачено раніше. Слід зазначити, що редагування програми неминуче. Бо при його створенні не можливо передбачити всі недоліки, які можуть виникати при використанні програми.

Тестування. На різних етапах розробки програми обов'язковим є внутрішнє тестування як на симуляторах, так і на реальних пристроях. Мета тестування - переконатися, що взаємодія програми з апаратною та програмною платформою смартфонів та планшетів буде саме такою, як передбачалося на етапі прототипування [9].

Створення передрелізної версії. В результаті серії тестів та доопрацювань програми має бути отримана робоча версія програми. Саме цю версію і додають до магазину застосунків: Apple App Store, Google Play, магазину застосунків Windows Phone (залежно від того, для якої платформи ведеться розробка) або у будь-який аналогічний сервіс для дистрибуції застосунків [9].

Додавання програми до магазину. Фінальний етап роботи студії - додавання програми на рев'ю в один із зазначених вище магазинів застосунків (у випадку Componentix йдеться про App Store або Google Play) [9].

Вимоги до системи. GooglePlay (попередня назва – Android Market) – магазин програм від компанії Google, що дозволяє власникам пристроїв з операційною системою Android встановлювати та купувати різні програми.

Обліковий запис розробника, який дає можливість публікувати програми, коштує $25. Платні програми можуть публікувати розробники не з усіх країн. У GooglePlay можна знайти багато корисних та різноманітних програм. У магазині є платні та безкоштовні програми.

31 жовтня 2015 року компанія Google оголосила, що кількість застосунків досягла 700 000 застосунків, і кількість завантажень досягла 25 мільярдів. Однак одночасно з цим користувачі скаржаться, що в магазині часто містяться програми низької якості. Відповідно до цього очевидна важливість розробки якісного програмного продукту. Ключовим моментом початку розробки є пошук та аналіз аналогічних або подібних за тематикою програмних продуктів, у тому числі порівняльний. Основне завдання тут – визначити плюси та мінуси вже існуючих застосунків з погляду користувача.

Пошук аналогів та аналіз їх переваг та недоліків. Бенчмаркінг конкурентоспроможності – це вимірювання характеристик підприємства, дослідження специфічних продуктів, можливостей процесу чи адміністративних методів та зіставлення їх із характеристиками конкурентів. У термінах застосунків для мобільних телефонів, бенчмаркінг є оцінкою за десятибальною шкалою <1,..,10>, отриману при тестуванні функціональності застосунку на заявлені вимоги [9].

2 ПРОЄКТУВАННЯ ЗАСТОСУНКУ НА БАЗІ АНДРОЇД

Програми для андроїду містять велику кількість технологій і усі можуть збільшити якість та досвід при використанні застосунку і системи. Крім того сама операційна система Андроїд за доволі маленький період розвинулась до високого рівня та усунула більшість вразливостей, проте деякі ще з’являються, і це те, що допомагає андроїд системі прогресувати.

2.1 Архітектура андроїд системи

Напочатку визначимо деякі базові принципи функціонування та роботи андроїд додатку:

– операційна система Андроїд – що розрахована на велику кількість користувачів, в основі якої ядро Linux, у цій системі кожен застосунок як окремий процес та навіть інші застосунки не зможуть вплинути на його працездатність;

– у кожного процесу власна віртуальна машина, отже програмний код функціонує окремо від інших подібних програм;

– кожна програма функціонує у особистому процесі Linux. Андроїд зможе запустити процес, якщо необхідно виконати запит програми, а далі відключити даний процес, якщо він вже не потрібен чи коли системі необхідно більше пам'яті чи потрібно відновити пам'ять задля іншої програми [4,5].

На рис. 2.1 зображено запуск операційної системи Андроїд та запуск самої програми із головного екрана.

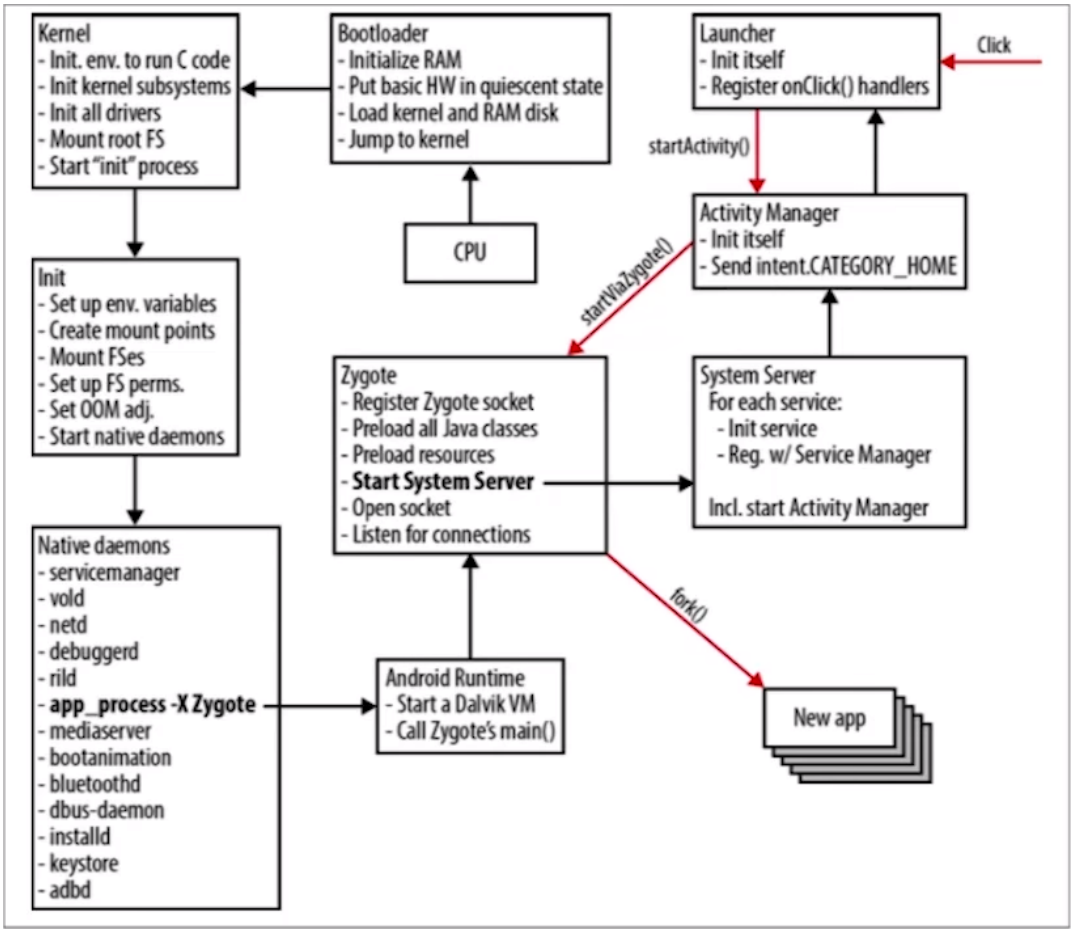

Рисунок 2.1 – Запуск операційної системи Андроїд та запуск самої програми із головного екрана

2.2 Архітектурна частина андроїд застосунку

Компоненти даної програми є одним із необхідних інструментів у системі Андроїд. Кожен із них як точка входу, крізь яку система чи користувач має змогу підключитись у застосунок. Це можуть бути такі компоненти:

- Activity, по типу одного вікна керування, має відобразити у собі дрібніші частинки, так звані фрагменти (англ. Fragment);
- Service вважається будь-якою роботою, що виконано на задньому плані, як приклад завантаження даних із мереж, завантаження файлів та інше;
- широкомовні певні повідомлення;
- поставник контенту.

Щоб спростити та зменшити код використана стандартна практика, розподіл структур у пакети та з використанням однієї активіті (англ. Activity), що зображено на рис 2.2 [6,7].

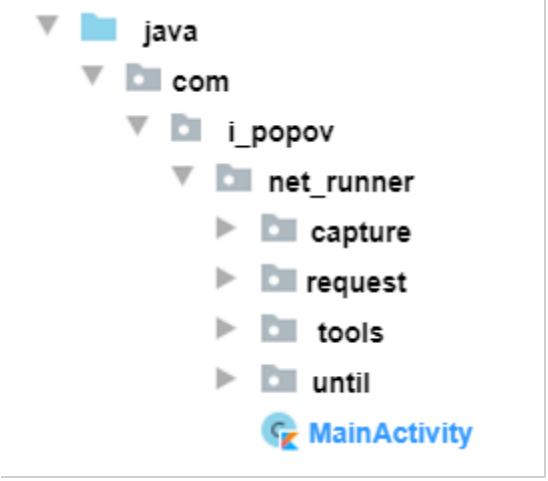

Рисунок 2.2 – Структура основного модуля

З рисунку бачимо, що маємо одну активність (англ. Activity) та декілька пакетів Capture, Request, Tools, що відповідають за певні екрани, що відобразяться у вікні Activity [5]

2.3 Файли ресурсів та розмітка екрану

Застосунок Андроїд містить не тільки код і маніфест, але й папку ресурсів, це інші, незалежні від вихідного кода файли, як приклад: зображення, розмітки екрана (англ. Layout), усе, що пов'язано із візуальним представленням застосунку. Використовуючи ресурси програми можна швидко оновлювати різноматнітні характеристики застосунку незмінюючи вихідний код. Проте також важливе у контексті андроїда, застосунки, це можливість підлаштовувати усі ресурси під доволі велику кількість різного типу пристроїв під керуванням операційної системи Андроїд. А саме, у смартфоні, у якому екран 5 дюймів та версія Андроїд 10.0 чи на телефоні 3.5 дюйма та версією андроїд 5.0 застосунки будуть виглядати так само однаково. Це дуже важливо. Розмітка екрана у андроїді на даний час встановлюється у XML розмітці та використовується певний простір андроїд імен. Як приклад, розмітка для основного екрана, розмітка MainAcitivity – розмітка екрана, у якому буде знаходитись інші екрани. На рис 2.3 і рис 2.4 зображено файл розмітки головного екрану та скрин основного екрана у редакторі [4,5].

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    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/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:focusable="true"
    android:focusableInTouchMode="true"
    tools:context=".MainActivity">

    <FrameLayout
        android:id="@+id/nav_host_fragment"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:background="@color/colorBackground"
        app:layout_constraintBottom_toTopOf="@id/bottom_navigation"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottom_navigation"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="0dp"
        android:layout_marginEnd="0dp"
        app:itemIconTint="@drawable/bottom_navigation_selector"
        app:itemTextColor="@drawable/bottom_navigation_selector"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:menu="@menu/navigation" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Рисунок 2.3 – Файл розмітки головного екрану

Можна сказати, що у андроїд системі побудова інтерфейса користувача формується по ієрархії певних компонентів, тобто ієрархії відображення (View). Напочатку необхідно обрати контейнер, що буде мати у собі розмітку, він має назву View Group та може утримувати дрібніші компоненти, а саме кнопки, навігаційне меню, текст і інше. У даному випадку View Group це Constraint Layout, що дає можливість з'єднати окремі певні компоненти додаючи їх до країв чи батьківського контейнеру (Constraint Layout) або ж до країв інших View.

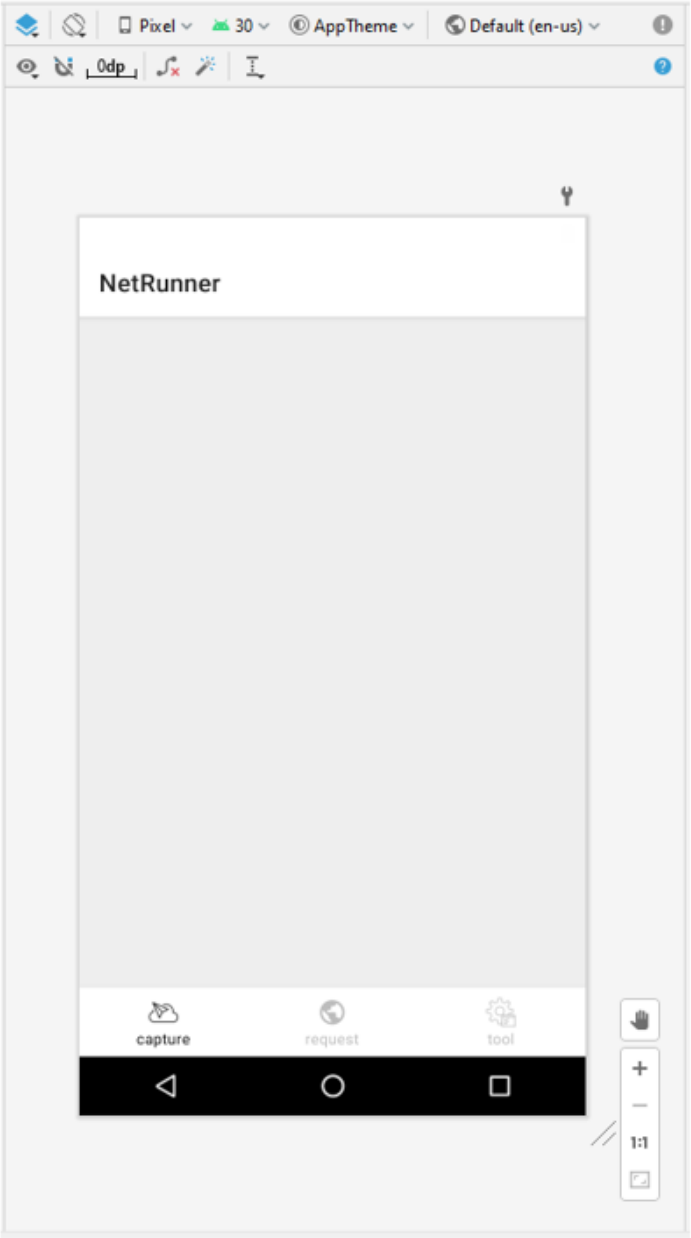

Рисунок 2.4 – Фото головного екрану у редакторі

Перша розмітка – Frame Layout, вбудована розмітка для наступних екранів, що можуть замінюватись та нижнє навігаційне меню – Bottom Navigation Menu. У кожної View маються свої певні атрибути, які необхідні, це layout_width та layout_hight, що записують ширину та розмір висоти. В даному випадку, є дві View. Також необхідно звертати увагу щодо позначення простора імен: xmlns: андроїд, xmlns: app, xmlns: tools. Це під’єднуються спеціальні певні теги, що визначають атрибути, що використано. Можна також відзначити, що на розмітку та на вигляд програми впливають також кольори, теми, стилі [6].

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- LIGHT theme -->
    <color name="colorPrimary">#FFFFFF</color>
    <color name="colorPrimaryDark">#BDBDBD</color>
    <color name="colorAccent">#000000</color>

    <color name="colorBackground">#EEEEEE</color>
    <color name="colorUnSelected">#BDBDBD</color>
    <color name="colorCursor">#263238</color>

    <color name="start">#81C784</color>
    <color name="stop">#E57373</color>
    <color name="text_color_request">#004D40</color>
    <color name="text_color_response">#BF360C</color>
    <color name="ic_launcher_new_background">#000000</color>
</resources>
```

Рисунок 2.5 – Відображення вмісту файла colors.xml

У кольорах можна побачити позначення для головної білої теми, також додано власних кольорів для деяких елементів, які призначено для інтерфейсу користувача. Далі вже стилі. Стилі та теми у андроїд дуже важливі, у стилях можна звернутись до цікавих певних темі, також є певні потрібні атрибути. Можна переписати необхідний атрибут і використовувати вже певну переписану тему для інших View [7].

Можна побачити App Theme – стиль, що має тему Day Night та декілька значень, ця тема встановлена як головна у маніфесті, застосовується до усього додатка, проте якщо є необхідність у зміні теми, то можна це зробити за допомогою нового стилю. Як приклад, є певна тема App Theme. No Action Bar, при її використанні встановлюється, що у даному дисплеї не буде верхньої панелі дій (Action bar).

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.MaterialComponents.DayNight">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>

        <item name="android:statusBarColor">@color/colorPrimary</item>
        <item name="android:windowLightStatusBar" tools:targetApi="23">true</item>

    </style>

    <style name="AppTheme.NoActionBar">
        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
    </style>

    <style name="noActionBar" parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>

        <item name="android:windowLightStatusBar" tools:targetApi="23">true</item>
        <item name="android:statusBarColor">@color/colorBackground</item>

        <item name="tabSelectedTextColor">@color/colorAccent</item>
    </style>
```

Рисунок 2.6 – Відображення вмісту файла styles.xml

2.4 Інструмент збірки застосунку gradle

Gradle – це один із найважливіших інструментів для андроїд застосунку, це засіб його збірки. На сьогодні, gradle – це дуже добрий вибір у такій досить складній процедурі. Gradle, а також спеціальний плагін Андроїду до нього дають певний гнучкий засіб компіляції і також збірки та упакування застосування чи бібліотеки Андроїд. Є безліч засобів його оптимізації, що виконано у застосунку, що розробляється, усі вони додають гарну продуктивність у швидкість збірки. На рис. 2.8 показано файл build.gradle модулю app. В ньому ключовою структурою є функція розширення: андроїд, repositories, dependencies.

```
ext {
    //compile config
    compileSdkVersion = 30
    buildToolsVersion = "30.0.0"
    minSdkVersion = 22
    targetSdkVersion = 30

    //Package names
    applicationId = "com.ninpou.netrunner"

    //version number
    versionCode = 1
    versionName = "1.0"
```

Рисунок 2.7 – Файл ext.gradle, що для позначення змінних

```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'

android {
    compileSdkVersion rootProject.ext.compileSdkVersion
    buildToolsVersion rootProject.ext.buildToolsVersion

    defaultConfig {
        minSdkVersion rootProject.ext.minSdkVersion
        targetSdkVersion rootProject.ext.targetSdkVersion
        applicationId rootProject.ext.applicationId
        versionCode rootProject.ext.versionCode
        versionName rootProject.ext.versionName

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {...}
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {jvmTarget = "1.8"}
}

repositories {mavenCentral()}

dependencies {
    implementation fileTree(dir: "libs", include: ["*.jar"])
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    testImplementation junit
    androidTestImplementation junit
    androidTestImplementation espressoCore

    implementation project(':packetCapture')
    implementation appcompat
```

Рисунок 2.8 – Файл build.gradle необхідний для збірки проєкта

3 РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ

Напочатку буде описано інструменти, які надані для андроїд розробника, це і Android Studio, і також документація, що знаходиться на офіційному сайті developer.андроїд.com. За цією документацією досить просто розпочати писати перші кроки та дізнаватись будову будь-якого андроїд-проєкта (рис 3.1).

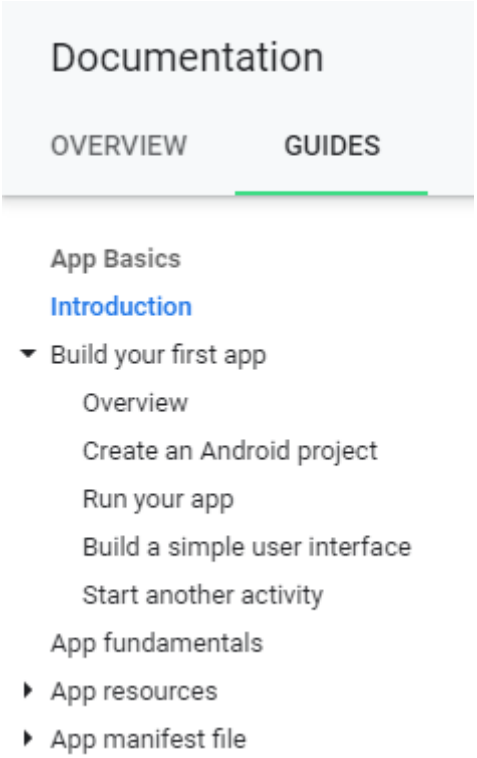

Рисунок 3.1 – Початкові етапи у документації для розробників

Далі слідуючи за кроками можна створювати проєкти із допомогою спеціального вбудованого помічника у Android Studio. Є можливість імпортувати проєкт, який вже зроблений і також можна зробити новий. Є можливість імпортувати тільки певні конфігураційні дані чи файли або ж завантажити із веб-сервіса для хостинга проєкти GitHub за допомогою git, google cloud, subversion чи інших.

Далі обирається назва проєкта, версії, що підтримуються, також вибір мови Java, Kotlin та проєкт створюється (рис 3.2) [1,8,9].

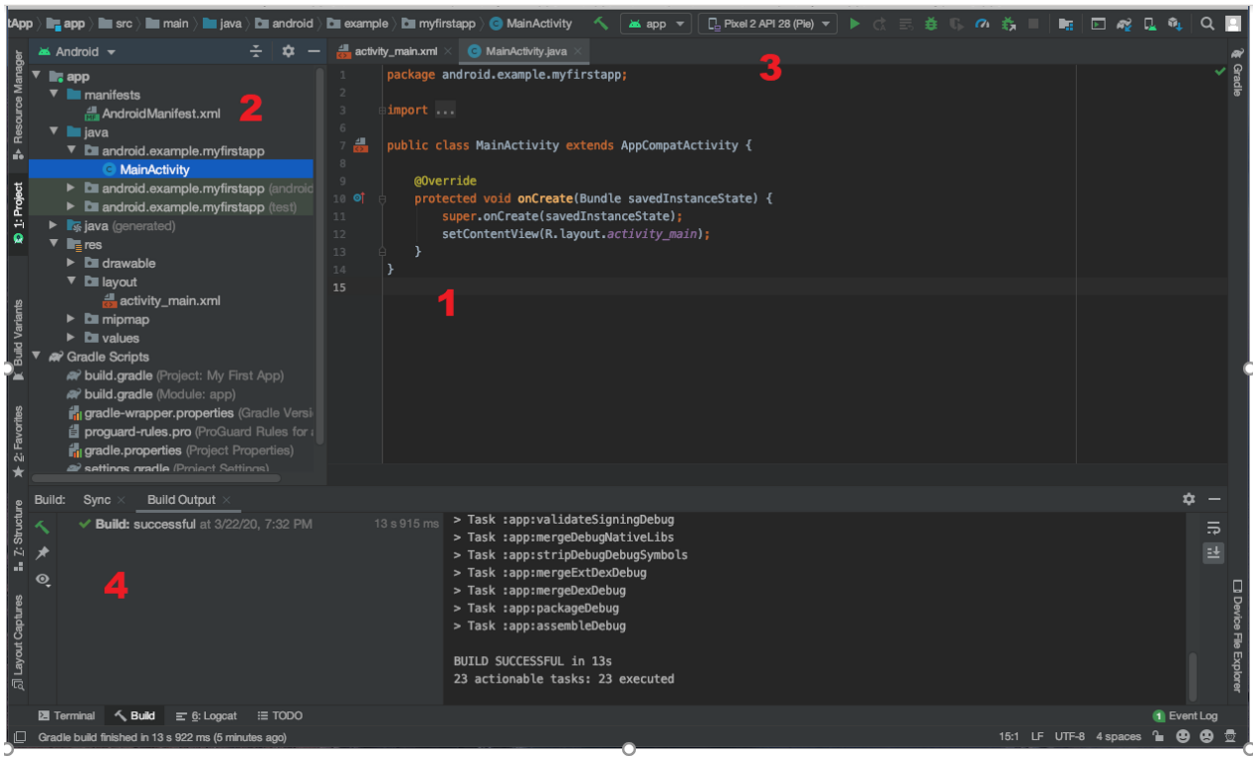

Рисунок 3.2 – Головний екран Android Studio

На зображенні, що знаходиться зверху написані цифри:

1) редактор кода, дає можливість написати, виправляти код і також має функції підказки, автоматичне форматування та інше.

2) структура проєкта – знаходяться усі файли, що належать до проєкта.

3) пристрій для запуска, у даному варіанті емулятор і також підписано версію андроїд з якої запускається, при правильному налаштованому проєкті та емуляторі, цей пристрій автоматично буде відображено у цьому ж вікні.

4) найнижче меню інструментів дає можливість провести відстеження збірки, використати термін – “Лог” та інше.

3.1 Основна архітектура застосунку

Основні терміни у андроїд застосунку, це основні компоненти за допомогою яких можна створити навігацію, передавання даних між екранами, а також інтерфейс, що призначено для користувача та ін. [1, 10,11].

– Activity є одним із основних компонентів, котрий можна представити елементом, що призначений для користувацького інтерфейсу;

– Fragment є спеціальним класом, який має відношення до фреймворку андроїд. Фрагмент (Fragment) подібний активіті (Activity), проте

може знаходитись тільки усередині активності (Activity) та охоплювати певну область чи усе вікно. Fragment легший за Activity і найчастіше використовують їх як незалежні екрани для інкапсуляції матеріалів певних екранів;

– View є спеціальним класом, що описує працездатність основних функцій по кожному з компонентів екрана, як приклад кнопка чи текст. Отже, кожен з тих компонентів береться від нього та реалізує свої певні відмінності чи функції. Також можна додати, що цим класом, зазвичай не користуються, а береться готовий клас, як приклад Button;

– View Group є контейнером, щоб зберігати окремі View, також є безліч View Group, кожна з яких визначається власною поведінкою усередині себе та встановлює розмітку;

– Layout є файлом із розширення .xml, що зберігає код для будь-якої розмітки.

На рис. 3.3 відображено початковий екран застосунку NetRunner.

Рисунок 3.3 – Початковий екран застосунку NetRunner

Із головного екрана застосунку бачимо таке відображення (рис 3.4).

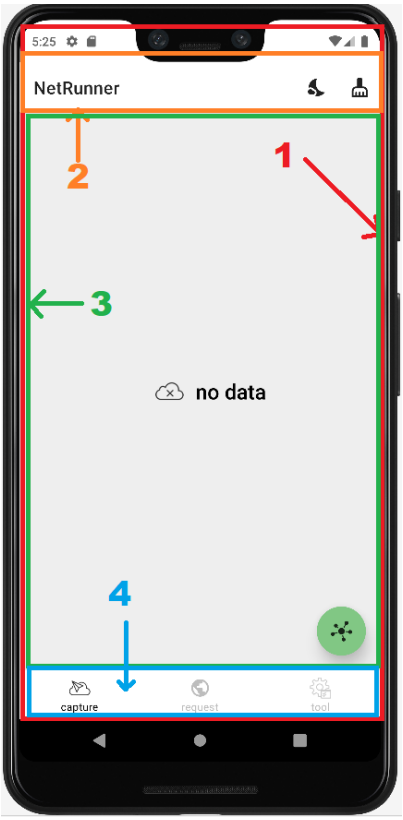

Рисунок 3.4 – Головний екран з позначеннями

1) перший контур це Activity, тобто вікно управління, що містить у собі усі елементи які призначені для користувацького інтерфейсу. Також у Activity є певний життєвий цикл, під керуванням системи, як приклад при згортанні застосунку, він зостанеться у пам'яті та буде там знаходитись, поки пам'ять не буде потрібна. Коли користувач зможе відкрити застосунок то він повернеться у ту ж точку, звідки вийшов у попередній раз;

2) другий контур ActionBar чи ToolBar є панеллю дій, та окремим елементом, який є View, може бути необхідним для того, щоб знайти екран у якому ми знаходимось, також відображення кнопок для доволі швидкого користування;

3) третій контур – фрагмент (Fragment). Fragment знаходиться усередині активності (Activity) у спеціальному контейнері, у якому можуть замінюватись. Як приклад видно, що екран capture та фрагмент завантажено також CaptureFragment.kt, що і показує кнопку для захоплення трафіка, та текст показує, що наразі список є порожнім;

4) четвертий контур є панеллю нижньої навігації. За переходами до іконки унизу, екрани можуть змінюватись, тобто кожен новий Fragment завантажиться і буде вставлено у контейнер.

Коли використовується одна Activity можемо розділити саму відповідальність між певними незалежними фрагментами (Fragment), де у кожному буде реалізовано необхідний функціонал чи екран [11].

На рис. 3.5 зображено принципи заміни Fragment у контейнері. Описується заміна фрагменту, як доволі складний процес, який у деталях розглядатиметься далі, проте на рисунку його показано у простій формі.

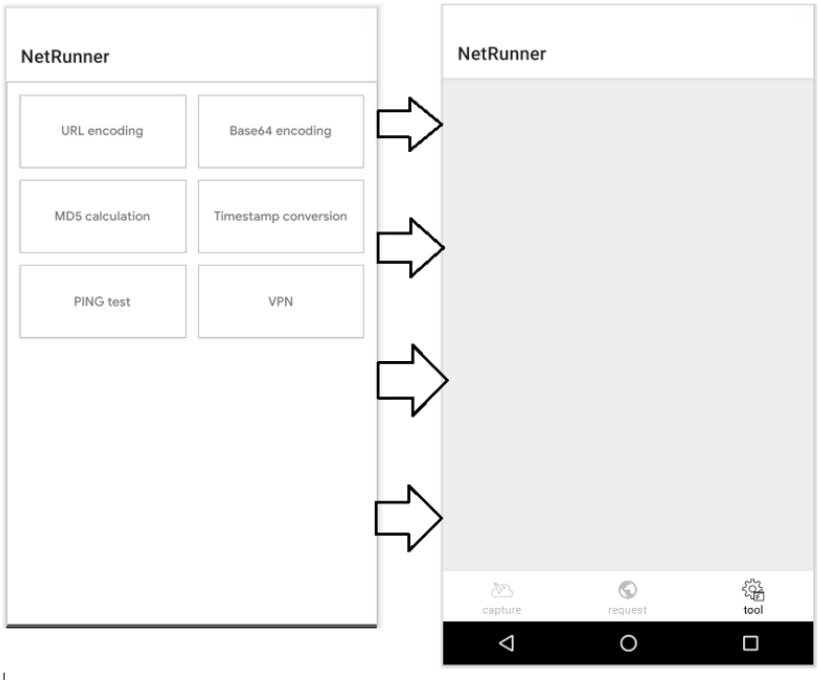

Рисунок 3.5 – Принцип заміни Fragment у контейнері

3.2 Розробка головного мережного модулю та реалізація VPN

NetRunner – це програма створена для того, щоб захопити пакети, вона захоплює та відображає HTTP-з'єднання, проте не HTTPS. Одним з основних моментів є створення VPNService та його ж активація, що змусить увесь трафік даних проходити крізь віртуальний інтерфейс, який керується даним застосунком та його простором, де і можна отримувати IP-пакет за допомогою читання із віртуального інтерфейсу. Цей метод, як один з найбільш захищених та немає потреби у root-правах. Коли буде запущено VPN, застосунок зможе отримати кожен байт, який відсилається пристроєм, а також може повернути і необроблені байти. Окрім того, щоб перенаправляти ці байти до провайдера VPN, проводиться дослідження їх і опрацювання, а далі вже просто поміщуємо їх до реальної мережі. У даному випадку є можливість побачити мережні байти,

проте важливішим є те, що не втручаємося безпосередньо у мережу пристрою та для цього не потрібно мати зовнішній VPN. На рис. 3.6 зображено принцип підключення андроїд мережі до шлюзу VPN за допомогою VpnService [11].

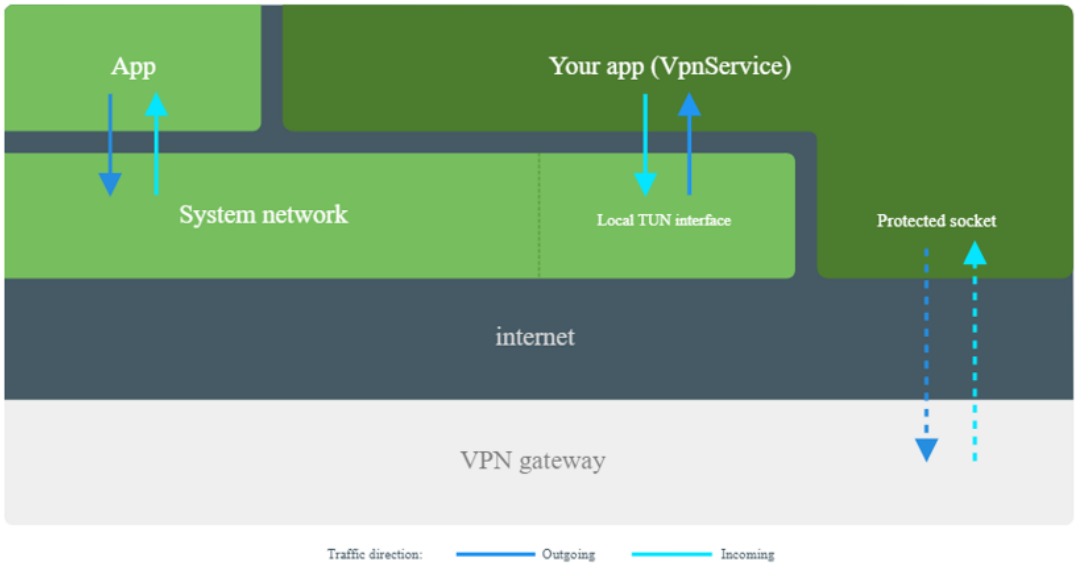

Рисунок 3.6 – Підключення андроїд мережі до шлюзу VPN за допомогою VpnService

Так як увесь процес створення певного VPN сервіса є доволі складним та громіздким, тому на початку розберемо низькорівневу абстракцію, що допоможе створити VPN та точку крізь яку можна отримати дійсні необхідні дані із мережі.

Також класи IpPacketHeader, TcpPacketHeader, UdpPacketHeader створюють заголовок даних пакету, представляють собою прості дата класи, які мають більшу кількість полів. Як приклад клас IpPacketHeader, можна побачити змінні, щоб зберігати дані про пакети (рис. 3.7) [11].

```
/**
 * IP message format
 * 0                                        15  16                                  31
 * |  ------------------------------------------------------------------------------ |
 * | 4 digits | 4 digits first | 8 digits service type |      16 digits total length        |
 * | Version number | Section length | (TOS)           |      (total  length)                |
 * |------------------------------------------------------------------------------ |
 * |             16-bit identifier                     |  3 bit |      13 bit chip offset    |
 * |                                                   |  Mark  | (offset)                   |
 * |------------------------------------------------------------------------------ |
 * |        8-bit survival time TTL  | 8-bit protocol  |             16-bit header checksum   |
 * |                                                                                |
 * |--------------------------------------------------------------------------------|
 * |                        32-bit source IP address (source  address)              |
 * |--------------------------------------------------------------------------------|
 * |                         32-bit destination IP address (destination  address)   |
 * |--------------------------------------------------------------------------------|
 * |                                      32-bit option (if any)                    |
 * |--------------------------------------------------------------------------------|
 * |                                                                                |
 * |                                      data                                      |
 * |                                                                                |
 * |--------------------------------------------------------------------------------|
 **/
```

Рисунок 3.7 – Структура IP пакету у класі IpPacketHeader

Структура пакета потрібна для візуального відображення в основному, щоб зрозуміти проєктування IP-заголовку, а також інших заголовків, таких як TCP, UDP. Вони є основними класами для зберігання у пакетах (рис. 3.8) [11].

Так само зроблені заголовки в інших пакетах для того, щоб мати можливість їх проаналізувати та зберігати у майбутньому. Далі буде описуватись створення парсеру для HTTP запиту та створення NAT сесії (рис. 3.9 – 3.12). Основна працездатність HttpRequestHeaderParser у тому, щоб зрозуміти типи запита, потрібно щоб це був HTTP запит, потім отримати тип NAT сесії з необхідними параметрами і далі вже розібрати рядок, що складається із уніфікованого покажчика ресурсів (англ. URL) (рис 3.13) [1, 4].

```
// The serial numbers here are all byte positions
public static final short IP = 0x0800;
public static final byte ICMP = 1;

public static final byte TCP = 6; //6: TCP protocol number
public static final byte UDP = 17; //17: UDP protocol number
public static final byte offset_proto = 9; //9: 8-bit protocol offset
public static final int offset_src_ip = 12; //12: source ip address offset
public static final int offset_dest_ip = 16; //16: target ip address offset
static final byte offset_ver_ihl = 0; //0: version number (4bits) + header length (4bits)
static final byte offset_tos = 1; //1: service type offset
static final short offset_tlen = 2; //2: total length offset
static final short offset_identification = 4; //4: 16-bit identifier offset
static final short offset_flags_fo = 6; //6: flag (3bits) + chip offset (13bits)
static final byte offset_ttl = 8; //8: survival time offset
static final short offset_crc = 10; //10: first checksum offset
static final int offset_op_pad = 20; //20: option + padding

// ip message data
public byte[] data;
public int offset;

public IpPacketHeader(byte[] data, int offset) {...}

public int getDataLength() {
    return this.getTotalLength() - this.getHeaderLength();
}
```

Рисунок 3.8 – Реалізація заголовку IP пакета у коді

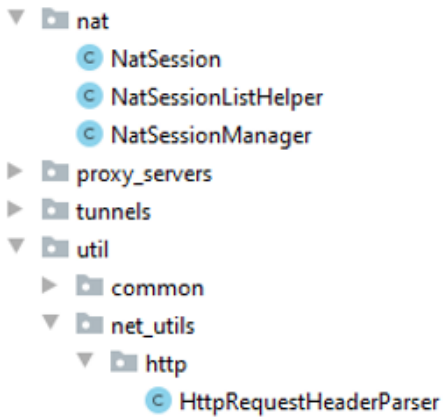

Рисунок 3.9 – Структура папок NAT та http

```
public static void getHttpHostAndRequestUrl(NatSession session, byte[] buffer, int offset, int count) {
    session.isHttp = true;
    session.isHttpsSession = false;
    String headerString = new String(buffer, offset, count);
    String[] headerLines = headerString.split( regex: "\\r\\n");
    String host = getHttpHost(headerLines);
    if (!TextUtils.isEmpty(host)) {
        session.remoteHost = host;
    }
    parseRequestLine(session, headerLines[0]);
}

  public static void parseRequestLine(NatSession session, String requestLine) {
      String[] parts = requestLine.trim().split( regex: " ");
      if (parts.length == 3) {
          session.method = parts[0];
          String url = parts[1];
          session.pathUrl = url;
          if (url.startsWith("/")) {
              if (session.remoteHost != null) {
                  session.requestUrl = "http://" + session.remoteHost + url;
              }
          } else {
              if (session.requestUrl.startsWith("http")) {
                  session.requestUrl = url;
              } else {
                  session.requestUrl = "http://" + url;
              }

          }
      }
  }
```

Рисунок 3.10 – Принцип працездатності отримання рядка запиту

```java
public class NatSession implements Serializable {
    public static final String TCP = "TCP";
    public static final String UDP = "UDP";
    public String type;
    public String ipAndPort;
    public int remoteIP;
    // The port number of the address to be accessed
    public short remotePort;
    public String remoteHost;
    public short localPort;
    // How much data has been uploaded (IP datagram or TCP datagram header is not included)
    public int bytesSent;
    /**
     * Record how many network packets have been sent in the current network session
     */
    public int packetSent;
    public long receiveByteNum;
    public long receivePacketNum;
    public long lastRefreshTime;
    public boolean isHttpsSession;
    /// The url that the user intends to visit,
    // but does not include the port number,
    // for example: the user is visiting http://192.168.100.103:901/?a=2 then here will
    // Becomes http://192.168.100.103/?a=2
    public String requestUrl;
    public String pathUrl;
    public String method;
    public ApplicationInfo applicationInfo;
    public long connectionStartTime = System.currentTimeMillis();
    public long vpnStartTime;
    public boolean isHttp;
```

Рисунок 3.11 – клас NatSession

Клас NatSession необхідний для зберігання у собі даних про кожну сесію і застосунок який використовує у певний час з'єднання. Цей клас має більшу важливість у цілому для даного застосування, він як сполучна ланка між низькорівневим запитом та роботою із мережею і інтерфейсом, що призначено для користувача, і де уся інформація та дані будуть відображатися. Також є класи NatSessionManager та NatSessionListHelper. NatSessionManager створює подібну сесію як і на (рис 3.12), проте NatSessionListHelper потрібен, щоб заповнити основний список пакетами з сесії та видалення (рис 3.13) [4,10].

```java
/**
 * Create a session
 *
 * @param portKey source port
 * @param remoteIP remote ip
 * @param remotePort remote port
 * @return NatSession object
 */
public static NatSession createSession(short portKey, int remoteIP, short remotePort, String type) {
    if (sessions.size() > MAX_SESSION_COUNT) {
        clearExpiredSessions();
    }

    NatSession session = new NatSession();
    session.lastRefreshTime = System.currentTimeMillis();
    session.remoteIP = remoteIP;
    session.remotePort = remotePort;
    session.localPort = portKey;
    // If there is no destination ip, convert the digital ip to 192.168.0.1

    if (session.remoteHost == null) {
        session.remoteHost = Packets.ipToString(remoteIP);
    }
    session.type = type;
    session.refreshIpAndPort();
    sessions.put(portKey, session);

    return session;
}

public static void removeSession(short portKey) {
    sessions.remove(portKey);
}
```

Рисунок 3.12 – Створення та видалення NAT сесії

Можна побачити яким саме чином робиться NAT сесія, а саме отримання функції на вхід, порту, віддаленого ip, віддаленого порту, типу мережі. Далі присвоюються ці ж дані у змінні у дата клас NatSession та отримується нова сесія у реальному часі [6].

```
public class NatSessionListHelper {
    public static Collection<NatSession> getAllSessions() {
        File file = new File( pathname: TcpDataSaver.CONFIG_DIR
                + TimeFormatter.formatToYYMMDDHHMMSS(VpnProxyServer.getStartTime()));
        ACache aCache = ACache.get(file);
        String[] list = file.list();
        ArrayList<NatSession> baseNetSessions = new ArrayList<>();
        if (list != null) {
            for (String fileName : list) {
                NatSession netConnection = (NatSession) aCache.getAsObject(fileName);
                baseNetSessions.add(netConnection);
            }
        }
        PortSessionInfoService portSessionInfoService = PortSessionInfoService.getInstance();
        if (portSessionInfoService != null) {
            Collection<NatSession> aliveConnInfo = portSessionInfoService.getAndRefreshSessionInfo();
            if (aliveConnInfo != null) {
                baseNetSessions.addAll(aliveConnInfo);
            }
        }
        Collections.sort(baseNetSessions, (Comparator) (o1, o2) → {
                return Long.compare(o2.lastRefreshTime, o1.lastRefreshTime);
        });
        return baseNetSessions;
    }

    public static void clearCache() {
        String data = TcpDataSaver.DATA_DIR;
        String config = TcpDataSaver.CONFIG_DIR;
        File dataDir = new File(data);
        File configDir = new File(config);
        FileManager.deleteUnder(dataDir);
        FileManager.deleteUnder(configDir);
        NatSessionManager.clearAllSession();
    }
}
```

Рисунок 3.13 – Клас NatSessionListHelper

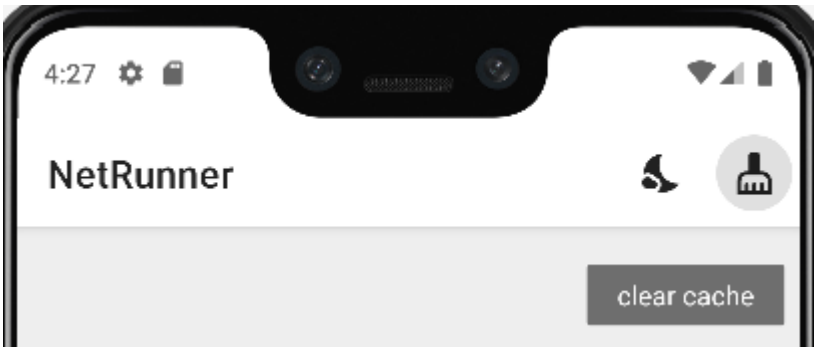

Рисунок 3.14 – Відображення кнопки clearCache у інтерфейсу

У даному класі отримуються всі сесії, створюється список із типом NatSession, поміщається у ньому всі активніші сесії по шляху перевірки з отримання об'єкта PortSessionInfoSercvice, це клас що виконує певні команди по доволі тривалому зчитуванні даних з програми у фоновому режимі. Якщо ж він працює, то це означає, що сесії валідні та можуть нести у собі інформацію. Далі вже йде clearCache(), що проводить видалення ресурсів і повністю видаляє від попередньої сесії сеанс (рис 3.15) [4].

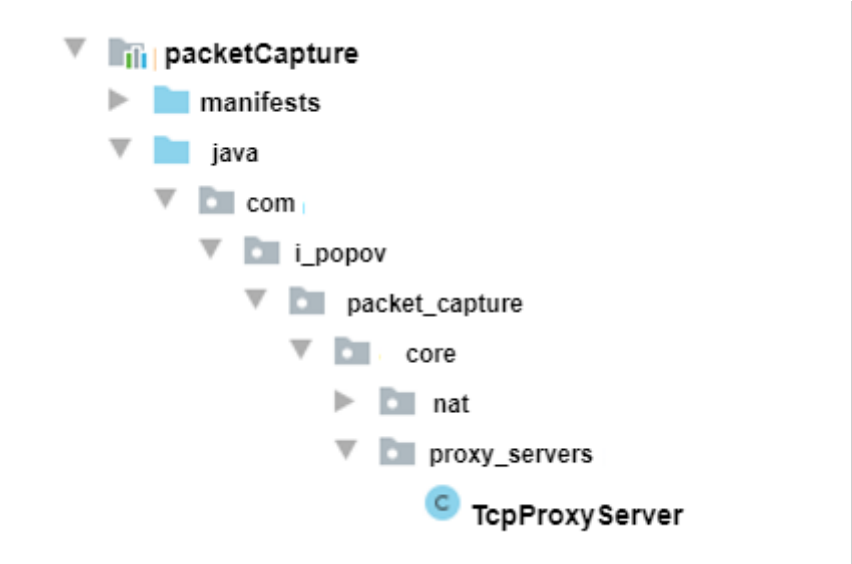

Рисунок 3.15 – Структура модулю proxy

Проксі-сервер – засіб чи спосіб зв'язку у будь-якій комп'ютерній мережі. Він дає можливість передавати через свій адрес, запити клієнтів на сервер, також не пов'язує цільовий сервер та користувача, отже він буде деякою середньою ланкою, яку можна контролювати. Завдяки TcpProxyServer є можливість фільтрувати мережну комунікацію та кеш даних і тому анонімно проаналізовувати інформацію у переданих пакетах. TcpProxyServer є дуже масивним і продемонструвати його складову будову доволі складно, тому на рис. 3.16 зображено основну складову [1, 4].

```
public TcpProxyServer(int port) throws IOException {
    selector = Selector.open();
    serverSocketChannel = ServerSocketChannel.open();
    // Register Channel to Selector, Channel must be non-blocking.
    // So FileChannel does not apply to Selector, because FileChannel cannot be switched to non-blocking mode,
    // Because FileChannel does not inherit SelectableChannel. Socket channel can be used normally.
    serverSocketChannel.configureBlocking(false);
    serverSocketChannel.socket().bind(new InetSocketAddress(port));
    serverSocketChannel.register(selector, SelectionKey.OP_ACCEPT);
    // When the passed port is 0, the system will randomly distribute a usable port
    this.port = (short) serverSocketChannel.socket().getLocalPort();
}

public boolean isStopped() { return stopped; }

public short getPort() { return port; }

public void start() {
    serverThread = new Thread(runnable);
    serverThread.start();
}

public void stop() {
    stopped = true;
    if (selector != null) {
        try {
            selector.close();
            selector = null;
        } catch (Exception ignored) {
        }
    }
    if (serverSocketChannel != null) {
        try {
            serverSocketChannel.close();
            serverSocketChannel = null;
        } catch (Exception ignored) {
        }
    }
```

Рисунок 3.16 – Структура TcpProxyServer

Можна побачити, що він працює у незалежному потоці, також за стандартом можна виділити порт 0, зазвичай для усіх операційних систем має ознаку стандарту, де є можливість зробити підбір діючого порту для того, щоб отримати дані від серверу [1, 6].

Далі показані TCP тунелі, що необхідні для передавання даних протоколами. Тунелювання – це дія у процесі якої відбувається створення деякого логічного з'єднання між обома точками із залученням інкапсуляції різноманітних протоколів. На рис. 3.17 зображено структуру TCP і UDP тунелів.

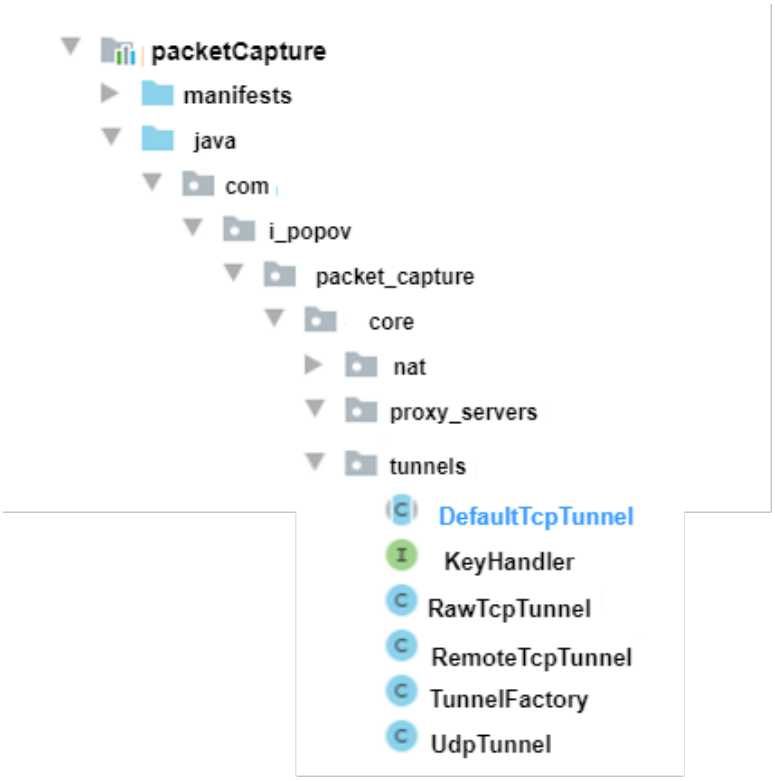

Рисунок 3.17 – Структура TCP і UDP тунелів

Далі перейдемо до VPN сервіса та працездатності усіх розглянутих класів разом. Як видно з рис 3.18, структура VPN має 4 класи. Перший клас CoreVpnRunner – клас, що функціонує на задньому плані та є потоковим, він може зчитувати на час увімкненого VPN усі пакети (TCP, UDP), довжину та інші дані. На рис 3.19 показано метод handlePacket, що визначає типи пакетів, а також на рис 3.20 зображено розбирання на прикладі TCP-пакету [4, 10].

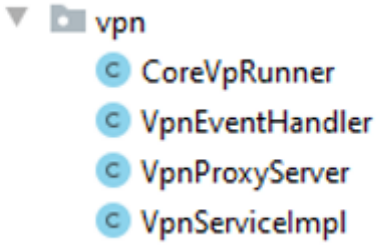

Рисунок 3.18 – Пакет VPN

```
private boolean handlePacket(byte[] data, int len) throws IOException {
    IpPacketHeader ipPacketHeader = new IpPacketHeader(data, offset: 0);
    switch (ipPacketHeader.getProtocol()) {
        case IpPacketHeader.TCP:
            return disassembleTcpPacket(ipPacketHeader, len);
        case IpPacketHeader.UDP:
            return disassembleUdpPacket(ipPacketHeader, len);
        default:
            return false;
    }
}
```

Рисунок 3.19 – Функція handlePacket

```
private boolean disassembleTcpPacket(IpPacketHeader ipPacketHeader, int len) throws IOException {
    TcpPacketHeader tcpPacketHeader = new TcpPacketHeader(ipPacketHeader.data, ipPacketHeader.getHeaderLength());
    short srcPort = tcpPacketHeader.getSourcePort();
    if (srcPort == tcpProxyServer.getPort()) {
        NatSession session = NatSessionManager.getSession(tcpPacketHeader.getDestinationPort());
        if (session != null) {
            ipPacketHeader.setSourceIP(ipPacketHeader.getDestinationIP());
            tcpPacketHeader.setSourcePort(session.remotePort);
            ipPacketHeader.setDestinationIP(Packets.ipToInt(VpnProxyServer.getAddress()));
            Packets.computeTcpChecksum(ipPacketHeader, tcpPacketHeader);
            outPutStream.write(ipPacketHeader.data, ipPacketHeader.offset, len);
        }
```

Рисунок 3.20 – Функція disassembleTcpPacket

Тому у разі, коли все добре, маємо активну сесію, що використовує VPN, та можна отримати усі дані потрібної мережі та даного з'єднання та зберегти. VpnEventHandler дає можливість проводити контроль зміни VPN, а саме при натисканні кнопки запуску, він проводить ініціалізацію усіх ресурсів та запускається. Далі опишемо VpnProxyServer, у цього класу є статичні поля та методи. Цей клас дає можливість вказувати певні дійсні налаштування VPN сервіса, вказувати поточну IP-адресу, вказувати розмір мінімальної одиниці даних, що пересилається, вказувати час запуска VPN. Отже це клас дає можливість поділитись певними конфігураційними даними із наступними компонентами мережного шара [1, 6].

```java
// An intermediate server in computer networks
// that acts as an intermediary between and the target server
public class VpnProxyServer {
    private static WeakReference<VpnServiceImpl> vpnService = null;

    public static void setVpnService(VpnServiceImpl vpnService) {
        VpnProxyServer.vpnService = new WeakReference<>(vpnService);
    }

    public static boolean protect(Socket socket) {
        if (vpnService == null || vpnService.get() == null) return false;
        return vpnService.get().protect(socket);
    }

    public static boolean protect(DatagramSocket socket) {
        if (vpnService == null || vpnService.get() == null) return false;
        return vpnService.get().protect(socket);
    }

    public static int getMtu() { return VpnServiceImpl.MTU; }

    public static String getAddress() { return VpnServiceImpl.ADDRESS; }

    public static long getStartTime() {
        if (vpnService == null || vpnService.get() == null) return -1;
        return vpnService.get().getStartTime();
    }
}
```

Рисунок 3.21 – клас VpnProxyServer

Клас VpnServiceImpl. Цей клас є реалізацією VPN сервіса і інкапсулює у собі окрім стандартної функції VPN більш детальні настройки з'єднання (рис. 3.22). Можна побачити налаштування з'єднань, вказівки IP, DNS, MTU та прописання ключових слів vpnThread. Також можна побачити, що цей клас є від VpnService, це пакет у Андроїд Framework, що може допомогти працювати із мережами та має багато інших інтерфейсів (рис. 3.23, 3.24) [7].

```java
public class VpnServiceImpl extends VpnService {
    /**
     * The maximum transmission unit of the virtual network port.
     * If the length of the packet sent exceeds this number,
     * it will be sub-packaged; default set to 1500, now 4096 for more stable packaging
     */
    static final int MTU = 4096;
    static final String SESSION = "NetRunner";
    /**
     * Set the IP address of the VPN (only IPv4 is supported here)
     * This address can be checked, the address in 360 Flow Guard is 192.168.*.*;
     * Many also use 10.0.2.0 OR 10.0.0.10; not sure, you can try. Here is {@linkplain#LOCAL_IP}
     */
    static final String ADDRESS = "10.0.0.10";
    /**
     * Only the matched IP packets will be routed to the virtual port. If it is 0.0.0.0/0,
     * all IP packets will be routed to the virtual port;
     */
    static final String ROUTE = "0.0.0.0"; // Intercept everything
    // Below are some common DNS addresses
    //GOOGLE SET
    static final String DEFAULT_DNS = "8.8.8.8";
    //unused for now
    static final String GOOGLE_DNS_FIRST = "8.8.8.8";
    static final String AMERICA = "208.67.222.222";
    static final String CHINA_DNS_FIRST = "114.114.114.114";
    private static final String KEY_CMD = "key_cmd";
    private ParcelFileDescriptor vpnInterface;
    private Thread vpnThread;
    private long startTime;

    public VpnServiceImpl() {
    }
```

Рисунок 3.22 – Оголошення полів у класі VpnServiceImpl

```
Decompiled .class file, bytecode version: 52.0 (Java 8)
Sources for 'Android API 30 Platform' not found.
1   /.../
5
6   package android.net;
7
8   import ...
20
21  public class VpnService extends Service {
22      public static final String SERVICE_INTERFACE = "android.net.VpnService";
23      public static final String SERVICE_META_DATA_SUPPORTS_ALWAYS_ON = "android.net.VpnService.SUPPORTS_ALWAYS_ON";
24
25      public VpnService() { throw new RuntimeException("Stub!"); }
```

Рисунок 3.23 - VpnService в пакті андроїд.net

```
private void establish() {
    Builder builder = new Builder();
    builder.setMtu(MTU);
    builder.setSession(SESSION);
    builder.addAddress(ADDRESS,  prefixLength: 0);
    builder.addRoute(ROUTE,  prefixLength: 0);
    String dns = Shells.getDns();
    if (dns == null || dns.isEmpty()) {
        builder.addDnsServer(DEFAULT_DNS);
        // It is to add automatic completion of DNS domain name.
        // The DNS server must be searched by the full domain name,
        // But it is too troublesome to enter the full domain name every time you look up,
        // you can simplify it by configuring the automatic completion rule of the domain name.
        // .addSearchDomain()
        /*
         * Set the name of this session. It will be displayed in system-managed dialogs
         * and notifications. This is recommended not required.
         */
        // .setSession(getString(R.string.app_name))
    } else {
        builder.addDnsServer(Shells.getDns());
    }
    vpnInterface = builder.establish();
    FileDescriptor fd = vpnInterface.getFileDescriptor();
    vpnThread = new Thread(new CoreVpRunner(fd));
    vpnThread.start();
    VpnProxyServer.setVpnService(this);
    startTime = System.currentTimeMillis();
    VpnEventHandler.getInstance().notifyStart();
}
```

Рисунок 3.24 – Створення VPN

Із допомогою Builder та функції established() можемо встановити усі значення, які задано і далі створити дискриптор для файлів, що будуть надходити, щоб їх розібрати і потім запустити новий потік із вже наявним класом CoreVpnRunner, а також Service, компонент андроїд, що відповідає за певні планові операції у системах андроїд [11].

Отже, після усього вище описаного, а також налаштувань, що призначено для інтерфейсу користувача та після запуску, можна бачити наступне повідомлення (рис. 3.25) і (рис. 3.26) [1, 7].

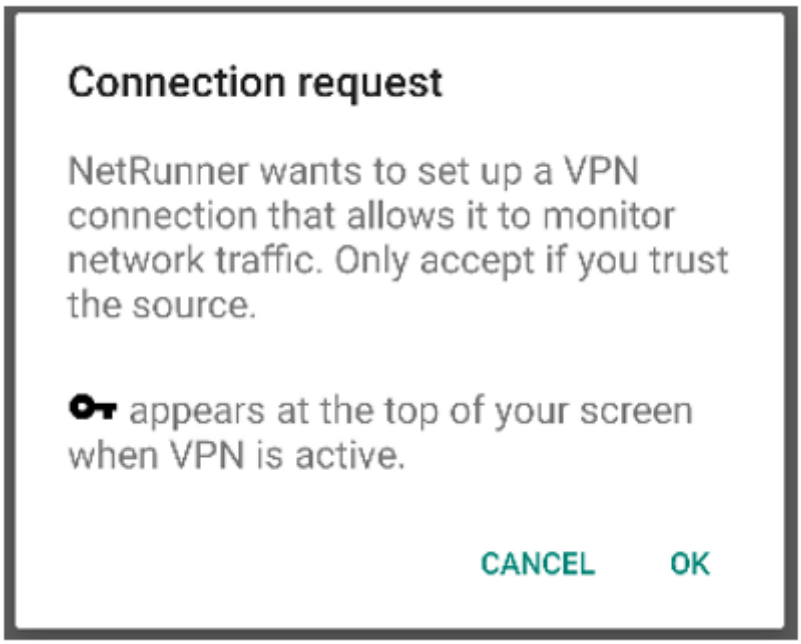

Рисунок 3.25 – Запит на VPN з'єднання

Рисунок 3.26 – Вікно, яке показує роботу VPN

3.3 Реалізація інтерфейсу користувача

Із боку користувача є певний фрагмент (Fragment), у якому зберігається розмітка призначена для інтерфейсу користувача та функції для керування нею (рис. 3.27) [8].

```
private fun initView(root: View) {
    adapter = PacketAdapter(packets, sessionList, requireContext())
    val recyclerView: RecyclerView = root.findViewById(R.id.rv_packet)
    root.start_capture.setMinAndMaxProgress(0.0f, 0.90f)
    recyclerView.layoutManager = LinearLayoutManager(requireActivity())
    recyclerView.addItemDecoration(DividerItemDecoration(requireActivity(),
            DividerItemDecoration.VERTICAL))
    recyclerView.adapter = adapter

    if (packets.size == 0) {
        showViews(root.placeholder_no_data, root.cloud_img_no_data)
    }

    root.cardViewStartStop.setOnClickListener {  it: View!
        if (buttonStateStart) {
            startCapture()
        } else {
            stopCapture()
        }
        buttonStateStart = !buttonStateStart
    }

    //open packet and get details about current request/response
    adapter?.setOnItemClickListener(OnItemClickListener { _, _, position, _ ->
        if (sessionList.size == 0) return@OnItemClickListener
        val session = sessionList[position]
        val dir = StringBuilder()
                .append(TcpDataSaver.DATA_DIR)
                .append(TimeFormatter.formatToYYMMDDHHMMSS(session.vpnStartTime))
                .append("/")
                .append(session.uniqueName)
                .toString()
        val intent = Intent(requireActivity(), PacketDetail::class.java)
        intent.putExtra(KEY_DIR, dir)
        startActivity(intent)
    })
```

Рисунок 3.27 – Зображено функцію initView

Розташована вона у одній із функцій життєвого цикла фрагменту (Fragment), яка проводить виклик у певний час операційної системи Андроїд (рис. 3.28) [5].

```
override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?,
                          savedInstanceState: Bundle?): View? {
    // Inflate the layout for this fragment
    val rootView = inflater.inflate(R.layout.fragment_capture, container, attachToRoot: false)
    initView(rootView)
    return rootView
}
```

Рисунок 3.28 – Розташування функції onCreateView

Ця функція може вирішити важливі завдання, установку пакету, ініціалізацію усіх компонентів та запуск фрагменту (Fragment) у контейнері. Далі надані функції для верхнього меню інструментів (рис. 3.29) [5].

```
private fun initView(root: View) {...}

//override options menu
override fun onCreateOptionsMenu(menu: Menu, inflater: MenuInflater) {...}

override fun onOptionsItemSelected(item: MenuItem): Boolean {...}

@SuppressLint( ...value: "CommitPrefEdits")
private fun sharedPrefsInit() {...}

private fun darkMode() {...}
```

Рисунок 3.29 – Функція onCreateView

Далі є функції onStart() та onStop(), що можуть описати поведінку при згортанні застосунку, а також і функції, необхідні для початка захоплення та зупинки (рис. 3.30) [1, 5].

```
override fun onStart() {...}

override fun onStop() {
    super.onStop()
    VpnEventHandler.getInstance().cancelAll()
}

private fun startCapture() {
    val intent = VpnService.prepare(requireContext())
    if (intent == null) {
        onActivityResult( requestCode: 0, Activity.RESULT_OK,  data: null)
    } else {
        startActivityForResult(intent,  requestCode: 0)
    }
}

private fun stopCapture() {
    val intent = Intent(requireActivity(), VpnServiceImpl::class.java)
    intent.putExtra(KEY_CMD,  value: 1)
    APP_ACTIVITY.startService(intent)
}

companion object {
    private const val KEY_CMD = "key_cmd"
    private const val KEY_DIR = "key_dir"
    fun newInstance(): CaptureFragment {
        return CaptureFragment()
    }
}
```

Рисунок 3.30 – Функції захоплення та зупинки по захопленні пакетів

4 РОЗРОБКА ПЛАНУ ПРОСУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

4.1 Огляд методів просування мобільного застосунку

Після того, як програма створена та розміщена в мережі, про неї необхідно повідомити потенційних споживачів. На початку існування програми більшість відвідувачів потрапляють на сторінки через пошукові системи в Play market. Для того, щоб програма почала виводитися за тими чи іншими пошуковими запитами, вона повинна бути проіндексованою пошуковими роботами та занесена до їх баз даних контенту. Підняти сторінки в пошуковій видачі та підвищити відвідуваність сайту можна кількома способами:

1) оптимізація – це унікальна назва програми та наявність ключових слів в описі програми. Але не тільки. Потрапляння в ТОП залежить, так само, від кількості завантаження програми, а також від її рейтингу серед користувачів. При цьому важливо своєчасно оновлювати програму, оскільки застарілі версії не просто видаляють з пошукової видачі, а й можуть зовсім видаляти з каталогів;

2) інтегрована реклама на інших застосунках. Для цього необхідно вчинити такі дії:

- комуніціювати з потенційними клієнтами у застосунках, які вони вже встановили;

- розмістити рекламу у популярних застосунках;

- категоризувати цільову аудиторію за інтересами, обравши саме ті застосунки, які мають попит;

- впровадити банери. Такі можливості широко надає Google AdWords;

3) контент-маркетинг. Після аналізу цільової аудиторії необхідно підібрати ресурси, на яких проводить час цільова аудиторія, і опублікувати ряд гостьових постів, в яких ви розповісте про особливості і переваги вашого застосунку. Для цього потрібно:

- записувати аудіо-підкасти;

- створити відеоролик;

- проявляти активність у соціальних мережах.

Використовуйте контент-маркетинг комплексно, розумно та не нав'язливо

впроваджуючи його всілякі механізми: гостьовий постінг, SMM, відеомаркетинг;

4) робота з лідерами думок. При просуванні програми важко обійтися без допомоги лідерів думок - авторитетів для цільової аудиторії. Щоб достукатися до мільйонів, вам потрібно визначити кілька людей, яких слухають ці мільйони. Потрібно зробити так, щоб ваш продукт сподобався, лідерам і їм захотілося розповісти про нього всім;

5) промо-сайт. Необхідно створити для мобільного застосунку офіційне представництво у мережі. Це може бути як окремий промо-сайт, так і опрацьована посадкова сторінка в рамках головного сайту. Ефективність ресурсу залежить від виду програми: стартап це або елемент комунікації великої компанії. Наявність промо-сайту дає додаткові переваги при просуванні мобільного застосунку:

- можливість візуалізації його особливостей;
- інструкції із застосування мобільних застосунків;
- контекстна реклама;
- націлення в соціальних мережах [12].

Розглянемо, як ранжуються програми.

Користувачі частіше встановлюють програми, які ранжуються у загальному топі або топах категорій магазинів, а також відображаються на верхніх позиціях у видачі за популярними запитами. Google не розкривають алгоритми ранжирування продуктів у Google Play. Відомі ключові фактори, що впливають позиції програм Google Play. До них входять такі:

- загальна кількість установок;
- динаміка установок. В Google Play розрахунковий період становить 48 – 72 години;
- рейтинг чи оцінки користувачів;
- число коментарів;
- динаміка видалення програми з пристрою;
- кількість запусків програми користувачами.

Фахівці з просування застосунків для операційної системи Android стверджують, що на позиції Google Play також впливає ціна продукту і зовнішні посилання. Місце застосування у пошуковій видачі магазину залежить від релевантності назви та опису запитів користувачів, а також від якості

скріншотів та іконок [12].

Щоб застосунок потрапив у ТОП, його має встановити велика кількість користувачів уперші дні після додавання до магазину. Кількість установок варіюється в залежності від регіону, категорії та магазину. Порядок числа – тисячі та десятки тисяч установок на добу.

Варто зазначити, що алгоритми ранжирування програм у магазинах Google Play Store регулярно оновлюються, але досконалими їх назвати складно. А це, у свою чергу, призводить до появи цілого загону фахівців із просування, обізнаних у тонкощах роботи з головними джерелами мобільного контенту.

Розглянемо інструменти моніторингу відвідуваності мобільного застосунку.

За виявленими методами просування надано рекомендації та пропозиції, реалізація яких дозволить розробити ефективний застосунок для залучення користувача.

4.2 Інструменти моніторингу відвідуваності мобільного застосунку

Розглянемо найбільш вживані інструменти моніторингу відвідуваності мобільного застосунку.

1. YouScan - перша і лідируюча система для професійного моніторингу соціальних медіа**.** YouScan відслідковує згадки брендів, продуктів, конкурентів у блогах, форумах, соціальних мережах Facebook, Twitter і навіть у YouTube**,** і надає результати моніторингу у зручному аналітичному інтерфейсі з функціями командної роботи [13].

2. BuzzLook **-** це російськомовний сервіс моніторингу соціальних медіа: Facebook, Flickr, YouTube та Twitter. Ця система моніторингу соціальних медіа дозволяє: стежити за репутацією вашого бренду; вивчати діяльність конкурентів у мережі; відповідати на запитання ваших клієнтів у їхньому середовищі (соціальних мережах); збирати пропозиції від клієнтів; підтримувати ваші on-line спільноти; працювати з запереченнями у мережі; дослідити ринки; краще просувати ваш продукт [13].

3. Twitalyzer **-** аналітична програма-клієнт для Google Play, що дозволяє відстежувати кількість переходів, аналізує позитивні та негативні коментарі та настрої та сегментує аудиторію. Інтегрована з Google Analytics, виводить

інтерактивні діаграми та графічні інструменти [12].

4. Google Analytics - дозволяє аналізувати вплив мобільних технологій на ваш бізнес. Якщо ви розробляєте мобільні програми, ви можете скористатися SDK для iOS та Android, щоб отримувати дані про їх використання [12].

Проведений аналіз напрямку моніторингу відвідуваності мобільного застосунку показує, що оптимальним буде вбудування у застосунок двох систем: Google Analytics та YouScan. Завдяки цим системам можемл виявити ряд недоліків реалізації, функціонування та реалізації застосунку, включаючи невідповідність мобільного застосунку цілям просування послуг.

Також до недоліків слід віднести малу ефективність використання якогось одного методу. Їх варто застосовувати комплексно, хоча це не завжди дозволяє досягти високих результатів у пошуковій видачі за рахунок низького рівня знань оптимізатора.

4.3 План просування мобільного застосунку в Інтернеті

Для подальшого просування розробленого застосунку та послуг за допомогою онлайн-каналів комунікації пропонується використовувати пошукову оптимізацію сайту та контекстну рекламу як найефективніші засоби залучення клієнтів в інтернеті.

Комплекс заходів спрямований на просування застосунку серед потенційних споживачів, які є жителями України, та спеціалістами ІТ-сфери, тому що дзастосунок в першу чергу спрямований на отримання точної інформації про споживаний трафік та його особливості.

Цей комплекс включає наступні заходи.

1. Реєстрація програми у системі Google Play.
2. Використання лічильників статистики відвідування.
3. Реєстрація у соціальних мережах.
4. Створення промо-сайту.
5. Реєстрація програми у безкоштовних каталогах.
6. Реєстрація програми на дошках оголошень.

Розглянемо їх докладніше.

Реєстрація програми у системі Google проводиться за три кроки.

Крок 1. Створення сертифікату. Будь-яка програма, що викладається в

магазин, повинна мати підписаний сертифікат. Сертифікат дозволяє вам ідентифікуватися як автору програми. І якщо хтось спробує викласти програму з таким самим ім'ям, як у вас, то йому буде відмовлено через конфлікт імен. Під ім'ям програми мається на увазі повна назва пакета.

Коли програма запускалася на емуляторі або телефоні, то середовище розробки автоматично підписувало програму сертифікатом налагодження. Для поширення через магазин налагоджувальний сертифікат не підходить, і вам потрібно підписати програму своїм унікальним сертифікатом.

Створимо підписаний APK-файл, який є файлом як notepad.exe у Windows. З'явиться діалогове вікно майстра, яке потрібно заповнити даними (рис. 4.1).

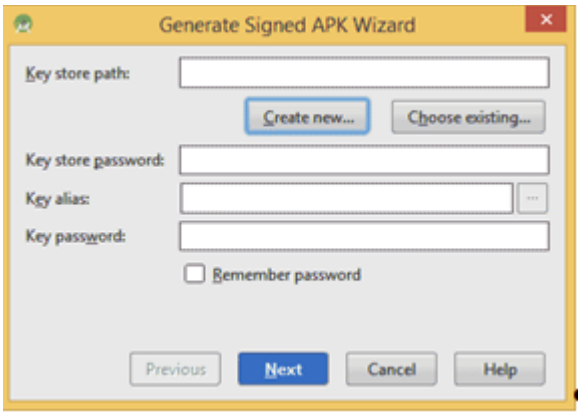

Рисунок 4.1 - Вікно створення сховища додатку

Далі заповнюються поля. Поля Password і Confirm .Тепер створюєте ключ для програми. В полі Alias (Псевдонім) вводиться назва ключа. Для ключа також потрібно створити пароль та підтвердити його (рис. 4.2).

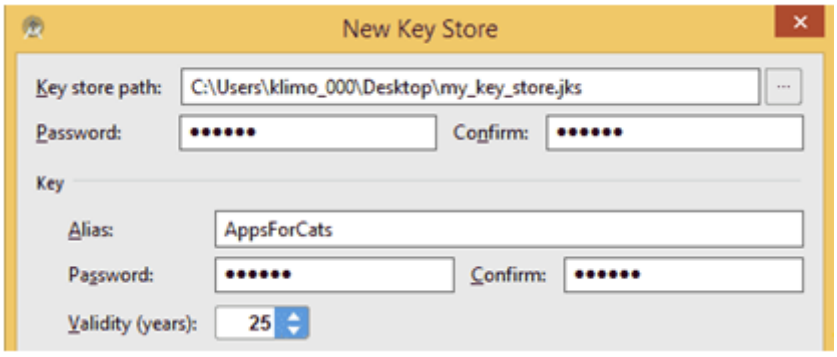

Рисунок 4.2 - Створення пароля та його підтвердження

Крок 2. Реєстраційний внесок. Переходимо на сторінку розробників, заповнюємо необхідні поля та вносимо оплату = 25$.

Крок 3. Завантаження програми. Якщо платіж пройшов успішно, посилання на наступний крок буде доступним і ви потрапляєте в спеціальний особистий кабінет, де можете додавати свої програми.

Весь інтерфейс англійською. Завантажуємо підготовлений APK-файл, а також необхідні картинки-скриншоти та значок У процесі додавання програми можна видаляти картинки і файл програми, так само оновлювати.

Використання лічильників статистики відвідування.

Після реєстрації програми в маркеті, відразу ж в сайт запроваджуються лічильники, які надалі збиратимуть статистику відвідувань та аналізуватимуть поведінку застосунку. У сайт вбудовані два найбільш популярні лічильники, а саме, AppMetrica, Google Analytics. Різноманітність лічильників дозволить охопити безліч різних звітів та вивести більш достовірні дані про відвідуваність.

Створення промо-сайту.

Промо-сайт – це особливий вид веб-ресурсів, створений для розкручування ринку бренду, товару, послуги чи події. Найчастіше такий сайт створюється для проведення саме рекламної компанії, тому інформація на ньому має відповідний характер: умови акцій, місця проведення, терміни, анонс, новини та інші відомості. Головну увагу при розробці промо-сайтів роблять на графічне оформлення, тому вони зазвичай дуже яскраві та динамічні, що чимось нагадують сам рекламний відеоролик [13].

Реєстрація у соціальних мережах.

Для мобільного розробленого застосунку було створено обліковий запис у популярній соціальній мережі Twitter.

Крім реєстрації, проведено роботу з пошуку учасників групи, до яких було доведено інформацію про створення сайту, на якому користувачі можуть ознайомитися з можливостями застосунку, а також оновлювалася стрічка записів з новинами про компанію.

Комплекс проведених безкоштовних заходів дозволить через деякий час оцінити ефективність просування за допомогою впроваджених на сайт лічильників статистики. Отримана інформація допоможе відібрати інструменти, які придять на сайт найбільше відвідувачів. Крім того, статистичні

дані дозволять визначити стратегічний напрямок діяльності компанії в мережі.

Реєстрація сайту у безкоштовних каталогах.

При завантаженні сайт автоматично проходить реєстрацію у найпопулярніших каталогах Інтернет. Однак, чи будуть опубліковані результати реєстрації, стане відомо як мінімум через 1 місяць.

Після того, як застосунок створено та розміщено в мережі, про нього необхідно повідомити потенційних споживачів. На початку існування користувача більшість відвідувачів потрапляють на сторінки через пошукові системи. Для того, щоб програма почала виводитися за тими чи іншими пошуковими запитами, вона має бути проіндексована пошуковими роботами та занесена до їх баз даних магазину застосунків. Зазвичай на це йде від двох тижнів і більше. Підняти рейтинг у пошуковій видачі та підвищити скачування програми можна кількома способами, а саме, безкоштовними та платними, кожен з яких має плюси та мінуси. Зосередимося на безкоштовних інструментах просування.

4.4 Безкоштовні інструменти просування

1. Реєстрація в керуючих та аналізуючих системах.

Після розміщення програми в маркеті необхідно почекати, щоб динаміка установок у Play Market повністю оновилася за 48-72 години. Але просто завантаження програми мало, необхідно просувати програму, для цього необхідно скористатися спеціальним сервісом основних пошуковиків, таких як Google.

У Google сервіс для індексації програми називається Google analytics - це безкоштовний інструмент, який у реальному часі збирає інформацію про джерела аудиторії, аналізує її поведінку у додатку та фіксує помилки, з якими стикаються наші користувачі. Як тільки завантажується програма в Play Market сервіс автоматично встановлюється і веде статистику [12].

2. Реєстрація сайту в каталогах.

Важливим інструментом безкоштовного просування мобільного застосунку є розміщення його у каталогах. Для того, щоб наростити цільову аудиторію програми.

Найбільш популярними каталогами в Інтернеті прийнято вважати:

- Google.Каталог;

- ProAndroid.net;
- 4PDA.com;
- Androtop.com.

Як відомо, коли користувачі шукають інформацію в мережі, насамперед вони звертаються до пошукових систем. Саме пошукові системи дозволяють залучати безкоштовних цільових відвідувачів на сайт програми та на сам магазин.

Розглянемо список безкоштовних інструментів просування програми через сайт.

1. Keyword Tool.

Ось деякі з переваг його використання.

- безкоштовна версія Keyword Tool генерує до 750+ ключових слів для кожного пошукового запиту;
- стабільність. Keyword Tool - це дуже надійний інструмент, який працює у 99,99% випадків;
- не потребує облікового запису.

Подібні рішення представляють такі відомі системи, як Google AdWords та SeoPult. Кожна з них має свій унікальний інтерфейс та певні налаштування.

Google AdWords безкоштовно проводить добір ключових слів за адресою URL, за фразою або за категорією. Дозволяє відстежити статистику підібраних запитів, виявляє рівень конкуренції та визначає чи можливо отримувати більше кліків за тим чи іншим запитом [12].

2. Аналіз контенту сторінки та її виділення.

Ще одним способом підвищення відвідуваності сайту є текстові критерії. Важливо визначити чи є на сторінці ключові слова та їх співвідношення із загальною кількістю слів – це називається щільністю ключових слів. Її оптимальне значення прийнято вважати рівним 5%. Ключові фрази, що присутні на сторінці, радять розташовувати ближче до початку тексту і акцентувати увагу на них спеціальними способами, наприклад, присвоювати їм заголовки (h1-h6) або виділяти напівжирним або курсивним шрифтом.

3. Розміщення сайту на дошках оголошень.

Дошки оголошень - це електронні майданчики, які дозволяють безкоштовно рекламувати сайт. Такі реклама сприятливо позначиться на видачу сторінок по пошуковим запитам у разі, якщо розміщувати оголошення

на кількох десятках майданчиків, так як їх також індексують пошукові роботи, що підвищує можливість видачі на запит. Це своє чергу привабить потенційних клієнтів, бо зазвичай читачі там опиняються невипадково, а заходять цілеспрямовано для пошуку необхідної інформації.

Комплекс проведених безкоштовних заходів дозволить через деякий період оцінити ефективність просування за допомогою впроваджених у застосунок лічильників статистики. Отримана інформація допоможе відібрати інструменти, які показують, скільки людей скачало програму та відвідало сайт.

Перелічені інструменти допоможуть привести користувачів до магазину програм. Головний плюс таких способів просування – відсутність прямих фінансових витрат. Однак, тимчасові витрати дуже високі, оскільки потрібен постійний контроль з боку оптимізатора, і є основним недоліком.

4.5 Інструменти моніторингу відвідуваності мобільного застосунку

Важлива характеристика мобільних програм – це їх частота завантажень, а особливо для мобільних програм комерційної спрямованості. Достовірна, актуальна статистика дозволяє простежити попит Інтернет-аудиторії загалом до застосунку. Також статистичні дані зможуть допомогти проаналізувати ефективність рекламної кампанії, що застосовується, і визначитися зі стратегією просування та подальшого розвитку. Для якісного проведення аудиту сайту створено спеціальні інструменти, що дозволяють відстежувати завантаження програми в даний час.

Розглянемо ці інструменти.

1. Seesmic - безкоштовний популярний сервіс для моніторингу програм. Підтримує моніторинг таких торгових майданчиків: Google Play та AppStore. Вказує, наскільки збільшилися або зменшилися завантаження програми [13].

2. YouScan – перша та лідируюча система для професійного моніторингу російськомовних соціальних медіа. YouScan відслідковує згадки ваших брендів, продуктів, конкурентів у блогах, форумах, соціальних мережах Facebook, Twitter і навіть у YouTube, і надає результати моніторингу у зручному аналітичному інтерфейсі з функціями командної роботи [12].

YouScan пропонує 5 тарифних пакетів, у тому числі один безкоштовний.

- Freemium. безкоштовно: кількість тем - 1; користувачів – 2;

- базовий. $1 990 на рік: кількість тем - 5; користувачів – 5;

- професійний. $349 на місяць: кількість тем – 10; користувачів – 10;

- корпоративний. $799 на місяць: кількість тем - 25; користувачів – 20;

- Агентство. $1499 на місяць: кількість тем - 50; користувачів - 40.

Можна безкоштовно скористатися повнофункціональним сервісом 14 днів.

3. HPE AppPulse Mobile – це інструмент для моніторингу продуктивності мобільних програм, який дозволяє відстежувати реальну взаємодію з користувачами. Фахівці з технічного обслуговування отримують цінні з практичної точки зору дані для пріоритезації труднощів, що заважають досягненню максимальної продуктивності.

- пробний тариф:
    - залучення активних користувачів за місяць –2500 тис.;
    - зберігання даних-30 днів;
    - постійний моніторинг-12 годин;
- експрес 99$/міс:
    - залучення активних користувачів за місяць-10000 тис.;
    - кількість додатків-необмежену;
    - зберігання даних-30 днів;
    - постійний моніторинг-18 годин/д.;
- преміум 499$/міс:
    - залучення активних користувачів за місяць-25000 тис.;
    - кількість додатків-необмежену;
    - зберігання даних-1 рік;
    - відкритий API (для отримання даних)-постійний аналіз;
    - наявність IT-операцій з управління інтеграцією;
    - постійний моніторинг-24 години [13].

Для якісного проведення аудиту сайту взято інструмент моніторингу YouScan, тому що він дозволяє грамотно відстежувати реальну взаємодію з користувачами не тільки в маркеті застосунків, а й у різних соціальних мережах.

ВИСНОВКИ

В даній кваліфікаційній роботі розглянуто мережі, операційну систему та середовище для розробки. Було проведено велику дослідницьку роботу за принципами реалізації та створення програми для перехоплення пакетів на операційній системі Андроїд.

Результатом роботи є розроблення застосунку: на операційній системі андроїда; на двох мовах програмування: Java та Kotlin: з повною підтримкою усіх версій, навіть починаючи із 5ї версії андроїда; відкритий код, який розташовано на GitHub: маються тести для UI та для UNIT тести. Більша частина кода описана в коментарях, що і пояснює працездатність деяких компонентів.

Зроблено повноцінний проєкт, у якому використовується безліч новітніх технологій. Застосунок безпосередньо протестовано та він повноцінно сумісний із будь-якими діагоналями та будь-якими мобільними пристроями.

На основі розглянутої інформації у четвертому розділі були вироблені рекомендації для просування мобільного застосунку, що сприяють залученню нових користувачів. Також були описані заходи щодо просування мобільного застосунку, досліджено безкоштовні методи просування мобільного застосунку, обрано інструменти моніторингу відвідуваності застосунку.

Як додаткові рекомендації щодо просування мобільного застосунку можна запропонувати такі опції:

- широке використання рекламних ресурсів;
- збільшення бюджету рекламної діяльності.

Ці рекомендації допоможуть підвищити популярність програми шляхом залучення нових користувачів.

Практична значимість випускної кваліфікаційної роботи полягає у розробці комплексу заходів щодо створення та просування мобільного застосунку для збільшення кількості його потенційних відвідувачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Джеймс Гослинг, Билл Джой, Гай Стил, Гилад Брача, Алекс Бакли. Язык программирования Java SE 8. Подробное описание / Диалектика Вильямс. – 2021. – 672с.

2. Оліфер В. Г. Комп'ютерні мережі. Принципи, технології, протоколи : підручник / В. Г. Оліфер, Н. А. Оліфер. – СПб. – 2016.

3. Таненбаум Е. Комп'ютерні мережі / Е. Таненбаум, Д. Уезеролл, 5-е видання, 2012. – 960с.

4. Девід Гріффітс. Програмування для Android / Девід Гріффітс, Дон Гріффітс. Head First. – 2019.

5. Frank Ableson. Development your first Android application. – 2018. Режим доступу https://developer.ibm.com/tutorials/develop-android-applications-with-android-studio/

6. Жемеров Д. Kotlin в дії / Дмитро Жемеров, Світлана Ісакова // Розвиток освіти, науки та бізнесу: результати 2020: тези доп. міжнародної науково-практичної інтернет-конференції, 3-4 грудня 2020 р. – Україна, Дніпро, 2020. – Т.2. – 596 с.

7. Барі Бурда, Джон Пол Мюллер. Android Application Development All-in-One. – 2020.

8. Джош Скин. Kotlin. Программирование для профессионалов Пер. с англ. – СПб: «Питер». – 2020. – 464с.

9. Вікторія Гонда, Фернандо Спровієро, Ленс Глісон. Android, Test-Driven Development. – 2019.

10. Ховард М., Леви М., Вэймир Р. Разработка защищенных Web-приложений: Пер. с англ. – СПб: «Питер». – 2001.

11. Sean Peek. What Is Mobile App Development? – 2022. Режим доступу https://www.businessnewsdaily.com/5155-mobile-app-development.html

12. Вроблевський Люк. Спершу мобільні! Видавництво: Манн, Іванов та Фербер, 2012, - 214 с.

13. Мобільні телекомунікації. Журнал. Випуск №06-07, 2014, Вересень-Жовтень.