Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інформаційних технологій та комп'ютерної інженерії
(повне найменування інституту)

Кафедра обчислювальної техніки
(повна назва кафедри)

Пояснювальна записка

до магістерської кваліфікаційної роботи

магістр

(освітньо-кваліфікаційний рівень)

на тему: Програмний засіб батьківського контролю на платформі Android

Виконала: студентка 2 курсу, групи КІ —19м
спеціальності:
123 «Комп'ютерна інженерія»
(шифр і назва напряму підготовки)

Стягайло І. В.
(прізвище та ініціали)

Керівник: к.т.н., доц. Богомолов С. В.
(прізвище та ініціали)

м. Вінниця — 2020 рік

АНОТАЦІЯ

Магістерська кваліфікаційна робота присвячена дослідженню та розробці програмного засобу батьківського контролю на платформі Android. Продукт полегшує процес контролю за місцеперебуванням дитини батьками, а також є корисним інформаційним додатком для самої дитини, адже виконує функцію візуалізації розпорядку робочого дня в просторі.

Під час роботи над проєктом проведений огляд та аналіз методів та засобів, які використовуються для розробки мобільних застосунків. Для реалізації програмного засобу обрано мову програмування Java та середовище програмування Android Studio. Проведено оцінювання доцільності розробки на основі огляду наявних аналогів.

Результатом магістерської кваліфікаційної роботи є програмний засіб, який наділений інтуїтивно зрозумілим інтерфейсом користувача, а також функціоналом, який забезпечує коректний збір, обробку та виведення інформації та надає можливість контролювати підозрілу активність дитини без зайвих на те затрат часу зі сторони батьків.

Відповідність реалізованих характеристик застосунку до заявлених підтверджено за допомогою ручного тестування. Також проведено економічні розрахунки, що дають можливість зробити висновок про доцільність розробки.

ANNOTATION

The master's thesis is devoted to the research and development of software for parental control on the Android platform. The product facilitates the process of monitoring the child's whereabouts by parents and is also a useful information application for the child because it performs the function of visualizing the daily routine in space.

While working on the project, a review, and analysis of methods and tools used to develop mobile applications. The Java programming language and Android Studio programming environment were chosen for the implementation of the software. An assessment of the feasibility of development based on a review of existing analogs.

The result of the master's qualification work is a software that is endowed with an intuitive user interface, as well as functionality that provides correct collection, processing, and output of information and allows you to control the suspicious activity of the child without wasting time on the part of parents.

The conformity of the implemented application characteristics to the declared ones was confirmed by means of manual testing. Economic calculations were also made, which make it possible to draw a conclusion about the feasibility of development.

ЗМІСТ

ВСТУП

Проблема зникнення дітей є актуальною у всьому світі. Зокрема за даними служби розшуку дітей в Україні 32 особи, які не досягли повноліття вважаються зниклими безвісти. Також вважається, якщо дитину не знайдено за перші години або дні, то ймовірність знаходження її живою різко знижується. Причинами зникнення дітей є як конфліктні ситуації з батьками та рідними, так і причини, коли зникнення особи відбувається проти її волі та із застосуванням насильницьких дій [1]. Така сумна статистика змушує задуматися над проблемою та мотивує до пошуків її вирішення.

Зважаючи на те, що популярність смартфонів безперервно зростає і вони є наявними як в дорослих, так і в дітей, то доцільно скористатися цією тенденцією. Тому одним із можливих варіантів розв'язку поставленої задачі є використання смартфону, як засобу, на базі якого відбувається забезпечення постійного контролю над місцеперебуванням дитини протягом дня та оперативного сповіщення батьків про підозрілу активність дитини. Програмний засіб батьківського контролю реалізується у вигляді нативного додатка, зважаючи на переконливі переваги саме цього види мобільних додатків.

Нативні додатки оптимізовані під конкретну платформу, завдяки чому їх робота є швидкою та стабільною. Також вони можуть використовувати усі можливості операційної системи (ОС) та функціонал пристрою, такий як GPS, що є ключовим для реалізації програмного засобу батьківського контролю на платформі Android, що представлена в дипломній роботі. Дизайн нативних додатків є інтуїтивно зрозумілим та таким, що не вимагає великих затрат для освоєння. Анімація, кольорова гама та положення елементів керування є схожими та не дисонують з набутим досвідом користувача [2].

Програмний засіб батьківського контролю являтиме собою застосунок для смартфону, який буде доступний для завантаження з маркету Google Play. Загальна схема роботи додатку полягає в наступному: функціонал притаманний пристрою дитини та дорослого реалізований в одному застосунку. В залежності від обраного того чи іншого режиму (дитина, батьки) на екрані смартфону

відображається різний дизайн та стають доступні різні функції. У режимі батьківського контролю доступні функції вибору точок на карті, додатково вказавши радіус кола, за який дитина не повинна виходити у проміжок часу, вказаний батьками.

У випадку, коли місцеперебування дитини не збігається із зазначеним, на смартфон батьків надходить сповіщення та стає доступна поточна геолокація дитини. Також для забезпечення конфіденційності даних в даному режимі відбувається авторизація при вході, що унеможливлює виконання несанкціонованих дій сторонніми особами. Що стосується дитячого режиму, то існує можливість бачити на карті власне місцеперебування, а також дозволений периметр пересування з часовим діапазоном, що своєю чергою може слугувати таким собі розпорядком дня у просторі.

Метою дослідження магістерської роботи є покращення методів контролю за місцеперебування користувача, завдяки використанню сучасних можливостей синхронізації даних та підходів визначення точної геолокації.

Задачі дослідження магістерської роботи:

— проаналізувати та обрати засоби, необхідні для розробки;

— дослідити ринок на наявні аналоги та провести порівняння;

— проаналізувати та обрати тип системи управління базами даних;

— створити дизайн програмного засобу та окреслити його функціонал;

— прописати головну логіку застосунку;

— виконати тестування продукту на відповідність до заявлених можливостей.

Об’єктом дослідження магістерської роботи є процес обробки зібраних даних та порівняння їх із допустимими з подальшим реагуванням в залежності від результатів їх опрацювання.

Предмет дослідження магістерської роботи — методи роботи з різноманітними сервісами Google та вбудованими можливостями мобільного пристрою, з метою сповіщення користувача про розбіжність між заздалегідь вказаними даними та наявними.

Методи дослідження магістерської роботи: використано принципи об'єктно-орієнтованого програмування, принципи роботи з базою даних та методи авторизації користувача для реалізації програмного засобу.

Наукова новизна отриманих результатів магістерської роботи полягає у реалізації розширеного функціоналу в порівнянні з наявними аналогами на ринку, а також підвищенню швидкості автоматичного сповіщення користувача, що дозволяє ефективно запобігати проблемі зникнення дітей.

Практичне значення одержаних результатів магістерської роботи: розроблено програмний засіб батьківського контролю на платформі Android.

Апробація результатів дослідження була здійснена у процесі подання тези доповіді, яка була опублікована:

Стягайло І. В., «Програмний засіб батьківського контролю на платформі Android» в Матеріали конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2021)», Вінниця, 2020. [Електронний ресурс]. Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/viewFile/10982/9181 Дата звернення: Листопад 2020.

1 ОГЛЯД ТА АНАЛІЗ СУЧАСНИХ МЕТОДІВ ТА ЗАСОБІВ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ЗАСОБУ БАТЬКІВСЬКОГО КОНТРОЛЮ

Розробка програмного забезпечення (ПЗ) для мобільних пристроїв займає сьогодні досить високі позиції. Головною особливістю таких результатів спостереження є те, що сучасні смартфони все більше накопичують у собі функціонал персонального комп'ютера (ПК). Разом з тим вони є більш мобільними, менш габаритними, їх ціни кардинально відрізняються від цін на ПК і не дивлячись на все це середньостатистичний користувач без потреби у спеціалізованих програмах та високій продуктивності може обійтися використанням лише смартфону у звичному житті.

Актуальність розробки ПЗ також проілюстрована тим, що користувач використовує все більше застосунків, які задовольняють найрізноманітніші сфери людського життя. Ігри різних типів, форм та смислів, відтворення, створення та обробки мультимедійного контенту (фото, відео, аудіо), ведення фінансів, обробки текстових документів, керування «розумним» будинком, відстежування показників власного організму, використання навчальних платформ та багато іншого — це все може вміщатися в одному ґаджеті, але й це все потребує необхідних знань та навичок, а також відповідних засобів для створення додатків.

1.1 Вибір операційної системи

Перш ніж почати роботу над проєктом необхідно обрати платформу, на якій він буде реалізований. Не зважаючи на бурхливий розвиток історії створення мобільних операційних систем, у сучасних реаліях існує лише два стабільних варіанти, що користуються неабиякою популярністю (рисунок 1.1).

Android — це мобільна операційна система, заснована на модифікованій версії ядра Linux та іншого програмного забезпечення з відкритим кодом, призначена в основному для сенсорних мобільних пристроїв, таких як смартфони та планшети. Це безкоштовне програмне забезпечення з відкритим кодом; його вихідний код відомий як Android Open Source Project, який в основному ліцензується за ліцензією Apache.

Year	2018	2019	2020	2021	2022	2023	2024
Android	85.1%	86.1%	85.0%	85.6%	86.0%	86.1%	86.2%
iOS	14.9%	13.9%	15.5%	14.4%	14.0%	13.9%	13.8%
Others	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
TOTAL	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**

Рисунок 1.1 — Перспективи розподілу частки ринку ОС

Однак більшість пристроїв Android постачаються з попередньо встановленим програмним забезпеченням, зокрема Google Mobile Services (GMS), яке включає такі основні програми, як Google Chrome, платформа цифрового розповсюдження Google Play та пов'язана з ними платформа розробки служб Google Play.

До переваг ОС Android можна зарахувати наступні пункти:

— додатки класифікуються за темами, і кожна тема має велику кількість програм для завантаження;

— можливість встановлювати сторонні програми, тобто користувач гнучко обирає, які додатки завантажити з маркету Google Play, а які з будь-якого іншого вебсайту, просто авторизувавши додаток для завантаження;

— дана ОС може працювати на широкому діапазоні пристроїв, що своєю чергою дає можливість обрати смартфон у різних цінових категоріях;

— підтримка опції розширеної пам’яті, за потреби для зберігання, можна під'єднати зовнішню карту пам’яті;

— можливість відновлювати та створювати резервні копії своїх даних, ця функція допомагає програмам відновлюватися після будь-якого неякісного оновлення або некоректної поведінки користувача;

— швидка та гнучка інтеграція з сервісами Google;

— підтримка плавного одночасного запуску програм;

— використання власних або сторонніх віджетів, підтримка кількох віджетів одночасно;

— підтримка масштабної спільноти як розробників, так і користувачів;

— використання хмарних технологій, що надає можливість синхронізуватикілька пристроїв за допомогою одного облікового запису.

Що ж стосується недоліків, то вони наступні:

— Google суворо регламентує критерії, яким має відповідати додаток, при виявлених будь-яких проблем можливе припинення роботи облікового запису розробника;

— низька якість деяких програм з метою заробітку лише через показ реклами, деякі програми не оновлюються протягом багатьох років, і це впливає на користувачів, які оновлюють свою операційну систему;

— труднощі виходу нових компаній та розробників на ринок Android, оскільки шанс реалізації абсолютно нової ідеї дуже низький;

— необхідна наявність облікового запису Google та Gmail;

— велика кількість реклами та сповіщень, які ускладнюють взаємодію з мобільним додатком;

— низький рівень захисту від різних типів вірусів;

— значний обсяг операційної системи, що може слугувати причинами повільної роботи та нагрівання мобільних пристроїв з низькими характеристиками [3].

iOS — це мобільна операційна система, створена та розроблена Apple Inc. виключно для свого обладнання. Саме ця операційна система керує багатьма мобільними пристроями компанії, включаючи iPhone та iPod Touch; вона також працювала на iPad до впровадження iPadOS, похідного від iOS, у 2019 році. Це друга за частотою встановлення мобільна операційна система у світі після Android.

До переваг iOS можна віднести:

— iOS має простий інтерфейс, що значно спрощує користування пристроєм при переході від старішої версії до новішої;

— ідеально підходить для розробників додатків, позаяк у порівнянні зAndroid має мало пристроїв, що зменшує кількість проблем, що виникають при розробці для різних розмірів екрану;

— усі ігри повинні проходити через групи затвердження додатків. Затверджені лише ті ігри, в яких немає проблем у коді і які можуть безперебійно працювати на пристрої;

— з кожними новими версіями iOS загальна продуктивність покращується;

— оскільки додатки в маркеті Apple Store повинні пройти суворий процес затвердження, то і безпека гарантується на найвищому рівні, що майже унеможливлює потрапляння вірусів до пристрою;

— операційна система підтримує багато жестів, що робить досвід користування пристроєм легшим.

До недоліків iOS відносяться:

— у порівнянні з Android більшість додатків є платними;

— відсутня підтримка віджетів для програм;

— iOS не є відкритим кодом, не можливо налаштувати і використовувати iOS на пристроях, відмінних від пристроїв Apple;

— додатки в iOS мають великий розмір файлів та займають багато місця на пристроях; більшість ігрових додатків мають розмір завантаження в ГБ;

— відсутність можливості збільшення обсягу пам’яті шляхом під’єднання SD-карти;

— одноманітність дизайну та стилю іконок не залежачи від версії ОС;

— iOS не надає підтримку радіо, для його використання необхідне завантаження програми [4].

Зважаючи на вище описані переваги та недоліки операційних систем, було обрано ОС Android як базу для створення додатку, оскільки деякі плюси стали ключовими.

Операційна система Android є повністю відкритою, на пристрої можна зберігати абсолютно різні типи файлів, завантажувати їх з мережі і навіть ділитися ними через месенджери.

На жаль, власники iOS можуть пересилати тільки фотографії або файли з iCloud Drive (починаючи з iOS 9), але завантаження файлів з мережі стандартними методами недоступне.

Власники пристроїв на базі Android можуть завантажувати з Інтернету або Google Play програми, що дозволяють змінювати різні параметри та втручатися в роботу ОС, підвищуючи її функціонал або прибираючи непотрібні функції.

Робота додатків у фоновому режимі на iOS суттєво обмежена. Не можна запустити якийсь менеджер завантажень, приховати його та користуватися іншою програмою. В такому випадку завантаження через деякий час перерветься. Власники Android можуть запустити будь-які програми у фоновому режимі й вони будуть залишатися активними до тих пір, поки в пристрої не закінчиться вільна оперативна пам'ять або процес завантаження не підійде до кінця. Щоб домогтися такого функціоналу на iOS доводиться тримати програму активною доти, поки файл повністю не завантажиться з мережі на пристрій.

Під’єднати флешку, дротову клавіатуру або мишку до iPhone або iPad досить важко, адже вибір спеціальних аксесуарів не великий. Щоб під’єднати практично будь-який периферійний пристрій від комп'ютера до Android необхідний лише OTG-кабель.

Обираючи між операційною системою Android та iOS слід також розглянути географічні та демографічні характеристики цільової аудиторії. Наприклад, Android-пристрої мають найбільшу частку світового ринку, тому орієнтуючись на широку аудиторію, Android є кращим вибором

Але головною причиною реалізації проєкту на платформі Android є те, що за відсутності пристрою Apple процес розробки та тестування програмного засобу на платформі iOS значно ускладнюється, а в деяких випадках і зовсім стає неможливим.

1.2 Огляд наявних аналогів програмних засобів батьківського контролю на платформі Android

Визначившись з операційною системою, на основі якої буде розроблено програмний засіб, необхідно зробити огляд уже наявних аналогів з подібним функціоналом. Для пошуку таких застосунків використовуємо маркет Google Play, що дозволяє власникам пристроїв з мобільною ОС Android завантажувати та

купувати різні застосунки, книги, фільми й музику [5]. Щоб розширити межі пошуку використовуємо ключові слова не лише українською мовою, а й англійською, на приклад: kids, children (діти), parents (батьки), control (контроль), tracker (трекер), find (шукати), link (зв'язок) та подібні. У результаті пошуку маємо наступні застосунки: KidControl, Google Family Link, Find My Kids (рисунок 1.2).

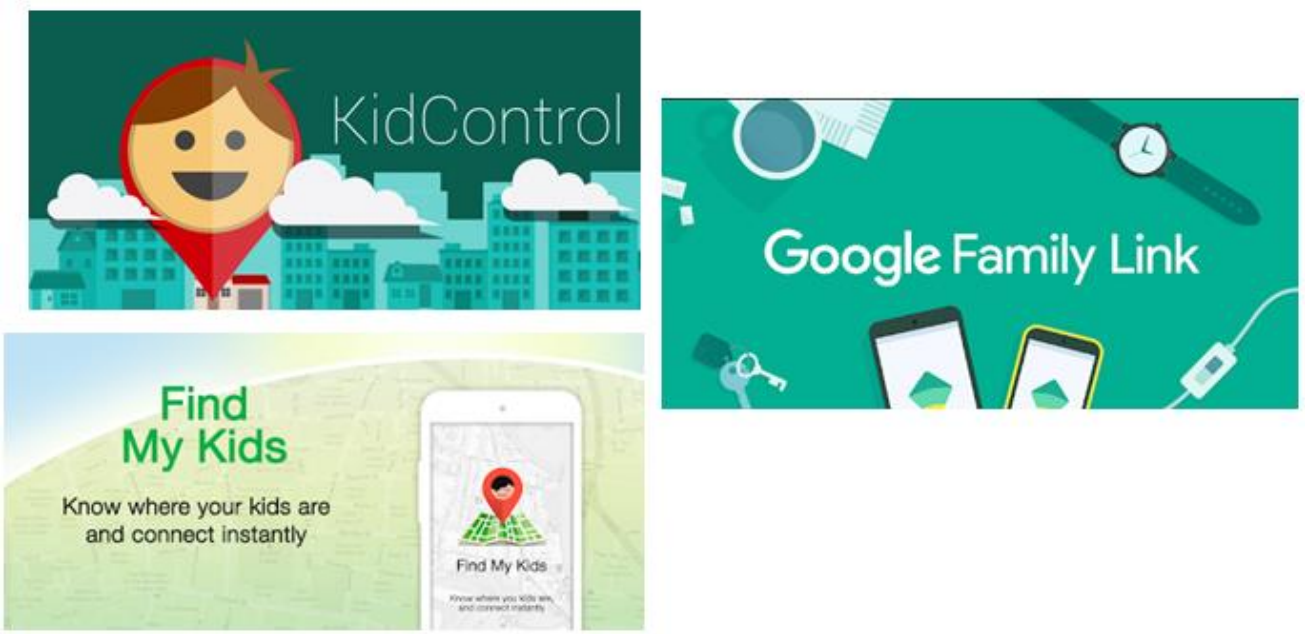


Рисунок 1.2 — Наявні аналоги додатку

Для оцінки доцільності розробки автора використано евристичний метод, що є концепціє інспектування ПЗ. За його допомогою ідентифікуються проблеми дизайну інтерфейсу користувача. Даний метод залучає оцінювачів дослідити інтерфейс та оцінити його відповідність до визначених принципів користувацьких інтерфейсів.

На ранніх стадіях дизайну простота використання евристичного оцінювання є доволі вигідним вибором, оскільки не вимагає велику кількість користувачів, місця проведення та оплати часу. Що стосується оптимальної кількості експертів при евристичному оцінюванні, то це команда з 4 до 6 незалежних експертів. Розподіл балів розраховується наступним чином: 1 — найнижча оцінка, 5 — найвища [6]. Евристичний метод оцінювання наведений в таблиці 1.1.

Таблиця 1.1 — Евристичний метод оцінки

	KidControl	Google Family Link	Find my Kids	Розробка автора	Ваговий коефіцієнт
Легкість синхронізації пристроїв	3	3	4	4	0,8
Відсутність обтяжуючого функціоналу	2	4	4	3	0,5
Локалізація	3	4	4	5	1
Ціна додатку	5	5	5	5	0,9
Сума балів	10,9	12,9	13,7	14,2	

За результатами оцінки можна зробити результати, що доцільність розробки є виправданою. Також необхідно зазначити ключові моменти, які вказують на недосконаліть наявних аналогів. По-перше, перенасичення додатку непотрібними функціями. Застосунки можуть також контролювати витрачений час дітей за різними програмами, показувати батькам заряд мобільного пристрою, маршрут пройдений дитиною. Увесь такий функціонал забезпечує надмірний контроль.

По-друге, додатки в змозі показувати поточне місцеперебування дитини, але у той же час не здатні сповіщати батьків про порушення меж дозволеної території, що змушує батьків самостійно стежити за пересуванням дитини, замість того, щоб лише в конкретних ситуаціях отримувати повідомлення. По-третє, відсутність можливості планування дня заздалегідь.

Отже, під час розробки програмного засобу батьківського контролю, будуть враховані усі недоліки попередників та за можливості буде доданий лише необхідний функціонал.

1.3 Вибір інтегрованого середовища розробки

Під інтегрованим середовищем розробки (англ. Integrated development environment — IDE) зазвичай розуміють комплексний засіб, що містить усі необхідні елементи для створення програмного забезпечення. Існує певний набір компонентів, які повинні бути присутніми в інтегрованих середовищах розробки. По-перше, це компілятор або інтерпретатор, по-друге — редактор вихідного коду програм (обов'язково хоча б з підтримкою підсвічування синтаксису тієї мови програмування, для якої призначене середовище), ну а по-третє — відлагоджувач (англ. debbuger). Відлагоджувач — це, мабуть, навіть більш істотна частина інтегрованого середовища розробки, ніж компілятор або інтерпретатор, оскільки нерідко саме налагодження програми стає найскладнішим етапом її створення. Звичайно, сучасні інтегровані середовища розробки пропонують набагато більше можливостей, ніж входять в описаний вище необхідний мінімум. Наприклад, багато сучасних IDE є візуальними — вони дозволяють створювати інтерфейс програми за допомогою мишки, точно в такому вигляді, в якому він постане потім користувачеві. IDE, які не є візуальними, вимагають від розробника написання спеціального коду, відповідального за створення призначеного для користувача інтерфейсу програми. Залежно від того, для яких платформ можна писати програми й на яких платформах працює саме IDE, середовища розробки поділяються на крос-платформні (підтримують роботу з різними платформами) або платформно-залежні (ті, які працюють тільки з однією платформою). Залежно від кількості підтримуваних мов програмування, середовища можуть бути багатомовними або одномовними. Список популярних середовищ розробки великий, і всі значущі продукти цього класу динамічно розвиваються у бік все більшої зручності для розробників. Для створення програмного забезпечення на базі ОС Android використовують наступні інтегровані середовища розробки: Android Studio, Eclipse, Visual Studio (Xamarin), IntelligJ IDEA, NetBeans, Komodo, Cordova, PhoneGap, Appcelerator Titanium, App Inventor, AIDE [7].

У якості інтегрованого середовища розробки для реалізації проєкту магістерської кваліфікаційної роботи було обрано Android Studio оскільки

починаючи з 8 грудня 2014 року саме це IDE було визнане компанією Google офіційним середовищем розробки під ОС Android. Воно замінило попередні інструменти для розробки Android на базі Eclipse (ADT) як краще IDE для платформи. Згідно з останньою версією списку IDE на індексі PYPL (PopularitY of Programming Language Index), Android Studio є третім за популярністю IDE у світі з часткою 12.57%. Це стандартний варіант для розробників Android, що легко інтегрується з Google Cloud Platform.

Розробник / власник: Google.

Ключові риси:

— миттєвий запуск;

— інтелектуальний редактор коду;

— емулятор Android;

— гнучка система побудови на основі Gradle;

— оптимізація пристроїв Android;

— інтеграція з GitHub, Subversion та іншими системами контролю версій;

— шаблони кодів;

— зразки додатків;

— комплексне тестування;

— редактор розкладки;

— студія Vecor Asset;

— аналізатор APK;

— редактор перекладів.

Підтримувані мови програмування: Java, C, C ++, Kotlin.

Цільові операційні системи: Android.

Працює на: Windows, MacOS, Linux.

Системні вимоги:

— Windows 7 або пізнішої версії, macOS 10.10 або пізнішої, або 64-розрядний Linux з робочим столом GNOME або KDE та бібліотекою GNU C;

— ОЗП об’ємом 3 ГБ плюс 1 ГБ для емулятора Android (рек. 8 ГБ);

— 2 Гб дискового простору (рекомендовано 4 ГБ);

— 1280 x 800 або вище розширення екрана.

Цільовою аудиторією можуть бути будь-які розробники Android, але найкращі для досвідчених розробників. Перший випуск продукції припадає на грудень 2014 року. Останнє оновлення та стабільний випуск продемонстровано версією 4.0.1 (14 липня 2020). Ліцензія продукту — Freeware, тобто продукт надається безкоштовно.

Наступні пункти ілюструють одні з головних особливостей даного середовища розробки, що стали ключовими під час його вибору.

Gradle дуже потужний інструмент збору проєктів. Найбільш поширеним варіантом використання є зв'язування бібліотеки, написавши рядок у файлі build.gradle на рівні програми. Однією з переваг системи є швидкість програми, яка збільшується з кожною новою версією. Постійне оновлення бібліотек та зміна окремих модулів займає багато часу, щоб знайти помилку в одному з файлів, гнучкість змінних gradle дозволяє уникнути цієї ситуації. У програмах, які розробляються, часто використовуються сторонні бібліотеки, які містять різні внутрішні ресурси, які нашим додаткам взагалі не потрібні. Завдяки плагіну Android Gradle є можливість встановити мови та розширення, що використовуються в додатку; інші видаляються, зменшуючи розмір файлу.

Вбудований SDK (англ. software development kit), є багато середовищ розробки з вбудованим Android SDK, але вони додаються не так щільно, як у студії. Наприклад, запускається старий проєкт із низьким рівнем API, який просто не завантажили. У результаті кількість помилок у проєкті зростає і зрозуміло, що чогось не вистачає. Подібна ситуація не виникає при використанні студії. Якщо використовується елемент, який не існує, студія визначить, де і чого не вистачає, і покаже вікно повідомлення. Крім того, всі необхідні елементи можна встановити за лічені секунди.

Зручний інтерфейс, на перший погляд, він нічим не відрізняється від такого ж в Eclipse. Однак одним клацанням миші можна переглядати екран на будь-якому пристрої від смартфонів різних брендів й аж до телевізорів та годинників. Самі елементи інтерфейсу, на відміну від інших середовищ розробки, у яких

віджети показуються у всіх версіях як одне стандартне зображення, відображаються точно так, як вони виглядатимуть у певній версії операційної системи (рисунок 1.3).

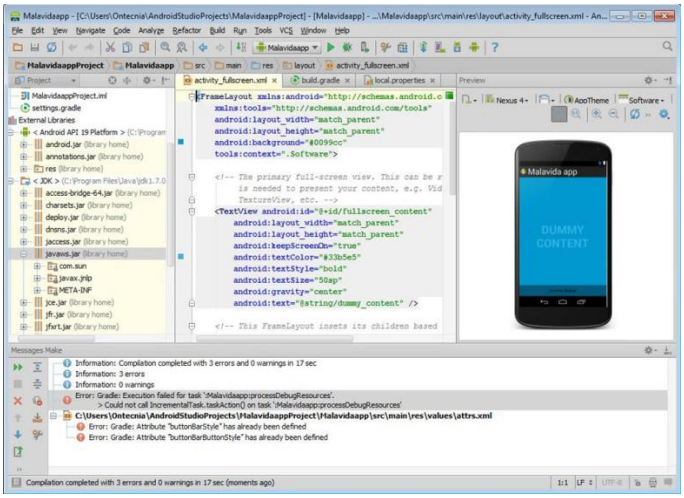


Рисунок 1.3 — Інтерфейс Android Studio

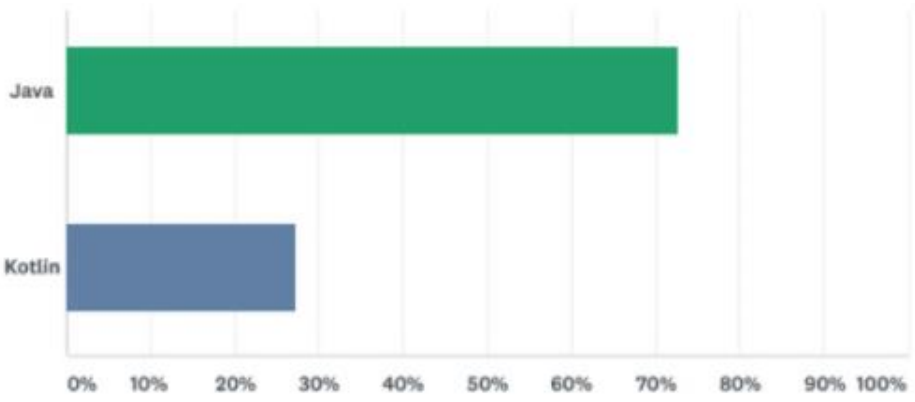
Зручний дизайн. Дуже велику частину простору займають вікна повідомлень, властивості елементів та сам проєкт. Однак для кожного вікна існує функція згортання в одну з частин інтерфейсу. Більше не доводиться закривати вікна без потреби, всі вікна знаходяться в одному місці в системній області [8].

1.4 Вибір мови програмування для розробки додатку

Ще одним досить важливим підготовчим кроком для створення Android-застосунку є вибір мови програмування. Сьогодні в основному додатки створюються за допомогою Java, але й існують інші мови програмування, які можуть позмагатися за лідерством в даному сегменті. Одна з таких мов є Kotlin. Слід детально розглянути, що собою являє та чи інша мова програмування, щоб підійти до питання вибору однієї з них більш аргументовано, зважаючи на переваги та недоліки кожної з них.

Java вважається однією з кращих мов для розробки додатків. Це строго типізована об'єктноорієнтована мова програмування. Kotlin — статично типізована мова програмування з відкритим вихідним кодом. Може ефективно запускатися на віртуальній машині Java. Kotlin розроблений JetBrains і офіційно

підтримується Google. Недавнє опитування Jexenter помістив Kotlin на шосте місце серед технологічних трендів (рисунок 1.4).



ANSWER CHOICES	RESPONSES
Java	72.83%
Kotlin	27.17%
TOTAL	

Рисунок 1.4 — Результат опитування розробників на платформі Android

Недавній TIOBE index (липень 2020) показав, що Java посідає друге місце серед топових мов програмування. Своєю чергою Kotlin займає 12 позицію у тому ж TIOBE index.

Переваги Java:

— проста у вивченні та розумінні;

— крос-платформеність;

— Android базується на Java — Android SDK налічує багато стандартних Java бібліотек;

— Java має величезну open-source (відкриті джерела) екосистему;

— прискорена збірка за допомогою Gradle;

— Java додатки дещо компактніші в порівнянні з написаними на Kotlin.

Недоліки Java:

— обмеження Java створюють проблеми з архітектурою Android API;

— будучи дуже докладною мовою (verbose language), Java вимагає написання великої кількості коду, що робить появу помилок більш ймовірною;

— Java повільніша в порівняння з компільованими мовами.

Що є в Java, чого немає в Kotlin:

— статичні члени;

— примітивні типи, які не є класами;

— приватні поля;

— Wildcard(шаблоні) типи;

— зазначені винятки.

Переваги Kotlin:

— бурхливо розвивається як мова під Android розробку, крім того, використовується на backend, наприклад в Spring 5;

— легкий перехід з Java на Kotlin — достатньо встановити плагін Kotlin, додати його в Gradle bild файли й конвертувати;

— наявність функцій розширення, які допомагають розробляти чисті API;

— наявність null в системі типів, проблема з null значеннями дуже поширена в Java;

— Kotlin лаконічна мова, що зменшує кількість потенційних помилок;

— сумісність з Java — модулі на Kotlin будуть працювати разом з наявним Java кодом, а також з усіма Java бібліотеками та фреймворками.

Недоліки Kotlin:

— досить крута крива навчання при перемиканні цілих команд на Kotlin через лаконічний синтаксис мови;

— менша швидкість компіляції в порівнянні з Java;

— значно менше ком’юніті розробників, завдяки чому може бути складно швидко отримати відповідь на питання на Stackoverflow;

— автодоповнення Android Studio і компіляція проєкту працює трохи повільніше в порівнянні з чисто Java проєктами.

Що є в Kotlin, чого немає в Java:

— шаблони рядків;

— одинак(Singleton);

— Null безпека;

— функції розширення;

— розумні приведення типів (smart casts) [9].

Отже, зважаючи на результати порівняння існують ключові аргументи, що впливають на рішення обрати Java як мову програмування для розробки застосунку, а саме: за допомогою віртуальної машини Java програми цією мовою можуть запускатися практично в будь-якій системі. Java прекрасно підходить для розробки крос-платформених додатків.

1.5 Вибір системи керування базою даних

Для реалізації програмного засобу батьківського контролю важливо також обрати систему керування базою даних (СКБД), оскільки головна логіка роботи проєкту безпосередньо пов’язана з використанням баз даних. Для обґрунтованого вибору потрібної СКБД необхідно провести огляд різних їх типів.

SQLite — це бібліотека на мові C, яка реалізує невеликий, швидкий, автономний, високонадійний, повнофункціональний механізм баз даних SQL. SQLite — це найбільш використовуваний механізм баз даних у світі. SQLite вбудований у всі мобільні телефони та більшість комп’ютерів і входить до складу безлічі інших програм, якими люди користуються щодня.

Формат файлу SQLite стабільний, крос-платформний і зворотно сумісний, і розробники зобов'язуються зберігати його таким щонайменше до 2050 року. Файли бази даних SQLite зазвичай використовуються як контейнери для передачі об’ємного вмісту між системами та як довгостроковий архівний формат даних. На цей час в активному використанні понад 1 трильйон баз даних SQLite.

База даних Firebase Realtime — це база даних NoSQL, розміщена в хмарі. Дані зберігаються в форматі JSON і синхронізуються в реальному часі з кожним підключеним клієнтом. При створенні крос-платформених додатків за допомогою SDK для iOS, Android і JavaScript, всі клієнти використовують один екземпляр бази даних Realtime й автоматично отримують оновлення з новітніми даними (рисунок 1.5).

Для реалізації застосунку обрано саме базу даних Firebase Realtime, тому що в ній наявний ряд переваг, які якісно впливатимуть на роботу програмного засобу.

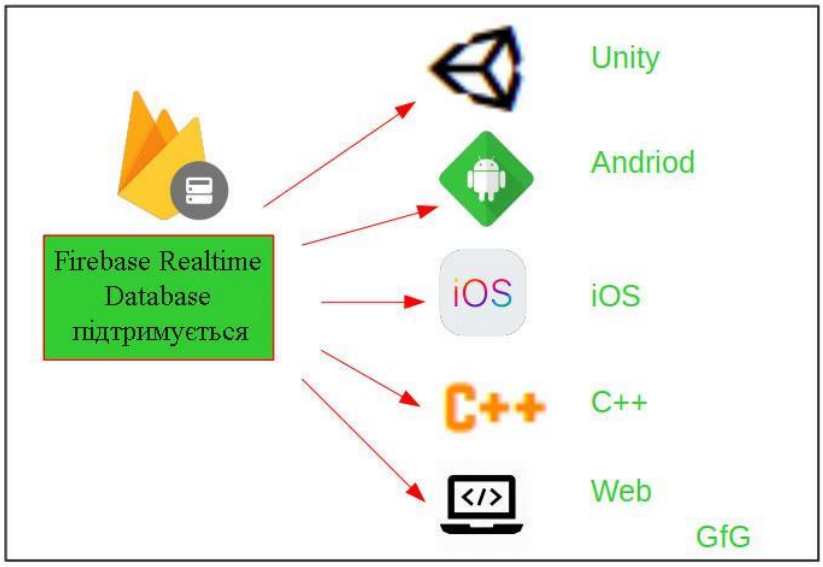


Рисунок 1.5 — Платформи, що підтримують Firebase Realtime Database

Її ключовими можливостями є:

— замість звичайних HTTP-запитів база даних Firebase Realtime використовує синхронізацію даних кожен раз, коли дані змінюються, будь-який підключений пристрій отримує це оновлення протягом мілісекунд;

— додатки Firebase залишаються працездатним навіть в автономному режимі, оскільки SDK Firebase Realtime Database SDK зберігає ваші дані на диску й після відновлення підключення клієнтський пристрій отримує всі пропущені зміни, синхронізуючи їх з поточним станом сервера;

— доступ до бази даних Firebase Realtime можна отримати безпосередньо з мобільного пристрою або веб-браузера, нема потреби в сервері додатків, а безпека і перевірка даних доступні через правила безпеки бази даних Firebase в реальному часі, правила на основі виразів, які виконуються при читанні або запису даних;

— з базою даних Firebase Realtime можна підтримувати потреби застосування в даних в потрібному масштабі, розділяючи дані між кількома екземплярами бази даних в одному проєкті Firebase, також є можливість оптимізувати аутентифікацію за допомогою Firebase Authentication в проєкті й аутентифікацію користувачів у примірниках бази даних, керувати доступом до даних в кожній базі даних за допомогою настроюваних правил бази даних Firebase Realtime для кожного екземпляра бази даних;

— Firebase надає бекенд, простий у використанні SDK і готові бібліотеки призначені для користувацького інтерфейсу, для реалізації аутентифікації користувачів у додатку, він підтримує аутентифікацію як за допомогою email і пароля, так і за допомогою таких популярних постачальників ідентифікації, як Google, Facebook, Twitter і GitHub, а також сервіс аутентифікації тісно інтегрується з іншими сервісами Firebase, використовує галузеві стандарти, такі як OAuth 2.0 і OpenID Connect, так що він може бути легко інтегрований [10].

Отже, проаналізувавши усі аспекти, що стосуються створення застосунків для мобільних пристроїв, було прийнято рішення розробити програмний засіб батьківського контролю на базі операційної системи Android в інтегрованому середовищі Android Studio за допомогою мови програмування Java, зважаючи на всі їхні переваги та недоліки. Також в якості бази даних вирішено використовувати Firebase Realtime Database, зважаючи на переконливі особливості роботи такого типу БД.

2 ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКУ БАТЬКІВСЬКОГО КОНТРОЛЮ

2.1 Основні типи нативних Android-додатків

Перш ніж розпочати роботу над створенням нативних мобільних додатків, доцільно розглянути їхні різновиди. Залежно від того, до якого типу належить той чи інший додаток, стає зрозумілим, які моменти під час розроблення потребують найбільшої уваги. Класифікація мобільних додатків має наступну структуру.

Передньопланові додатки реалізують власні функції тоді, коли їх видно на екрані, у протилежному разі їх виконання буде припинено. До таких додатків можна віднести відеоплеєри, різноманітні редактори тощо. Під час цих програм потрібно ретельно врахувати життєвий цикл активності, щоби перехід між фоновою роботою та роботою переднього плану був плавним. Аби досягти такої плавності, необхідно переконатися, що додаток зберігає свій стан під час переходу у фонову роботу і відновлює його в разі переходу на передній план. Важливо враховувати той момент, що інтерфейс додатку має бути зручним та інтуїтивно зрозумілим.

Фонові додатки працюють у прихованому стані та не вимагають взаємодії з користувачем після їхнього налаштування. Такими програмами є служби перевірки викликів, автовідповідачі та інше. Багато фонових додатків спрямовані на відстеження подій, що генеруються сервісами, системою чи іншими додатками та виконуються у фоновому режимі. Розробник спроможний проєктувати повністю невидимі сервіси, але в такому разі ними не можливо керувати. Мінімальна кількість дій, на які має бути спроможний користувач — це дати змогу запустити службу, налаштувати її, тимчасово зупинити та припинити її роботу, якщо це необхідно.

Переривчасті додатки здебільшого працюють у фоновому режимі, але дають змогу встановлювати зв’язок із користувачем після налаштування. Найчастіше взаємодія з користувачем обмежується сповіщенням про подію. До прикладу такими додатками є медіа плеєри, програми чатів та інше. Характерною рисою таких додатків є те, що вони здатні реагувати на надходження даних від

користувача без втрати продуктивності у режимі фонової роботи. Ці програмні засоби здебільшого мають як передньопланові активності, так і фонові служби та зобов'язані брати до уваги власний поточний стан під час взаємодії з кінцевим користувачем. Для того, щоб обновляти графічний інтерфейс, коли додаток перебуває на передньому плані, або відправити сповіщення користувачу з фону, аби інформувати його, необхідно зважати на ці характеристики під час розроблення таких програм [11].

Зазвичай, комплексні додатки можуть мати елементи будь-якого з описаних типів. Під час планування розроблення програмного засобу важливо визначити, у який спосіб додаток використовуватиметься, а потім перейти до самого проєктування та розроблення.

2.2 Загальна архітектура Android-застосунку

Архітектура Android-застосунку ґрунтується на використанні чотирьох основних компонентів програми (рисунок 2.1) та деяких додаткових. Компоненти додатків дуже важливі для створення програм. Вони працюють як точка входу для користувачів або системи для входу в програму. Кожен компонент має своє призначення та чіткий життєвий цикл.

Рисунок 2.1 — Основні компоненти Android-застосунку

Activity (активність) — це клас, який розглядається як точка входу для користувачів, що представляє єдиний екран. Здебільшого додатки складаються з декількох активностей. Кожна активність не залежить одна від одної. Серед усіх видів активностей одна — це MainActivity (), а решта — її ChildActivities (). Як правило, перша сторінка, яка з'являється на екрані при відкритті програми, називається MainActivity (). Ця основна активність взаємодіє з дочірніми активностями та дозволяє користувачеві отримати до них доступ.

Активності зберігаються у стеці активностей, де поточна активність займає найвищу позицію. На рисунку 2.2 зображено, як кожна нова активність додає елемент у зворотній стек, і як поточна активність знищується, а попередня відновлюється.

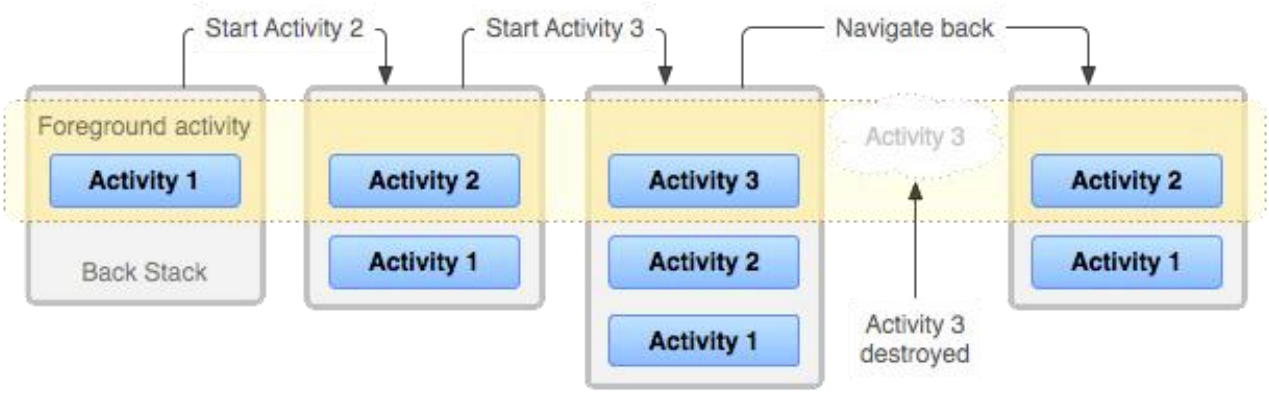


Рисунок 2.2 — Процес роботи стеку

Service (сервіс) — це компонент програми, який працюють у фоновому режимі. Це процес, який не потребує прямої взаємодії з користувачем. Оскільки він виконує тривалі процеси без втручання користувача, він не мають інтерфейсу користувача. Він може бути підключений до інших компонентів і здійснювати міжпроцесорний зв'язок.

Сервіси можуть бути трьох типів.

Сервіси переднього плану — це ті сервіси, які видимі для користувачів. Користувачі можуть вільно взаємодіяти з ними та відстежувати, що відбувається. Ці сервіси продовжують працювати навіть тоді, коли користувачі використовують інші програми. Чудовим прикладом цього є музичний плеєр та завантаження.

Довідкові сервіси працюють у фоновому режимі, так що користувач не може їх бачити або отримувати до них доступ. Вони не потребують контролю користувача. Синхронізація та зберігання даних можуть бути найкращим прикладом.

Пов'язані сервіси працюють до тих пір, поки до них прив'язані якісь інші компоненти програми. Багато компонентів можуть одночасно прив’язуватися до одного сервісу, але як тільки всі вони від’єднаються, сервіс знищиться.

Content provider (контент-провайдер) — це компонент, який дозволяє застосункам обмінюватися даними між кількома програмами. Він приховує деталі баз даних і може використовуватися для читання та запису приватних даних програми, яка не є спільною. На рисунку 2.3 проілюстровано механізм роботи контент провайдера. Прикладом використання контент-провайдера є можливість пошуку контактних даних у списку контактів, або завантаження фотографії з галереї в додаток.

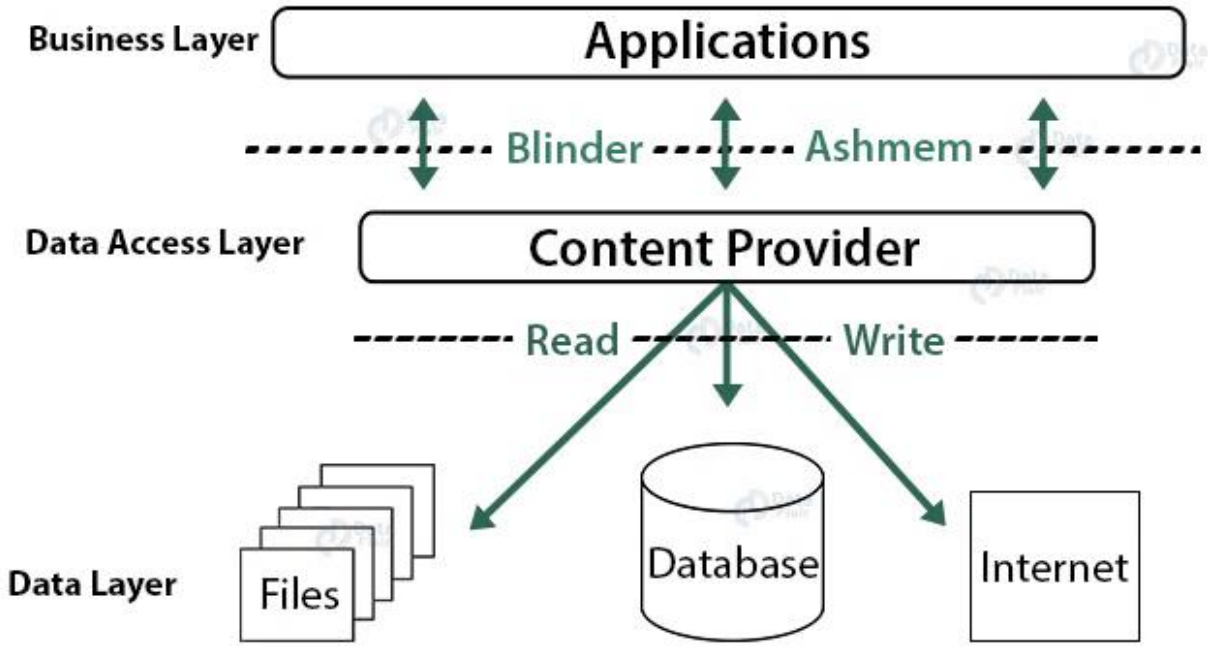


Рисунок 2.3 — Механізм функціонування контент-провайдера

Broadcast Receiver (приймач широкомовних повідомлень) — це компонент Android, який використовується для трансляції повідомлень до системи або інших програм. Повідомлення, що транслюється, називається подією чи наміром.

Трансляційні приймачі, на відміну від активностей, не мають інтерфейсу користувача. Він використовується для асинхронного міжпроцесорного зв'язку. Прикладом приймача широкомовних повідомлень може слугувати повідомлення користувача, що заряд акумулятора низький [12].

Додатковими компонентами програми являються наступні пункти.

Intent (намір) — це фреймворк для передачі повідомлень між додатками для зв'язку між компонентами Android. Він також використовується для передачі даних між різними видами активностей, а також для запуску нового сервісу та відображення списку в ListView. До прикладу, програма камери надсилає намір операційній системі, коли користувач вирішує поділитися зображенням.

Widget (віджет) — це різновид широкомовних приймачів та основний аспект налаштування головного екрану. Він відображає дані і дозволяєь користувачам виконувати дії над ним. Існують різні типи віджетів.

Інформаційний віджет відображає важливу інформацію та відстежує, як інформація змінюється з часом. Наприклад, віджети годинника та віджети, які відображають інформацію про погоду та час.

Віджет колекції — це колекція інформації того самого типу. Його використання призначене для перегляду інформації та відкриття будь-якого з елементів для перегляду деталей. Наприклад, музичні віджети, оскільки ми можемо пропускати паузи та відтворювати музику поза музичним додатком.

Віджет контролю відображає функціональні можливості, і використовуючи його, користувач може керувати програмою з головного екрану, не відкриваючи її, наприклад, призупинити та відтворити відео за межами програми.

Гібридний віджет поєднує функції всіх інших трьох віджетів. Віджет музичного програвача — це віджет управління, але він також повідомляє користувачеві про те, яка доріжка відтворюється в даний час, а це означає, що це поєднання контролю та інформації, тому він називається гібридним віджетом.

View (вид) відповідає за зовнішній вигляд елементів та обробку подій при взаємодії з ними. До таких елементів належать кнопки, поля для вводу, картинки, вибори з одним або декілька можливими значеннями та інше (рис. 2.4).

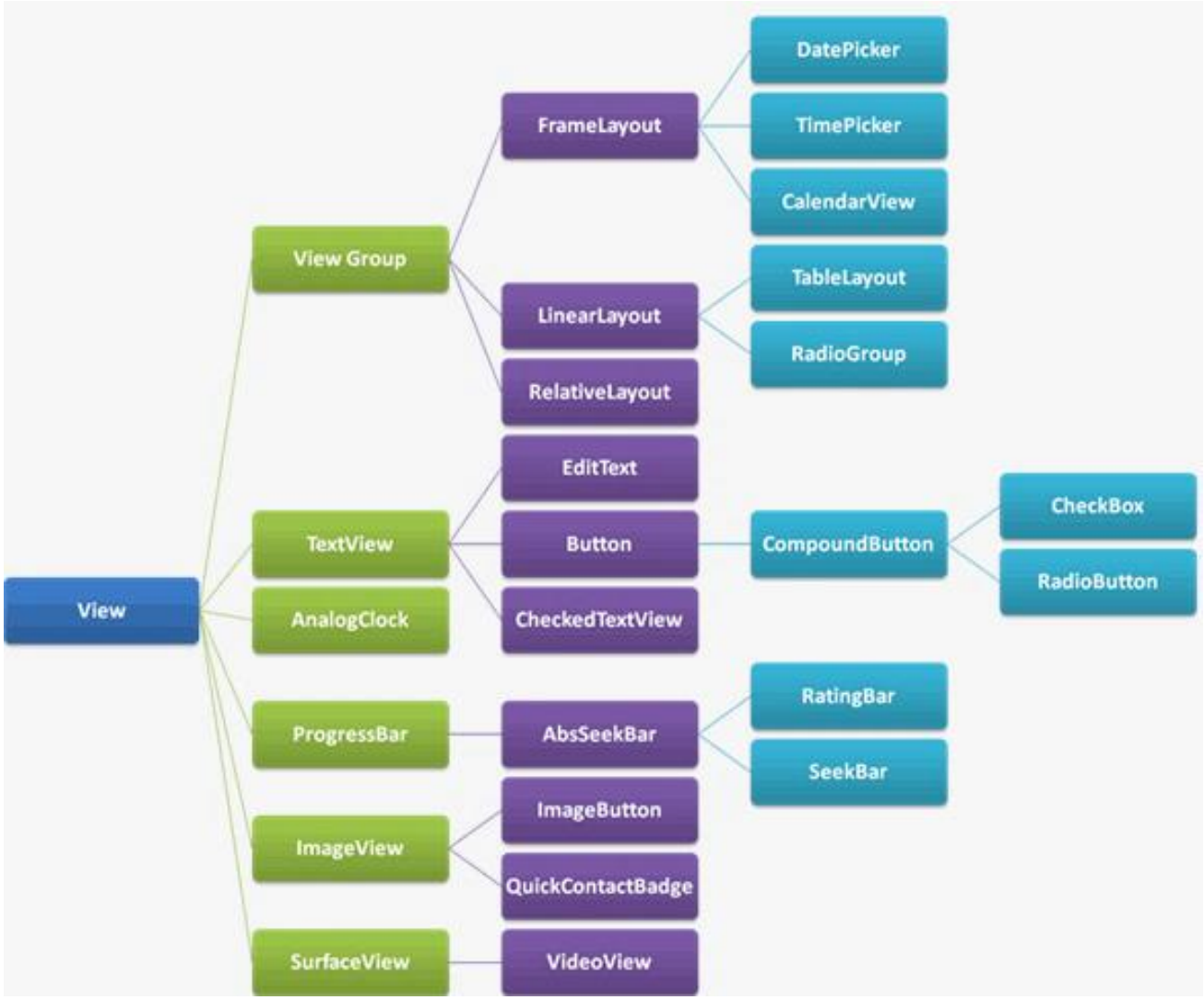


Рисунок 2.4 — Ієрархія компонента елементів

Notification (сповіщення) інформує користувачів, коли додаток не видно або він неактивний. Сповіщення з’являється на екрані, а потім зникає. Може супроводжуватися звуковими сигналами, а також активує індикатор сповіщень.

Fragment (фрагмент) — це частина загального інтерфейсу користувача. Користувачі можуть об'єднати більше одного фрагмента в одній активності, і ці фрагменти можуть бути використані повторно в декількох активностях. Фрагмент, як правило, містить Views і ViewGroups всередині них.

Layout XML files (розмітка XML-файлів) — це структура для інтерфейсу користувача в додатку. XML-файли забезпечують різні типи макетів для різних типів екранів, а також визначає, який компонент графічного інтерфейсу користувача містить активність чи фрагмент.

App APK files (APK-файли додатків) — це формат файлу пакету, який містить код програми, ресурси та інше. Операційна система Android використовує їх для встановлення мобільних додатків та проміжного програмного забезпечення.

Resources (ресурси) призначені для визначення зображень, текстів, рядкових значень. Все, що визначено у файлі ресурсів є доступним до посилань на них у вихідному коді.

2.3 Встановлення Android Studio та створення проєкту

Найпершим кроком для створення власного застосунку було завантаження та налаштування Android Studio й Java Development Kit, використовуючи офіційні посилання: http://developer.android.com/sdk/index.html та https://www.oracle.com/technetwork/java/javase/downloads/index.html відповідно. Використала програму встановлення для інсталяції Android Studio дотримуючись її інструкцій.

Наступним кроком є безпосередня взаємодія з Android Studio. При запуску на екрані з’являється меню, що зображено на рисунку 2.5. Для створення нового проєкту слід вибрати «Start a new Android Studio project». Наступним кроком було запропоновано обрати тип Activity в залежності від апаратної бази, для якої створюється додаток, та від виду додатку. У цьому вікні у вкладці «Phone and Tablet» обираємо «Empty Activity».

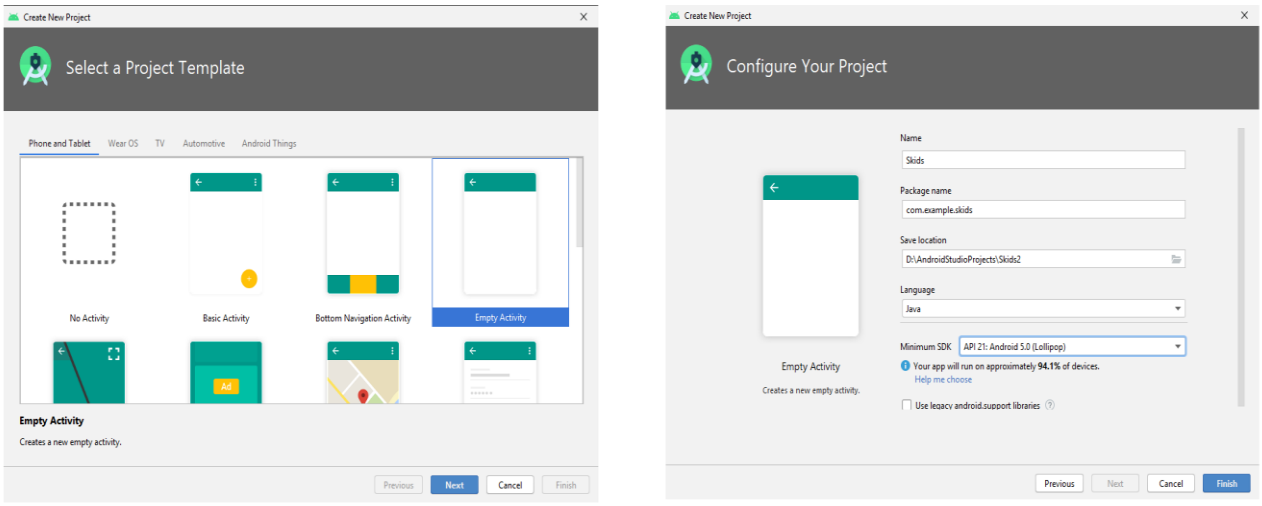


Рисунок 2.5 — Початкові налаштування нового проєкту в Android Studio

Обравши вид Actitvity з’являється діалогове вікно майстра створення проєкту, який містить наступні поля.

Name — назва програми, яка відображатиметься в заголовку. У стартових налаштуваннях назва пишеться латиницею, а локалізовані назви готують згодом. Цьому полю присвоєно назву «Skids» (Kids safety), що в перекладі з англійської означає «безпека дітей».

Package name — повна назва проєкту (згідно з правилами іменування пакетів в Java). Ім'я пакета повинно бути унікальним серед всіх пакетів, встановлених в системі Android. Встановити значення необхідно з урахуванням імені додатку та домену. Домен дозволяється вказувати довільний, не обов'язково реально існуючий. Package name для розроблюваної програми є наступним: com.example.skids.

Save location — це каталог системи розробника, що вміщує файли проєкту (D:\AndroidStudioProjects\Skids).

Language — це мова програмування за вибором (Java, Kotlin). У нашому випадку додаток створюється за допомогою мови програмування Java.

Minimum API level — це найраніша версія Android, яку підтримує розроблювана програма. Щоб охопити максимальну кількість підтримуваних пристроїв, можна встановити параметр Minimum API level в найменшу доступну версію, але при цьому втрачається можливість користуватися новими можливостями розробки, які доступні у більш високих API рівнях. Під час вибору API рівня використовують інформацію з офіційного сайту для Android-розробників https://developer.android.com, а саме розділ, у якому наведено дані про відносну кількість пристроїв, на яких запущено певну версію платформи Android. На сьогоднішній день розподіл версій Android та API рівнів мають наступний вигляд, продемонстрований на рисунку 2.6.

Для проєкту Minimum API level обрано API 21: Android 5.0 (Lolipop), оскільки саме використання цієї версії забезпечує можливість запуску додатку на 94.1% пристроїв [13]. До того ж дана версія забезпечує більш менше комфортний набір можливостей для розробки.

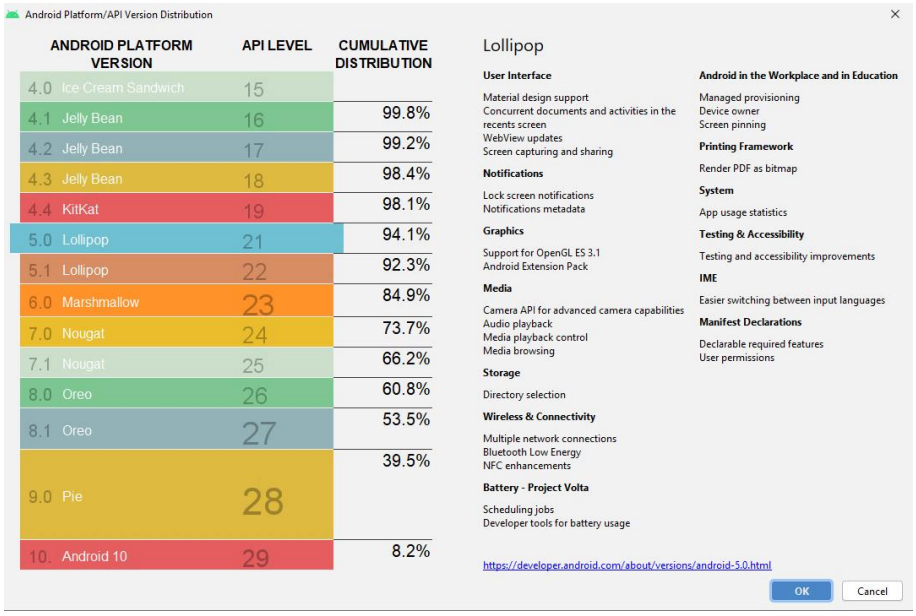


Рисунок 2.6 — Відсоткова частка користувачів різними версіями Android

Заповнивши всі поля, які представлені «майстром» за допомогою кнопки «Finish» можна створити проєкт для подальшої над ним роботи. У результаті маємо дві головних теки: app, що включає усі потрібні файли додатку та Grandle Scripts, який необхідний для різноманітних налаштувань, керування проєктом тощо. Тека app містить три каталоги:

— manifest, що вміщує один файл AndroidManifest.xml, у якому оголошуються активності, сервіси, приймачі широкомовних повідомлень та контент-провайдери застосунку, також вміщає в собі потрібні дозволи;

— java, що складається із трьох тек, з однієї робочої з файлами класів, та двох інших створених для тестів;

— res — це файли з ресурсами, які розташовані в осібних теках, на кшталт: drawable (графічні ресурси, що відповідають різним розмірам екрану), layout (xml-файли, що описують зовнішній вигляд форм, та їхніх складових; у разі запуску проєкту вже наявний файл activity_main.xml, що відповідає за вигляд головної активності застосунку), menu (ресурси меню; з самого початку доступний файл menu_main.xml і він відповідає за меню головної активності застосунку), mipmap (різноманітні іконки застосунку й за своєю структурою збігається з drawable) та values (стрічкові ресурси та ресурси тем, кольорів, стилів та інше, що використовуються в проєкті).

2.4 Реалізація головної активності застосунку

Розроблюваний проєкт розпочинає свою роботу із виведенням головної активності на екран пристрою. Ця активність має власний інтерфейс, створений за допомогою мови розмітки XML, та має власну логіку роботи, яка допомагає забезпечити комфортну, а головне, продуктивну взаємодію застосунку та користувача. Інтерфейс головного вікна додатку використовує відносний макет (RelativeLayout). Контейнер RelativeLayout представляє об’єкт ViewGroup (група елементів), який впорядковує всі дочірні елементи один відносно одного. Елементами головної активності є: дві кнопки та три текстові поля. Приклад реалізації однієї із кнопок головного вікна, текстового поля та загальна структура розмітки наведена нижче в лістингу програми:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent" <!--ширина елемента-->
    android:layout_height="match_parent" <!--висота елемента-->
    android:background="@drawable/background_new" <!--картинка-фон-->
    tools:context=".MainActivity">
<ImageView
        android:id="@+id/child_logo"
        android:src="@drawable/child_logo"/>
  <ImageButton
    android:id="@+id/button_child" <!--ідентифікатор кнопки-->
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/who_are_you" <!--відносна позиція кнопки-->
    android:layout_alignParentStart="true"
    android:layout_marginStart="25dp" <!--відступ зліва  -->
    android:layout_marginTop="50dp" <!--відступ зверху  -->
```

```
android:background="@color/fui_transparent" <!--прозорість фону  -->
android:contentDescription="@string/btnChild" <!--опис кнопки  -->
android:src="@drawable/kids_btn" /> <!--ресурс картинки  -->
<TextView
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:layout_below="@+id/button_child"<!--відносна позиція кнопки-->
 android:layout_alignParentStart="true"
 android:layout_marginStart="40dp" <!--відступ зліва  -->
 android:layout_marginTop="20dp" <!--відступ зверху  -->
 android:text="@string/child" <!--текстовий ресурс -->
 android:textColor="#fefefe" <!--колір тексту -->
 android:textSize="25sp" />
</RelativeLayout>
```

Увесь лістинг інтерфейсу головної активності прописаний у файлі activity_main.xml та наведений у додатку Б.

Важливою перевагою додатку над його аналогами є локалізація. Якщо користувач запустить додаток на телефоні з англійською локалізацією, то він побачить текст на знайомій йому мові. Якщо додаток запустить український користувач, то він побачить текст українською мовою відповідно. Програма буде підтримувати англійську та українську локалізації. Якщо на телефоні не буде знайдений потрібний локалізований ресурс, то буде застосовуватися англійський варіант за замовчуванням.

За допомогою майстра студії додано новий підкаталог values-uk в каталозі res. Клацнувши правою кнопкою миші на теці res і вибравши New | Android resource directory, у діалоговому вікні в лівій частині Available qualifiers: вибрано пункт Locale і перенесено його в праву частину Chosen qualifiers: за допомогою кнопки з двома стрілками вправо. У третій колонці вибираємо українську мову. В полі Directory name автоматично з'явилася потрібну назву теки. Додатково можна вказати й регіон в колонці Specific Region Only (рисунок 2.7).

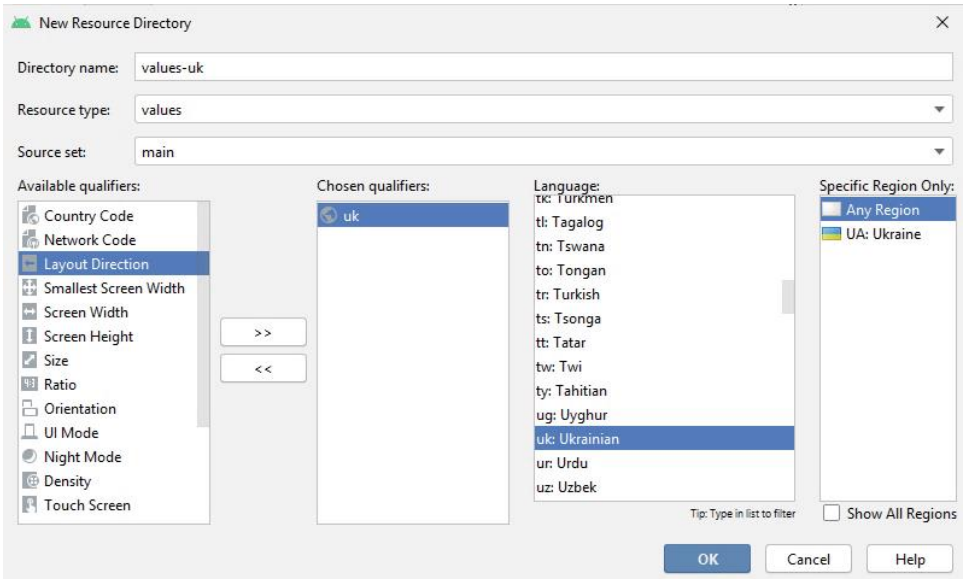
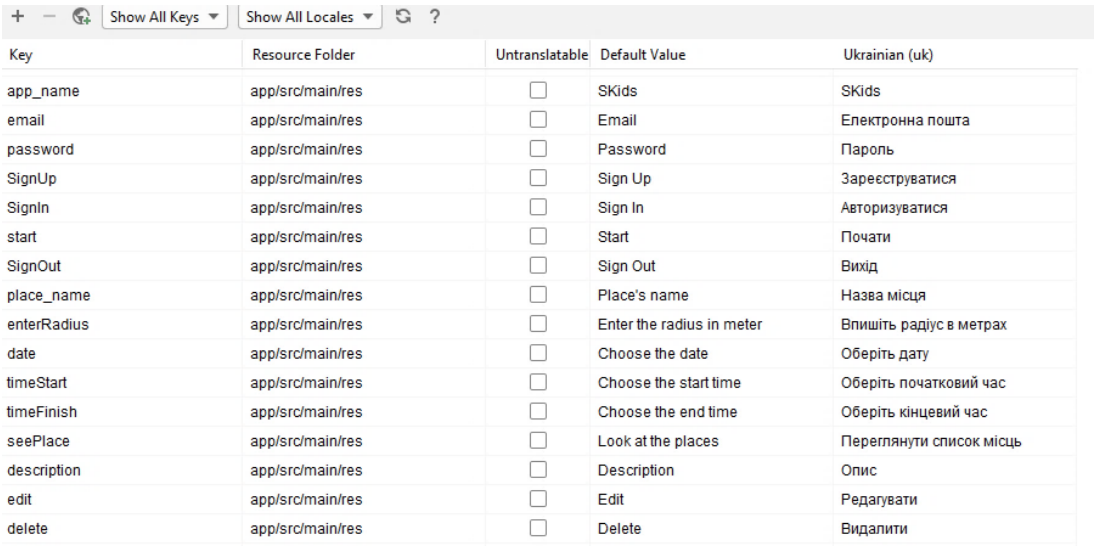


Рисунок 2.7 — Додавання української локалізації засобами майстра студії

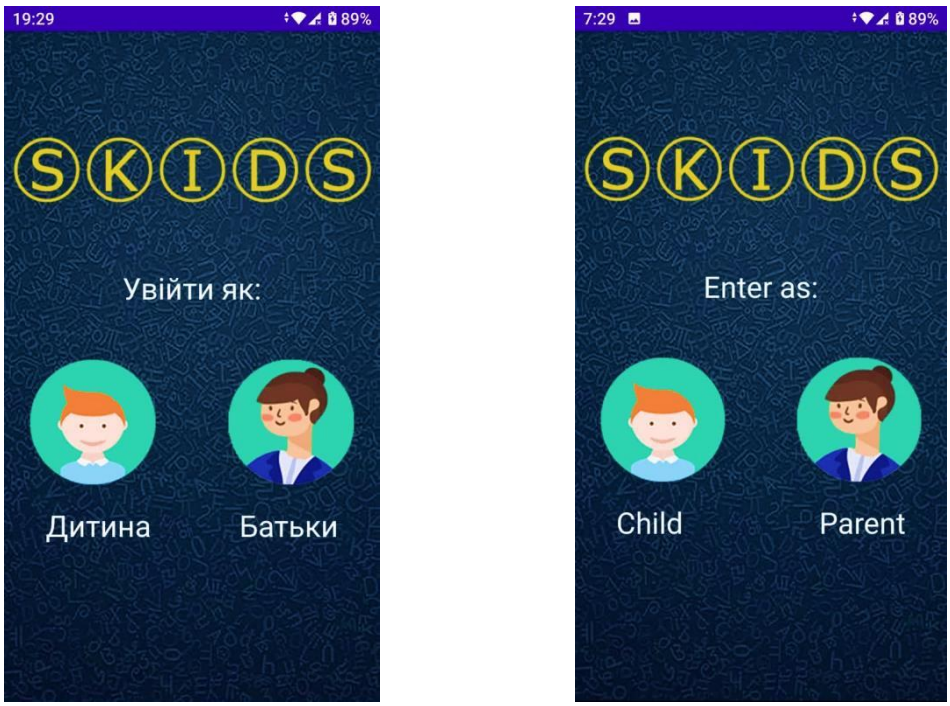
Для того, щоб побачити створений підкаталог, необхідно з режиму Android перейти у режим Project. Наступним кроком скопіювати файл з англійською локалізацією strings.xml та за допомогою Translation Editor (редактор перекладів) створити файл strings.xml(uk)(рисунок 2.8). У додатку Б наведений лістинг обох вище зазначених файлів.

У результаті запуску проєкту на екрані відображається наступний макет вікна, проілюстрований на рисунку 2.9. Але для того, щоб кнопки на екрані реагували на дії користувача, необхідно прописати обробники на їх натискання. У поточному підрозділі розглянемо обробку кнопок.



Key	Resource Folder	Untranslatable	Default Value	Ukrainian (uk)
app_name	app/src/main/res	☐	SKids	SKids
email	app/src/main/res	☐	Email	Електронна пошта
password	app/src/main/res	☐	Password	Пароль
SignUp	app/src/main/res	☐	Sign Up	Зареєструватися
SignIn	app/src/main/res	☐	Sign In	Авторизуватися
start	app/src/main/res	☐	Start	Почати
SignOut	app/src/main/res	☐	Sign Out	Вихід
place_name	app/src/main/res	☐	Place's name	Назва місця
enterRadius	app/src/main/res	☐	Enter the radius in meter	Впишіть радіус в метрах
date	app/src/main/res	☐	Choose the date	Оберіть дату
timeStart	app/src/main/res	☐	Choose the start time	Оберіть початковий час
timeFinish	app/src/main/res	☐	Choose the end time	Оберіть кінцевий час
seePlace	app/src/main/res	☐	Look at the places	Переглянути список місць
description	app/src/main/res	☐	Description	Опис
edit	app/src/main/res	☐	Edit	Редагувати
delete	app/src/main/res	☐	Delete	Видалити

Рисунок 2.8 — Редактор перекладів



а) локалізований б) міжнародний

Рисунок 2.9 — Інтерфейси головної активності

Реалізація цих елементів знаходиться у файлі MainActivity.java на початку якого йдуть визначення пакету та імпорт зовнішніх пакетів. Опис об'єктів винесено за межі методу onCreate(). Це зроблено для того, щоб мати змогу з будь-якого методу звертатися до них.

В методі onCreate() викликається метод addListenerOnButton(), в якому об'єкти прив’язано до відповідних графічних елементів екрану за допомогою методу findViewById() та прописані обробники на натискання на кнопку (їх також називають слухачами — listener), які присвоюються за допомогою методу setOnClickListener. Коли на кнопку натискають, обробник реагує і виконує код з методу onClick:

```
public class MainActivity extends AppCompatActivity {
    ImageButton btnChild, btnParent;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        addListenerOnButton();}
    public void addListenerOnButton() {
        btnChild = (ImageButton)findViewById(R.id.button_child);
        btnParent = (ImageButton)findViewById(R.id.button_parent);
        btnChild.setOnClickListener(
                new View.OnClickListener() {
                  @Override
                   public void onClick(View v){
                      Intent intentChild = new Intent(".ActivityChild");
                      startActivity(intentChild);}});
        btnParent.setOnClickListener(
                new View.OnClickListener() {
                   @Override
                   public void onClick(View v) {
                      Intent intentParent = new Intent(".ActivityParent");
                      startActivity(intentParent);}});
}
```

При натисканні на кожну із цих кнопок створюється та ініціалізується об’єкт Intent. У всіх випадках Intent є явним та виконує роль перемикача між різними активностями (ActivityParent, ActivityChild). Реалізація усіх частин застосунку мовою Java наведена у додатку В.

2.5 Реалізація активності авторизації батьків

Для цієї активності характерним є те, що вона реалізує два види користувацького інтерфейсу. Макет такої активності вміщає усі елементи цих двох екранів, але не всі вони показуються одночасно. Першим є екран із двома текстовими полями для вводу («Електронна пошта» та «Пароль») та двома кнопками («Зареєструватися» та «Авторизуватися») (рисунок 2.10). Опис кнопок у файлі розмітки є схожим до попередньої активності описаної у попередньому підрозділі.

Рисунок 2.10 — Інтерфейс активності реєстрації та авторизації

Текстові поля для вводу формуються подібно, але мають власні особливості такі як: тип введених даних та показ підказок для користувача:

```
<EditText
    android:id="@+id/etEmail"
    android:hint="@string/email"
    android:inputType="textEmailAddress""/>
 <EditText
    android:id="@+id/etPassword"
    android:hint="@string/password"
    android:inputType="textPassword"/>
```

Для реалізації реєстрації та авторизації користувача використано можливості бази даних (БД) реального часу Firebase. Для того, щоб ними користуватися необхідно спочатку ппиєднати дану БД до проєкту. На офіційному сайті Firebase клацнувши на іконку «Додати проєкт» розпочинається поетапне налаштування за допомогою майстра:

— вводиться ім’я проєкту;

— за бажанням додається Google-аналітика;

— ініціалізується проєкт;

— вводиться назва пакету Android;

— завантажується файл google-services.json;

— в Android studio вставити файл в теку Project->My Application->app;

доповнюється файл build.gradle рівня проєкту та застосунку відповідними залежностями та плагінами.

Після початкових налаштувань в консолі Firebase доступна можливість увімкнення одного із запропонованих способів аутентифікації користувача (рисунок 2.11).

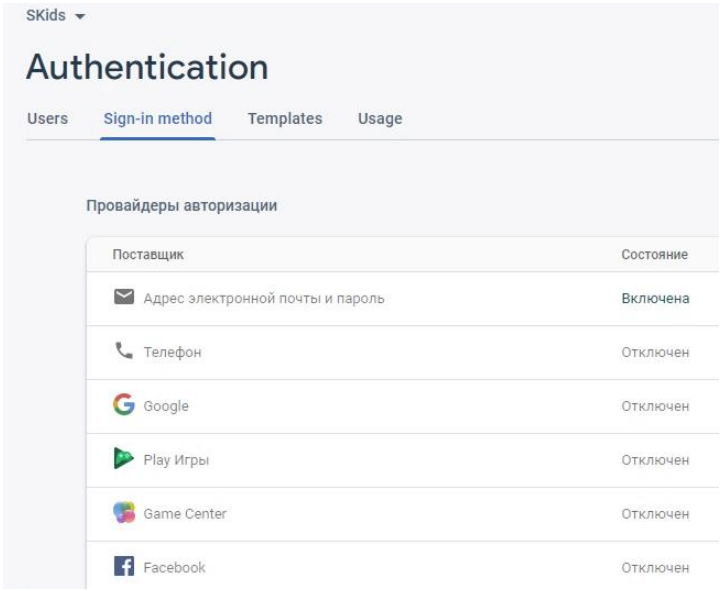


Рисунок 2.11 — Способи аутентифікації користувача

У розроблюваному проєкту використано спосіб аутентифікації за допомогою електронної пошти та паролю. Реалізувати цю функцію досить легко за допомогою асистента Firebase, відкрити якого можна, обравши розділ «Tools->Firebase» в меню Android studio.

Після завершення реєстрації та авторизації користувача вигляд користувацького інтерфейсу дещо змінюється (рисунок 2.12). Екран має дві кнопки («Почати» та «Вихід») та текстове поле, що відображає електронну пошту користувача.

Рисунок 2.12 — Інтерфейс активності початку роботи та виходу з акаунта

Програмна реалізація активності авторизації батьків має ряд методів, які виконують відповідні перевірки та певні дії:

onClickSignUp () — обробник кнопки «Зареєструватися». Якщо користувач не зареєстрований, то даний метод викликає showSigned() та sendEmailVer() методи. Якщо ж користувач вже зареєстрований то викликається метод notSigned() та висвітлюється попередження, що користувач вже зареєстрований.

onClickSignIn() — обробник кнопки «Авторизуватися». Метод перевіряє чи ел. пошта та пароль належать користувачу зі списку зареєстрованих та таких, що пройшли верифікацію. Якщо належить, то викликається метод showSigned(), у протилежному разі notSigned().

onClickSignOut() — обробник кнопки «Вихід». Метод реалізує вихід з акаунта та викликає метод notSigned().

onClickStart() — перехід на активність з фрагментом мапи та функцією додавання записів у БД.

sendEmailVer() — метод реалізації надсилання листа на ел. пошту користувача з метою його верифікації.

showSigned() — метод, що надає видимості графічним елементам інтерфейсу зображеного на рисунку 2.12.

notSigned() — метод, що надає видимості графічним елементам інтерфейсу зображеного на рисунку 2.10.

2.6 Реалізація активності з фрагментом мапи

Інтерфейс активності має вертикальне лінійне макетування екрану й складається лише з двох елементів: фрагменту Google мапи та кнопки для переходу на нову активність:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   tools:context=".ActivityParentMap">
   <fragment
      android:id="@+id/map"
      android:name="com.google.android.gms.maps.SupportMapFragment"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="10"/>
   <Button
      android:id="@+id/btnSeePlace"
      android:hint="@string/seePlace"
      android:background="@drawable/button_see_place"
      android:onClick="onClickSeePlace"/>
</LinearLayout>
```

Слід зауважити, що для використання у власному додатку сервісу GoogleMap необхідно отримати унікальний API ключ .Для того щоб його отримати необхідно ознайомитися з інструкцією, що висвітлена на офіційному сайті Google Cloud Platform [14]. Дотримуючись інструкцій ви перейдете за посиланням, у якому відразу надається допомога майстра для отримання API ключа (рисунок 2.13).

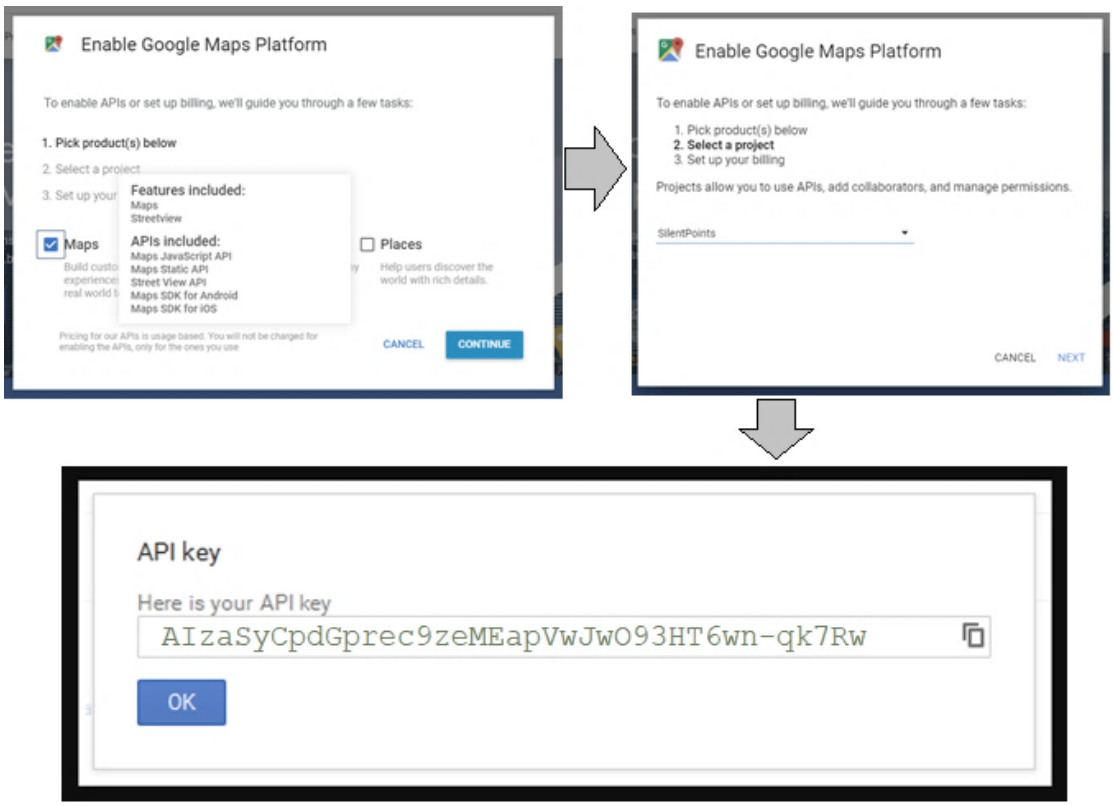


Рисунок 2.13 — Отримання API ключа за допомогою майстра

Для того, щоб мати змогу користуватися мапою, необхідно прописати цей ключ у відповідному місці в маніфесті:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.skids">
<application
<meta-data
        android:name="com.google.android.geo.API_KEY"
        android:value="AIzaSyDoMjz8zVuh3qKZ2ja9wJut_OZtWwuApzQ"/>
</application>
</manifest>
```

Відкривши активність екран матиме вигляд, який проілюстровано рисунком 2.14. Зеленим маркером з отвором всередині на мапі відмічається поточне місцезнаходження користувача-дорослого, а червоним маркером з аналогічним дизайном відмічається місце перебування дитини, у випадку, коли порушено правила перебування [15].

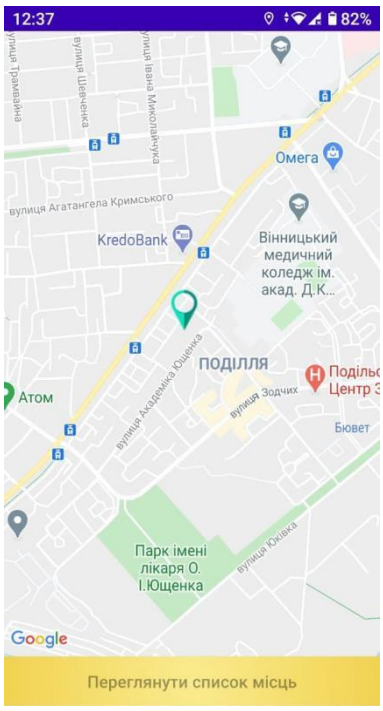


Рисунок 2.14 — Інтерфейс активності з фрагментом мапи

Головними задачами даної активності є визначення місцезнаходження дорослого, запис інформації до БД Place, а також читання з БД Notification та відображення місцезнаходження дитини за потребою, що базується на результатах роботи сервісу. Реалізація задач знаходиться у файлі ExampleJobService.java:

```
public boolean onStartJob(JobParameters params) {
        notificationManager = NotificationManagerCompat.from(this);
        doBackgroundWork(params);
        return true;
    }
```

Лістинг файлу ActivityParentMap.java прописаний у додатку В.

Для отримання інформації з подальшим її записом до БД необхідно натиснути на бажане місце на мапі, в результаті чого — з’являється діалогове вікно (рисунок 2.15), у якому можна вписати місце, радіус, обрати дату, початковий та кінцевий час, у проміжку якого дитина не повинна перетинати межі зазначеної території.

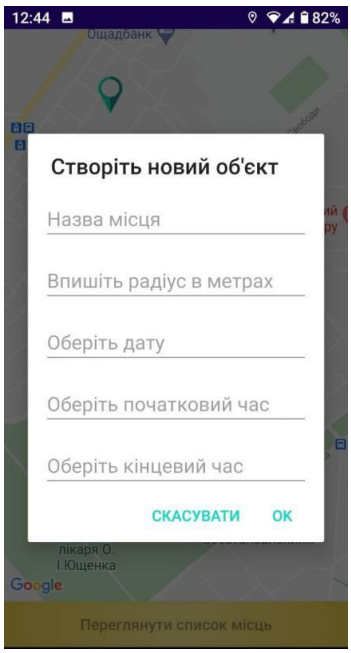


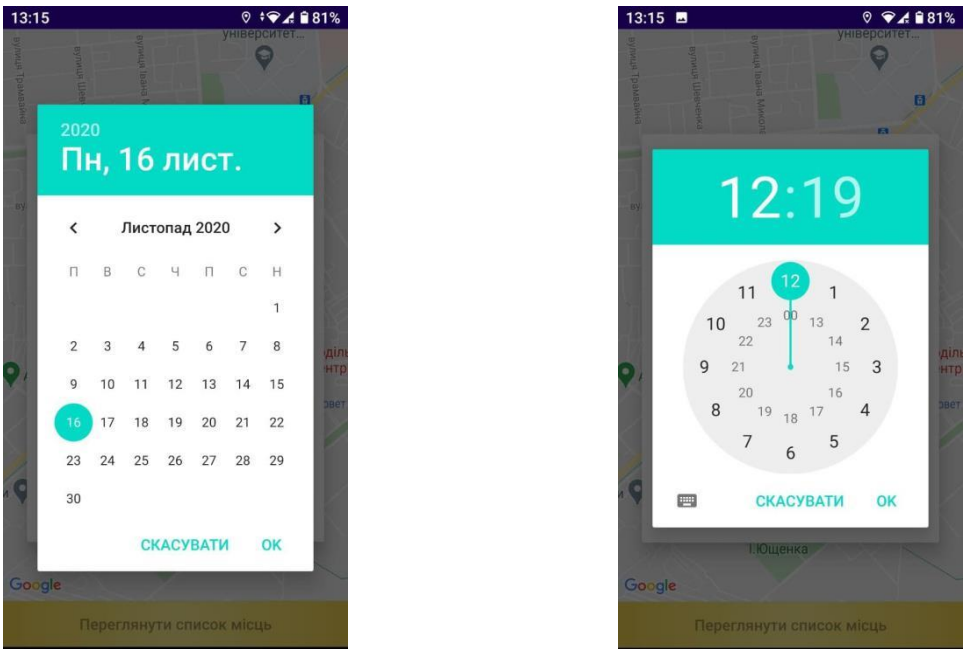
Рисунок 2.15 — Шаблон діалогового вікна

Розмітка даного шаблону діалогового вікна знаходиться в окремому файлі layout_dialog.xml та має лінійне вертикальне макетування, складається з п’яти текстових полів для вводу з підказками, які мають різні типи. Так от поле «Місце» має текстовий тип, тобто при натисненні користувач бачить стандартну розкладку клавіатури. Поле «Радіус» має числовий тип, що в результаті видає користувачу лише клавіатуру з цифрами.

Результатом натискання на поле «Дата» є відображення календаря (рисунок 2.16(а)), реалізація якого здійснення через виклик об’єкта класу DatePickerDialog:

```
myCalendar.set(Calendar.YEAR, year);
myCalendar.set(Calendar.MONTH, monthOfYear);
myCalendar.set(Calendar.DAY_OF_MONTH, dayOfMonth);
```

Відповідно функціонування поля «Початковий час» та «Кінцевий час» реалізовується через клас TimePickerDialog (рисунок 2.16(б)) [16]. Також діалогове вікно має заголовок та дві кнопки «ОК» та «Скасувати». При натисненні на першу інформація додається в БД, при натисненні на іншу діалогове вікно закривається і попередній екран знову стає активним.



а) дати б) часу

Рисунок 2.16 — Інтерфейси вибору

Починаючи з реалізації цієї активності необхідна взаємодія із базою даних. Як зазначалося раніше, для цих цілей використовується БД реального часу Firebase. Для її під’єднання до проєкту достатньо аналогічно до схеми під’єднання функції авторизації слідувати вказівкам асистента Firebase. Для перегляду збереженої інформації достатньо скористатися консоллю Firebase (додаток Г).

Але перш за все потрібно прописати правила доступу до бази даних. Так як в загальному випадку застосунок використовує дві бази даних, то і правила для них є різними. Для БД з місцями авторизованим користувачам дозволено записувати інформацію і читати, а не авторизованим лише читати. Що ж стосується БД з прапорцем сповіщення, то тут дозволено запис й читання не авторизованим користувачам (додаток Д) [17].

Головною ж метою проєкту є повідомлення батьків про порушення правил перебування дитиною. Ця функція реалізовується за допомогою системи умов та перевірок. Найперше здійснюється перевірка чи є у базі даних запис із поточною датою:

```
private void getData(){
    Query dateQuery = mDataBase.orderByChild(DATE).equalTo(currentDate);
```

```
dateQuery.addListenerForSingleValueEvent(new ValueEventListener() {
   @Override
   public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
     if(dataSnapshot.exists())
```

Наступним кроком є виклик сервісу ExampleJobService, який із заданим інтервалом буде перевіряти запис в БД Notification та за потреби надсилати повідомлення на телефон батьків:

```
public void sendOnChannel1() {
 Notification notification =
new NotificationCompat.Builder(this,CHANNEL_1_ID)
        .setSmallIcon(R.drawable.ic_one)
        .setContentTitle(getString(R.string.attention))
        .setContentText(getString(R.string.child_is_out))
        .setPriority(NotificationCompat.PRIORITY_HIGH)
        .setCategory(NotificationCompat.CATEGORY_ALARM)
        .setContentIntent(PendingIntent.getActivity(ExampleJobService.this,0,
             new Intent(ExampleJobService.this,ActivityParentMap.class),
             PendingIntent.FLAG_UPDATE_CURRENT))
        .build();
notificationManager.notify(1,notification);}
```

Також з даної активності здійснюється перехід на нову активність зі списком збережених місць за допомогою кнопки, розміщеної внизу екрану.

2.7 Реалізація активності зі списком збережених місць

Для реалізації макету цієї активності задіяно два xml-файли. Один з яких вміщу елемент RecyclerView та слугує контейнером для елементів карток:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   tools:context=".MainActivity">
```

```
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/list"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@drawable/background_new"/>
</LinearLayout>
```

В свою чергу елементи-картки мають власний макет, який складається з різноманітних поєднань відносних та лінійних, вертикальних та горизонтальних макетів, структура якого зображена на рисунку 2.17.

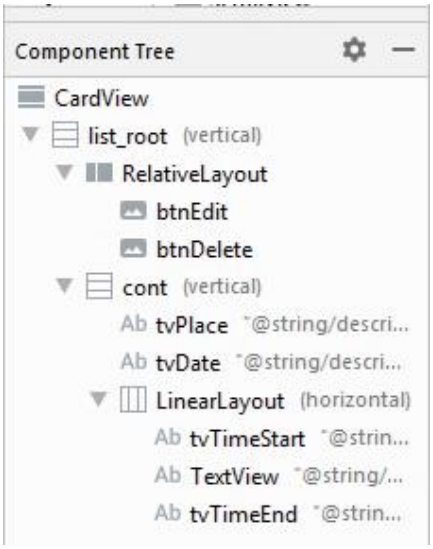


Рисунок 2.17 — Структура макету show_layout.xml

Програмна реалізація функцій цієї активності базується на фільтруванні актуальної інформації з БД та показ її кінцевому користувачу, а також можливості видаляти записи або ж редагувати їх. Для структуризації даних з БД використано клас FirebaseRecyclerAdapter, що використовує користувацький клас ViewHolder.

Записи із не актуальною датою видаляються автоматично. Також є можливість видалити запис вручну скориставшись кнопкою із знаком «хрестик» (рисунок 2.18). Записи також піддаються редагуванні при натисненні на кнопку із знаком «олівець у маркері», після чого на екрані з’являється діалогове вікно тотожне тому, що зображене на рисунку 2.15, але за однією відмінністю, поля вже заповнені даними [18].

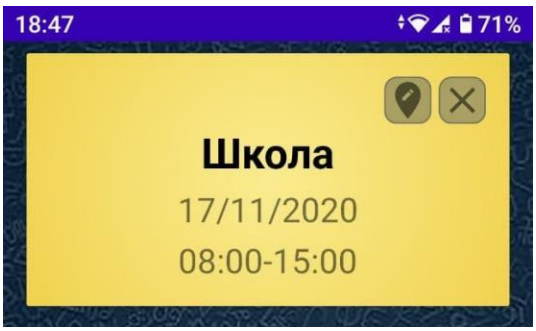


Рисунок 2.18 — Вигляд елемента-картки

2.8 Реалізація активності дитини

Активність дитини є найпростішою за своїм виглядом, тому що складається лише із фрагмента мапи. Що ж стосується програмної реалізації, то вона є дещо розширена і виконує декілька головних завдань. Найперше показує поточне місцеперебування дитини. Рисуються кола з відповідним радіусом, з маркером у центрі, при натисканні на який відображається назва місця та час, що визначає тривалість дії заборони на покидання даної зони (рисунок 2.19) [19]. Слід зауважити, що місця відображаються у відповідності з датою, а отже, у певний день, дитина може бачити лише ті місця, які створені на цей день, з метою зменшити кількість нагромадження інформації, яка не є актуальною за часом.

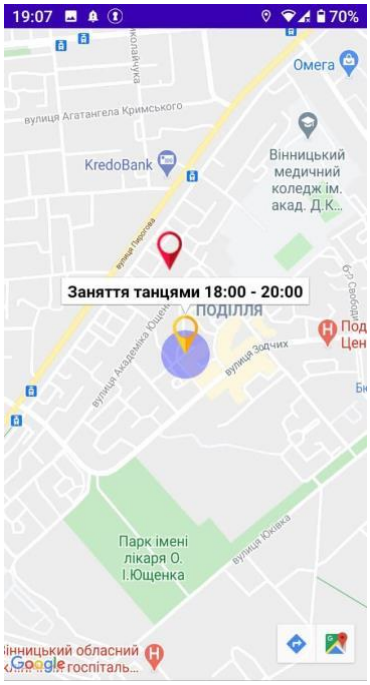


Рисунок 2.19 — Інтерфейс активності дитини

Програмна реалізація рисування кола та інших подій прописана в файлі ActivityChild.java. Читання даних з БД відбувається за допомогою методів класу ValueEventListener:

```
ValueEventListener valueEventListener = new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
           for(DataSnapshot ds : dataSnapshot.getChildren()) {
             placeName = ds.child(PLACE_NAME).getValue(String.class);
             radius = ds.child(RADIUS).getValue().toString();
             latitude = ds.child(LATITUDE).getValue(Double.class);
             longitude = ds.child(LONGITUDE).getValue(Double.class);
             timeStart = ds.child(TIME_START).getValue(String.class);
             timeEnd = ds.child(TIME_END).getValue(String.class);
             date = ds.child(DATE).getValue(String.class);
             drawMap();
             checkDate();}}};
    mDataBase.addValueEventListener(valueEventListener);
```

Користувацькі методи drawMap() та checkDate() відповідають за рисування кіл із нотатками та перевірку поточної дати з зазначеною, щоб розпочати відслідковування місцезнаходження дитини протягом дня відповідно:

```
private void drawMap() {
    Calendar calendar= Calendar.getInstance();
    SimpleDateFormat  dateFormat = new SimpleDateFormat(PATTERN_DATE, Locale.UK);
    String currentDate = dateFormat.format(calendar.getTime());
    if (currentDate.equals(date)) {
       MarkerOptions markerOptions = new MarkerOptions();
       LatLng center = new LatLng(latitude, longitude);
```

```
        markerOptions.position(center).title(placeName + " " + timeStart + " - " +
timeEnd)
                .icon(BitmapDescriptorFactory.fromResource(R.drawable.marker_place));
        mMap.addCircle(new CircleOptions().center(center)
                .radius(Double.parseDouble(radius)).fillColor(0x550000FF));
        mMap.addMarker(markerOptions);}}
private void checkDate() {
    if (currentDate.equals(date)) {
        Intent serviceIntent = new Intent(ActivityChild.this,
ExampleIntentService.class);
        ExampleIntentService.enqueueWork(ActivityChild.this, serviceIntent);}}
```

Розрахунки, які відносяться до перевірки чи знаходиться дитина за межами кола чи в ньому відповідає сервіс ExampleIntentService, який вмикається з певним інтервалом і може працювати навіть тоді, коли застосунок не активний. У даному сервісі зчитуються дані з БД, встановлюється поточне місце знаходження та порівнюються результати:

```
if(timeCur.after(timeS) && timeCur.before(timeE)){
            locationFirebase = new Location("");
            locationFirebase.setLatitude(latitude);
            locationFirebase.setLongitude(longitude);
            double distanceInMeters = currentLocation.distanceTo(locationFirebase);
            if (distanceInMeters>Double.parseDouble(radius))
```

В залежності від результату виконується різні дії. Якщо дитина знаходиться за межами, то створюється БД Notification, куди передається прапорець зі значенням істиності та поточні координати дитини для відображення на батьківському екрані:

```
        notiBase.child(NOTIFICATION_KEY).setValue(true);
        notiBase.child(CURRENT_LATITUDE_KEY).setValue(currentLatitude);
```

notiBase.child(CURRENT_LONGITUDE_KEY).setValue(currentLongitude);

Якщо ж дитина знаходиться у дозволених межах, то прапорець встановлюється значенням хиби, як і в стані за замовчуванням:

notiBase.child(NOTIFICATION_KEY).setValue(false);

Отже, у результаті розробки, було створено застосунок з користувацьким інтерфейсом для кожної активності та втілено правильну логіку їх роботи. Загалом розробка складається з одинадцяти java-класів (ActivityChild, ActivityParent, ActivityParentMap, App, Constants, ExampleIntentService, ExampleJobService, MainActivity, Place, ReadActivity, ViewHolder), семи xml-файлів макетів (activity_child, activity_main, activity_parent, activity_parent_map, layout_dialog, read_layout, show_layout), дев’ятнадцяти файлів ресурсів та додаткових файлів локалізації, кольорів тощо.

3 ТЕСТУВАННЯ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАСОБУ

3.1 Різновиди способів тестування програмного забезпечення

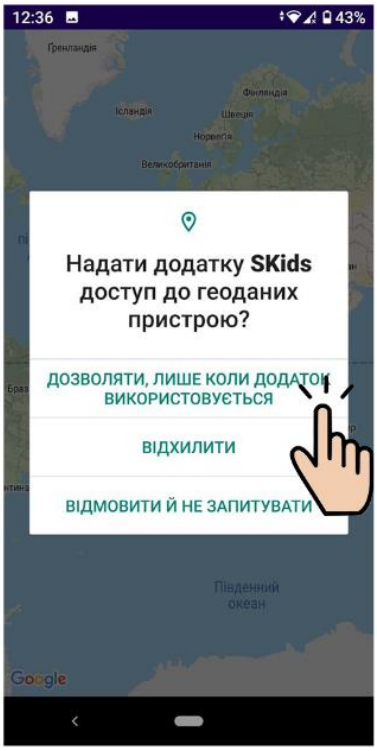
Тестування програми можна виконувати двома способами: на емуляторі або ж на реальному пристрої. Для того щоб запустити проект можна натиснути на кнопку «Run ‘app’» або скористатися комбінацією клавіш Shift+F10, в результаті чого з’явиться вікно, у якому можна вибрати спосіб запуску: або за допомогою емулятора, або під’єднаного пристрою. Також Android Studio надає можливість вибору віртуального пристрою, слідуючи інструкціям відповідного майстра.

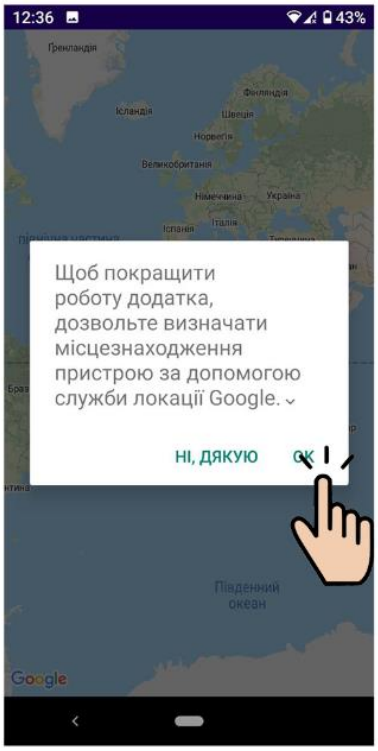
Але оскільки застосунок працює з сервісами Google та використовує GPS, то більш доцільно виконувати його тестування на реальному пристрої. Для цього на ньому потрібно в налаштуваннях увімкнути режим розробника та вибрати там прапорець режиму налагодження через USB (додаток Е). Після першого завантаження на телефон додаток доступний для використання. Для деяких пристроїв потрібні окремі драйвера, які необхідно шукати на офіційних сайтах виробників. Тестування програмного засобу проводилося на смартфоні Xiaomi Mi A2 (Android 10, API 29).

3.2 Тестування взаємодії з сервісом Google

При запуску програми на екрані смартфону з’являється іконка застосунку, при натисненні на яку, відкривається головна активність, з якої можна перейти у режим дитячого використання та у режим дорослого. Лише при першому вході до того чи іншого режиму додаток запитує дозвіл на доступ до геоданих пристрою, лише тоді коли додаток використовується. Для продовження взаємодії з додатком потрібно підтвердити такий дозвіл (рисунок 3.1(а)).

Наступним кроком програмний засіб виводить на перший план діалогове вікно, у якому вимагає дозволу для включення функції визначення місцезнаходження пристрою за допомогою служби локації Google (рисунок 3.1(б)). Таке діалогове вікно буде з'являтися кожного разу при вході в будь-який з режимів, якщо функція геолокації вимкнена, в протилежному випадку запит не буде відображатися.





а) доступ до геоданих пристрою б) функція визначення геолокації

Рисунок 3.1 — Налаштування дозволів

3.3 Тестування реєстрації та авторизації користувача

Під час переходу з головної активності у режим дорослого користувача з’являється інтерфейс, що зображений на рисунку 2.10. Першим кроком є реєстрація користувача за допомогою електронної адреси та паролю. Заповнивши відповідні поля та натиснувши на кнопку реєстрації система Firebase надсилає на вказану електронну адресу листа з підтвердженням (додаток Ж). І лише тільки при вдалій реєстрації користувач може рухатися далі, а саме авторизуватися. В результаті інтерфейс активності змінюється і користувача повідомляють під якою електронною адресою він зайшов, а також стає доступна функція виходу з акаунта (рисунок 2.12).

3.4 Тестування системи сповіщень

Головною метою створення даного програмного засобу є сповіщення батьків про порушення правил перебування дитини. Тому за для перевірки даного пункту необхідно змоделювати ситуацію, коли дитина знаходиться за

дозволеними межами у відповідний час. Для такого моделювання слід зазначити поточну дату та час, а саме 19 листопада 2020 14:10.

У батьківському режимі додамо місце яке є точно віддаленим від поточного та внесемо поточну дату, а діапазон часу вкажемо такий, щоб поточний час знаходився в ньому (рисунок 3.2).

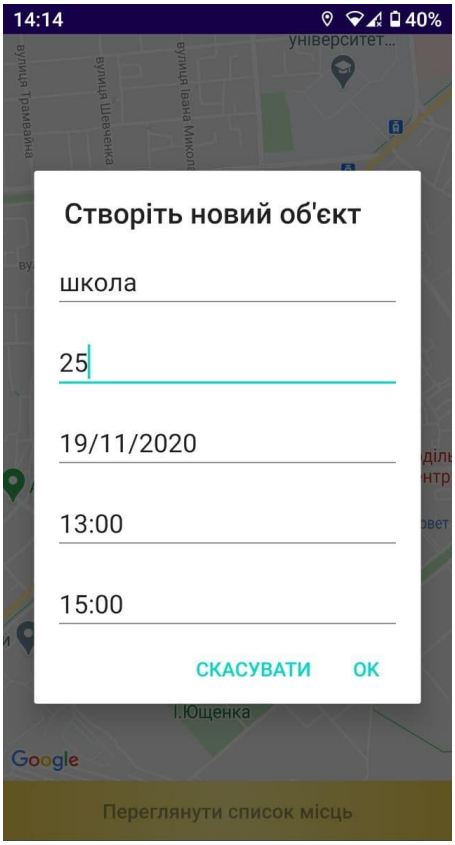


Рисунок 3.2 — Моделювання умов порушення правил

У результаті маємо реакцію на такі зміни у вигляді сповіщення у шторці телефону, що супроводжується звуковим сигналом, а також дає змогу відкрити активність з фрагментом карти при натисканні на нього (рисунок 3.3). Червоним маркером відмічається геолокація дитини, а зеленим — дорослого відповідно. Перевірка місця знаходження відбувається кожні три хвилини. Але слід зауважити одну важливу деталь, що коректна робота додатку ґарантується лише за наявністю стабільного під’єднання до мережі Інтернет, оскільки тільки за таких умов відбувається передача інформації. У випадку відсутності підключення до мережі, інформація буде занесена до БД і зчитана звідти при першій ж нагоді відновлення з’єднання.

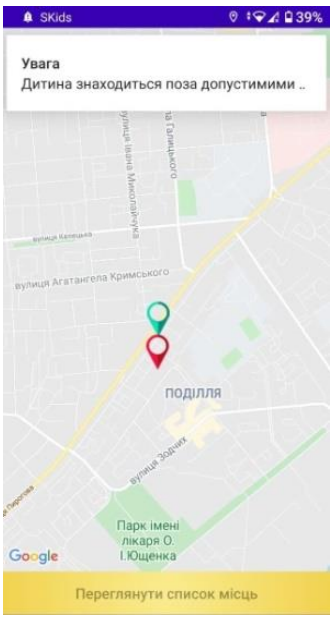


Рисунок 3.3 — Відображення сповіщення

3.5 Тестування взаємодії зі списком місць

Ще однією доступною для батьків функцією є перегляд збережених місць, а також їх редагування та видалення. Редагування об’єкту відбувається за допомогою діалогового вікна, яке з’являється при натисканні на кнопку з «олівцем в маркері».

Що ж стосується видалення, то для цього необхідно натиснути на кнопку «хрестик», після чого з’явиться діалогове вікно з попередженням, у якому потрібно підтвердити або скасувати подію (рисунок 3.4).

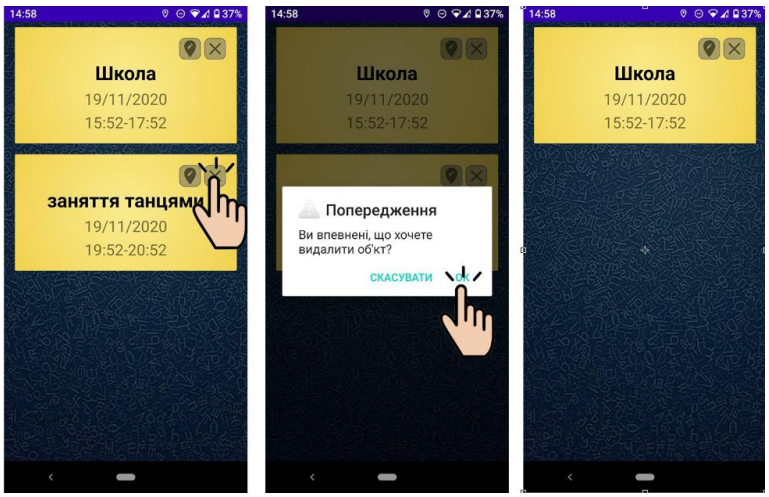


Рисунок 3.4 — Процес видалення об’єкта

У результаті виконання тестування розробленого програмного засобу на реальному пристрої проілюстровано функціонал застосунку, який відповідає раніше зазначеному. Інформація збирається, відтворюється та передається коректно, автоматичні обрахунки теж проводяться без помічених недоліків.

Інтерфейс користувача є доволі зручним та інтуїтивно зрозумілим, а за необхідністю супроводжується підказками для продуктивнішої та швидшої взаємодії між застосунком та користувачем. У ході тестування змодельовано різні можливі реальні ситуації, що слугували головними критеріями під час оцінки готовності проєкту.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки. Для проведення технологічного аудиту залучено 3-х експертів. Такими експертами виступають Богомолов С. В., Снігур А. В. та Войцеховська О. В. Здійснюємо оцінювання комерційного потенціалу розробки за 12-ма критеріями, наведеними в таблиці 4.1 за 5-ти бальною шкалою.

Таблиця 4.1 — Критерії оцінювання комерційного потенціалу розробки та їх бальна оцінка

Бали (за 5-ти бальною шкалою)					
Кри-те-Рій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція Підтверджена Експертними Висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздат-ність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому Ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни Аналогів
4	Технічні та споживчі властивості продукту значно гірші,ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші,ніж в ана-Логів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі,ніж в аналогів	Технічні та споживчі властивості продукту значно кращі,ніж в Аналогів
5	Експлуатаційні витрати значно вищі,ніж в аналогів	Експлуатаційні витрати дещо вищі,ніж в Аналогів	Експлуатаційні витрати на рівні експлуа-таційних ви-Трат аналогів	Експлуатаційні витрати трохи нижчі,ніж в аналогів	Експлуата-ційні витрати значно нижчі, ніж в аналогів

Продовження таблиці 4.1

Бали (за 5-ти бальною шкалою)					
Крите.	0	1	2	3	4
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної,так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної,так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси,які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси.Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали,що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10років	Термін реалізації ідеї більший за 5років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х Років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту,що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу розробки наведено в таблиці 4.2.

Таблиця 4.2 — Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	1. Богомолов С. В., к.т.н., доц. кафедри ОТ	2. Снігур А. В., к.т.н., доц. кафедри ОТ	3. Войцеховська О. В., доц. кафедри ОТ
	Бали, виставлені експертами:		
1	4	3	3
2	2	2	2
3	4	4	4
4	2	3	3
5	4	4	4
6	3	3	4
7	1	1	2
8	3	3	3
9	4	4	4
10	4	3	3
11	3	3	3
12	4	4	4
Сума балів	$СБ_1 = 38$	$СБ_2 = 37$	$СБ_3 = 39$
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = 38$		

Отже, з отриманих даних таблиці 4.2 видно, що нова розробка має рівень комерційного потенціалу вище середнього. Також дана розробка має соціологічний вплив, адже запобігає зникненню дітей, завчасно попереджуючи батьків про несподіване місцезнаходження дитини.

4.2 Прогнозування витрат на виконання науково-дослідної роботи та конструкторсько-технологічної роботи

Для розробки нового програмного продукту необхідні такі витрати. Основна заробітна плата для розробників визначається за формулою 4.1:

$$З_о = \frac{М}{Т_р} \cdot t \text{ грн.}, \quad (4.1)$$

де М — місячний посадовий оклад конкретного розробника;

$Т_р$ — кількість робочих днів у місяці, $Т_р$ = 21 дні;

t — число днів роботи розробника, t = 42 днів.

Розрахунки заробітних плат для керівника і програміста наведені в табл. 4.3.

Таблиця 4.3 — Розрахунки основної заробітної плати

Працівник	Оклад М, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Науковий керівник	8 447	402.25	3	1 206
Інженер-програміст	10 000	476.2	42	20 000
Всього:				21 206

Додаткова заробітна плата $З_д$ всіх розробників розраховується як 10 – 12% від суми основної заробітної плати всіх розробників:

$$З_д = (0{,}1 \ldots 0{,}12) \cdot З_о = 0{,}121\,206 = 2\,120{,}6 \text{ грн.}$$

Нарахування на заробітну плату $Н_{зп}$ для працівників бюджетної сфери становить 22% від суми основної та додаткової заробітної плати:

$$Н_{зп} = (З_о + З_д) \cdot 22\% = (21\,206 + 2\,120{,}6) \cdot 0{,}22 = 5\,131{,}85 \text{ грн.}$$

Розрахунок амортизаційних витрат для обладнання виконується за такою формулою:

$$А = \frac{Ц \cdot Н_а}{100} \cdot \frac{Т}{12},$$

де Ц — балансова вартість обладнання, грн;

$Н_а$ — річна норма амортизаційних відрахувань %;

Т — термін використання (Т=2 міс.).

Таблиця 4.4 — Розрахунок амортизаційних відрахувань

Найменування програмного забезпечення	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань, грн
Персональний комп'ютер	11 600	25	2	483,3
Смартфон	5 000	15	2	125
Всього:				608,3

Розрахуємо витрати на комплектуючі. Витрати на комплектуючі розрахуємо за формулою:

$$К = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i,$$

де n — кількість комплектуючих;

$Н_i$ — кількість комплектуючих і-го виду;

$Ц_i$ — покупна ціна комплектуючих і-го виду, грн;

$К_i$ — коефіцієнт транспортних витрат (приймемо $К_i$ = 1,1).

Таблиця 4.5 — Витрати на комплектуючі, що були використані для розробки програмного забезпечення.

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Кабель USB-Type C	шт.	43	1	43
Офісний папір формату А4	уп.	86	1	86
Ручка	шт.	8	1	8
Всього з урахуванням транспортних витрат				150,7

Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п ,$$

де В — вартість 1кВт-години електроенергії (В=0,9 грн/кВт);

П — установлена потужність комп'ютера (П=0,5кВт);

Ф — фактична кількість годин роботи комп'ютера (Ф=252 год.);

$К_п$ — коефіцієнт використання потужності ($К_п$< 1, $К_п = 0,6$).

$$В_е= 0,9 \cdot 0,5 \cdot 252 \cdot 0,6 = 68,04 \text{ грн.}$$

Інші витрати $І_в$ можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які були виконували дану роботу, тобто:

$$І_в=(1..3)\cdot(З_о+З_р)$$

Отже, розрахуємо інші витрати:

$$І_в = 1 \cdot (21\,206 + 2\,120,6) = 23\,326,6 \text{ грн.}$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи:

$$В = З_о + З_д + Н_{зп} + А + К + В_е + І_в$$

В = 21 206+2 120,6+5 131,85 +608,3+150,7+68,04+23 326,6 = 52 612,09 грн.

Розрахуємо загальну вартість наукової роботи $B_{заг}$ за формулою:

$$В_{заг} = \frac{В_{ін}}{\alpha},$$

де α — частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{52\,612,09}{1} = 52\,612,09 \text{ грн.}$$

Прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta},$$

де β — коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Отже, розрахуємо загальні витрати:

$$ЗВ = \frac{52\ 612{,}09}{0{,}9} = 58\ 457{,}87 \text{ грн.}$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Спрогнозуємо кількісно, яку вигоду можна отримати у майбутньому від впровадження результатів виконаної наукової роботи.

Виконання наукової роботи та впровадження її результатів буде здійснюватися протягом одного року. Основні позитивні результати від впровадження розробки очікуються протягом 3-х років після її впровадження. Одним із основних позитивних результатів є зростання величини прибутку.

При реалізації результатів наукової розробки покращується якість програмного продукту, що дозволяє підвищити ціну його реалізації на 200 грн. Кількість одиниць реалізації програмного засобу також збільшиться: протягом першого року — на 800 шт., протягом другого року — ще на 500 шт., протягом третього року — ще на 300 шт.

Реалізація продукції до впровадження результатів наукової розробки складала 40 шт., а ціна — 350 грн.

Спрогнозуємо збільшення чистого прибутку підприємства від впровадження результатів наукової розробки у кожному році відносно базового за такою формулою 4.2:

$$\Delta П_i = \sum_1^n (\Delta Ц_о \cdot N + Ц_о \Delta N)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\upsilon}{100}\right), \quad (4.2)$$

де $\Delta П_о$ — покращення основного оціночного показника від впровадження результатів розробки у даному році, зазвичай таким показником може бути ціна одиниці нової розробки;

N — основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN — покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$Ц_о$ — основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n — кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки;

λ — коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт 0,8333;

ρ — коефіцієнт, який враховує рентабельність продукту, що рекомендується приймати від 0,2 до 0,3;

υ — ставка податку на прибуток. υ = 18%.

Тоді, збільшення чистого прибутку підприємства протягом першого року складе:

$$\Delta П_1 = [200 \cdot 40 + (200 + 350) \cdot 800] \cdot 0{,}8333 \cdot 0{,}2 \cdot \left(1 - \frac{18}{100}\right) = 61\,224{,}22 \text{ грн}$$

Збільшення чистого прибутку підприємства протягом другого року (відносно базового року, тобто року до впровадження результатів наукової розробки) складе:

$$\Delta П_2 = [200 \cdot 40 + (200 + 350) \cdot (800 + 500)] \cdot 0{,}8333 \cdot 0{,}2 \cdot \left(1 - \frac{18}{100}\right) = 98\,806{,}05 \text{ грн}$$

Збільшення чистого прибутку підприємства протягом третього року (відносно базового року, тобто року до впровадження результатів наукової розробки) складе:

$$\Delta П_3 = [200 \cdot 40 + (200 + 350) \cdot (800 + 500 + 300)] \cdot 0{,}8333 \cdot 0{,}2 \cdot \left(1 - \frac{18}{100}\right) = 121\,355{,}15 \text{ грн}$$

Отже, протягом трьох років підприємство може розраховувати на збільшення чистого прибутку від реалізації наукової розробки.

4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

Визначимо абсолютну і відносну ефективність вкладених інвестором інвестицій та розрахуємо термін окупності.

Абсолютна ефективність $E_{абс}$вкладених інвестицій розраховується за формулою:

$$E_{абс} = (ПП - PV),$$

де ПП — приведена вартість всіх чистих прибутків, які отримає підприємство (організація) від реалізації результатів наукової розробки, грн.;

PV — теперішня вартість інвестицій PV = ЗВ = 58 457,87 грн.

Рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати вигляд, рисунок 4.1.

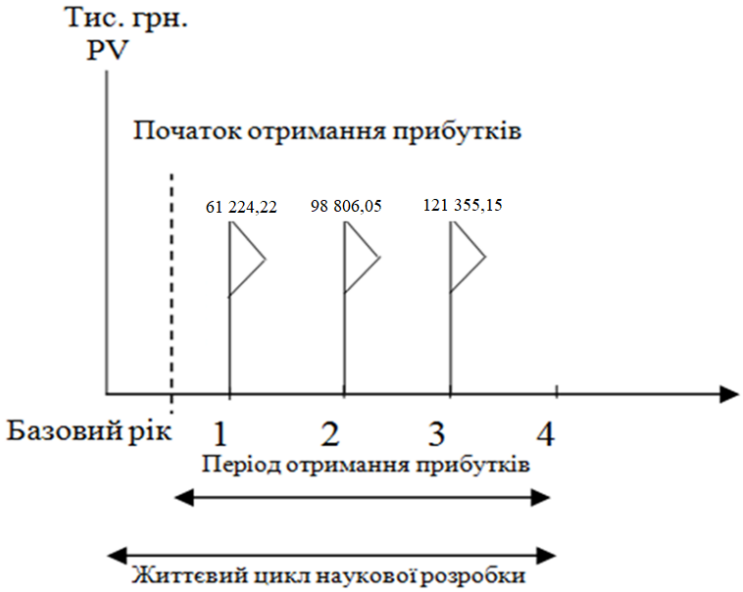


Рисунок 4.1 — Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів наукової роботи

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_1^T \frac{\Delta П_i}{(1+\tau)^t},$$

де $\Delta П_i$ — збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т — період часу, протягом якого виявляються результати впровадженої НДДКР, 3 роки;

τ — ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, для України цей показник знаходиться на рівні 0,1;

t — період часу (в роках) від моменту отримання чистого прибутку до точки “0”.

Отже, розрахуємо вартість чистого прибутку:

$$\text{ПП} = \frac{58\,457{,}87}{(1+0{,}1)^0} + \frac{61\,224{,}22}{(1+0{,}1)^2} + \frac{98\,806{,}05}{(1+0{,}1)^3} + \frac{121\,355{,}15}{(1+0{,}1)^4} = 266\,178{,}05 \text{ грн.}$$

Тоді розрахуємо $E_{\text{абс}}$:

$$\text{Е}_{\text{абс}} = 266\,178{,}05 - 58\,457{,}87 = 207\,720{,}18 \text{ грн.}$$

Оскільки $E_{\text{абс}} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР буде доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $\text{Е}_\text{в}$ за формулою:

$$\text{Е}_\text{в} = \sqrt[\text{Т}]{1 + \frac{\text{Е}_{\text{абс}}}{\text{PV}}} - 1,$$

де $\text{Е}_{\text{абс}}$ — абсолютна ефективність вкладених інвестицій, грн;

PV — теперішня вартість інвестицій PV = ЗВ, грн;

Т — життєвий цикл наукової розробки, роки.

Тоді будемо мати:

$$\text{Е}_\text{в} = \sqrt[3]{1 + \frac{207\,720{,}18}{58\,457{,}87}} - 1 = 0{,}66 \text{ або } 66\,\%$$

Далі, розраховану величина $\text{Е}_\text{в}$ порівнюємо з мінімальною (бар'єрною) ставкою дисконтування $\tau_{\text{мін}}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{\text{мін}}$ визначається за формулою:

$$\tau = \text{d} + \text{f},$$

де d — середньозважена ставка за депозитними операціями в комерційних банках, в 2020 році в Україні d = 0,2;

f — показник, що характеризує ризикованість вкладень, величина f = 0,1.

$$\tau = 0{,}2 + 0{,}1 = 0{,}3$$

Оскільки Ев = 66% > $\tau_{мін}$ = 0,3 = 30%, то інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

Термін окупності вкладених у реалізацію наукового проекту інвестицій Ток розраховується за формулою:

$$T_{ок} = \frac{1}{E_в},$$

Якщо $T_{ок}$< 3…5-ти років, то фінансування наукової розробки є доцільним.

$$T_{ок} = \frac{1}{0,66} = 1,5 \text{ роки}$$

Обрахувавши термін окупності даної наукової розробки, можна зробити висновок, що фінансування даної наукової розробки буде доцільним, оскільки $T_{ок}$<5 років.

Результати проведених розрахунків дають можливість зробити висновок про доцільність розробки та впровадження нашої наукової роботи. Це підтверджують такі показники як:

— абсолютна ефективність вкладених інвестицій, яка дорівнює 207 720 грн., що є більшим 0 і вказує на те, що інвестор може бути зацікавленим у нашій розробці;

— відносна ефективність наукової розробки становить 66%, що є вищим за мінімальну ставку дисконтування (30%), тому вкласти кошти у нашу розробку є вигідніше, ніж покласти кошти на депозит;

— термін окупності вкладених у реалізацію наукового проекту інвестицій складе 1,5 роки, що є менше 5-ти і вказує швидку окупність вкладених інвестицій. Крім того, розраховано, що наукова розробка приноситиме підприємству додатковий прибуток протягом 3-х років за рахунок покращення її якості порівняно з існуючими аналогами.

ВИСНОВКИ

У сучасному світі діти зустрічаються із різноманітними небезпеками, і не завжди в дорослих є можливість бути поряд з дитиною аби вберегти її від них. Тому головним завданням під час розробки було забезпечення дорослих можливостями батьківського контролю, а саме можливість отримувати сповіщення про підозріле місцеперебування дитини.

У магістерській кваліфікаційній роботі був розроблений програмний засіб батьківського контролю. Розробка виконана на базі мобільної операційної системи Android. Під час створення програми велика увага приділялася коректній обробці інформації, раціональному використанню ресурсів мобільного пристрою, а також зручному та привабливому дизайну. Розробка застосунку для Android вимагала глибоких знань і розуміння принципів ефективної взаємодії користувача з додатком. Також розробка може бути вдосконалена в подальшому за допомогою розширення функціональних можливостей, а також покращення способів сповіщення за умов відсутності доступу до мережі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Служба розшуку дітей [Електронний ресурс]. — Режим доступу: https://missingchildren.org.ua/

2. Нативні додатки [Електронний ресурс]. — Режим доступу: https://cheport.com.ua/ua/blog/native—app.

3. Advantages and disadvantages of android operating system [Електронний ресурс]. — Режим доступу: https://www.itrelease.com/2019/09/advantages—and—disadvantages—of—android—operating—system/

4. Advantages and disadvantages of iOS operating system [Електронний ресурс]. — Режим доступу: https://www.itrelease.com/2020/09/advantages—and—disadvantages—of—ios—operating—system/

5. 5 перевірених програм «батьківського контролю» під Android [Електронний ресурс]. — Режим доступу: https://kidsvisitor.com/uk/news/297—5—perevirenikh—program—batkivskogo—kontroliu—pid—android/

6. Евристичне оцінювання [Електронний ресурс]. — Режим доступу: https://uk.wikipedia.org/wiki/ Евристичне оцінювання.

7. Top Android IDEs for Developers [Електронний ресурс]. — Режим доступу: https://www.developer.com/ws/android/development—tools/top—android—ides—for—developers.html.

8. Яхненко О. В. Розробка Android—додатків на Android Studio під Ubuntu на Kotlin. FOSS LVIV: матеріали міжнар. наук.—практ. конф., м. Львів, 19— 22 квіт. 2016 р. Львів, 2016. С. 114—119.

9. Java и Kotlin: что будет лучшим выбором в 2019 году? [Електронний ресурс]. — Режим доступу:https://techrocks.ru/2019/01/28/java—vs—kotlin—best—language—for—android—in—2019/

10. Firebase Realtime Database [Електронний ресурс]. — Режим доступу: https://firebase.google.com/docs/database/

11. Введение в разработку приложений для ОС Android [Електронний ресурс]. — Режим доступу: https://www.intuit.ru/studies/courses/12643/1191/lecture/21983.

12. Програмування під Android/Структура Android програми [Електронний ресурс]. — Режим доступу: https://uk.wikibooks.org/wiki/Programmingfor Android/Android Structure Applications.

13. Distribution dashboard [Електронний ресурс]. — Режим доступу: https://developer.android.com/about/dashboards/index.html.

14. Google Cloud Platform [Електронний ресурс]. — Режим доступу: https://console.cloud.google.com/home/dashboard?project=skids-291814&hl=ru

15. Learning Android Google Maps / W. Raj Amal, Vikash Kumar Karn — Birmingham: Packt Publishing Ltd, 2015. — 356 p.

16. Android — Time Picker [Електронний ресурс]. — Режим доступу: https://www.tutorialspoint.com/android/android_timepicker_control.htm

17. Богомолов С. В. Програмно-апаратний комплекс управління об'ктами через інтернет / Богомолов С. В., Богомолова М. В., Чолій В. О. // Вимірювання, контроль та діагностика в технічних системах. Тези доповіді другої Міжнародної науково-практичної конференції. м. Вінниця, 29-31 жовтня 2013 року. — Вінниця: ВНТУ, 2013. — С.257.

18. Високолінійний зовнішній аудіо цап з USB інтерфейсом / Богомолов С. В., Богомолова М. В., Озеруга Д. В. // Вимірювання, контроль та діагностика в технічних системах. Тези доповіді другої Міжнародної науково-практичної конференції. м. Вінниця, 29-31 жовтня 2013 року. — Вінниця: ВНТУ, 2013. — С.133.

19. Богомолов С. В. Мережева комп'ютерна система офіційного теле-радіо моніторингу з підсистемою точного часу / Азаров О. Д., Крупельницкий Л. В., Богомолов С. В., Хурса Н. А., Шиманський В. В. // м. Львів — 2013.