Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)
Інститут деревообробних та комп'ютерних технологій і дизайну
(повне найменування інституту, назва факультету (відділення))
Кафедра інформаційних технологій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до дипломного проекту (роботи)
бакалавр
(рівень вищої освіти)

на тему ***Розроблення мобільного застосунку для туристів міста Париж***

Виконав: студент 5к., групи КНз-51
Спеціальності
122 Комп'ютерні науки
(шифр і назва спеціальності)
Фляк Роман Михайлович
(прізвище та ініціали)
Керівник: доцент кафедри ІТ ***Пірко І.Б.,***
асистент ***Опришко М. І.***
(прізвище та ініціали)
Рецензент к.т.н., ст. викладач
каф. АКІТ ***Мацишин Я.В.***
(прізвище та ініціали)

Львів 2022

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально науковий інститут деревообробних та комп'ютерних технологій і дизайну
Кафедра інформаційних технологій
Рівень вищої освіти бакалавр
Спеціальність 122 – "Комп'ютерні науки "
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТ
_____Крошний І.М.
"____" _____202_ року

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

Фляк Роман Михайлович
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) ***Розроблення мобільного застосунку для туристів міста Париж***

затверджені наказом вищого навчального закладу від "13" 12 2021 р. № С-618

2. Строк подання студентом проекту (роботи) ***11.04.2022***

3. Вихідні дані до проекту (роботи) ***Постановка завдання та його формалізація. Алгоритм побудови мобільних застосунків. Вихідні дані та зразки схожих систем. Графічне представлення вхідних та вихідних даних. Література за тематикою роботи.***

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

СУЧАСНИЙ СТАН ПРОБЛЕМНОЇ ОБЛАСТІ
ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ ТА ЗАСОБИ РЕАЛІЗАЦІЇ
РОЗРОБЛЕННЯ СИСТЕМИ
ВИСНОВОК

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) ***Підготовка матеріалів до доповіді***

6. Дата видачі завдання 15 грудня 2021 року

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Етапи бакалаврської дипломної роботи	Термін виконання етапів роботи	Примітка
1.	1. Огляд сучасного стану задачі, методів і засобів її вирішення. 2. Формування функціональних вимог та постановка задачі проекту. 3. Оформлення першого розділу пояснювальної записки.	02.12.2021 12.12.2021	
2.	1. Огляд інформаційного забезпечення. 2. Огляд існуючих систем та модулів. 3. Оформлення другого розділу пояснювальної записки.	13.01.2022 23.01.2022	
3.	1. Вибір програмних засобів для реалізації системи.	08.02.2022 27.02.2022	
4.	1. Програмна реалізація інформаційної системи. 2. Оформлення третього розділу пояснювальної записки.	05.03.2022 21.03.2022	
5.	1. Технічна та апаратна організація проекту.	01.04.2022	
6.	1. Здача пояснювальної записки на кафедру.	07.04.2022	

Студент _____ ***Фляк Р.М.***
(підпис) (прізвище та ініціали)

Керівник роботи _____ ***Опришко М.І.***
(підпис) (прізвище та ініціали)

Керівник роботи _____ ***Пірко І.Б.***
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

У дипломній роботі розроблено програмне забезпечення Андроїд-застосунку для швидкого пошуку необхідних туристам Парижу інформації та сервісів. На головному екрані відображається найбільш необхідні туристам сервіси (оренда житла, оренда авто, ресторани і т.д.) та окремо список найпопулярніших локацій міста. Реалізовано екран детального представлення кожного із представлених пунктів.

Користувачу надано можливість переглядати на карті найближчі кафе та ресторани міста. Реалізовано систему пуш-сповіщень, які будуть відображатись із заданою періодичність та пропонуватимуть доступні екскурсії.

У програмі створено бічне меню, де користувач зможе знайти відповіді на свої запитання стосовно екскурсій та отримати інформацію про власників проекту. У окремому пункті меню для користувача є можливість залишити свої побажання чи зауваженя.

Пояснювальна записка до дипломної роботи бакалавра складається з 56 сторінок, 22 рисунків і містить 3 додатки.

Ключові слова: Модуль, БД, API, IDE, RSS, КПК, ПК, ОС, APK, XML, , Сервіс, Система.

ABSTRACT

In the thesis, the software of the Android application was developed to quickly find the information and services needed by Paris tourists. The main screen displays the most necessary services for tourists (rental housing, car rental, restaurants, etc.) and a separate list of the most popular locations in the city. The screen of detailed presentation of each of the presented points is realized.

The user is given the opportunity to see on the map the nearest cafes and restaurants in the city. A system of push notifications has been implemented, which will be displayed at a specified frequency and will offer affordable tours.

The program has a side menu where the user can find answers to their questions about excursions and get information about the owners of the project. In a separate menu item, the user has the opportunity to leave their wishes or comments.

The explanatory note to the bachelor's thesis consists of 56 pages, 22 figures and contains 3 appendices.

Keywords: Module, DB, API, IDE, RSS, PDA, PC, OS, APK, XML, Service, System.

ТЕХНІЧНЕ ЗАВДАННЯ

Розробити та програмне забезпечення Андроїд-застосунку для швидкого пошуку послуг, інформації та сервісів необхідних туристам Парижу. На стартовому екрані аплікації повинна відображатись анімація завантаження.

На головному екрані потрібно відобразити найбільш необхідні туристам сервіси (оренда житла, оренда авто, ресторани і т.д.) та окремо список найпопулярніших локацій міста. Реалізувати екран детального представлення кожного із представлених пунктів.

Надати користувачу можливість переглядати на карті найближчі кафе та ресторани міста.

Реалізувати систему пуш-сповіщень, які будуть відображатись із заданою періодичність та пропонуватимуть доступні екскурсії.

У програмі має бути бічне меню, де користувач зможе знайти відповіді на свої запитання стосовно екскурсій та отримати інформацію про власників проекту. У окремому пункті меню користувачу повинні бути доступними контакти власників проекту для залишення своїх побажань чи зауважень.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

APK - Android Package

IDE - Integrated Development Environment

MVC - Model-View-Controller

MVP - Model-View-Presenter

MVVM - Model-View-View Model

БД – база даних

КПК - Кишеньковий персональний комп'ютер

ОС - Операційна система

ПК - Персональний комп'ютер

API - Application programming interface

RSS - англ. Rich Site Summary

XML-формат - EXtensible Markup Language

ВСТУП

На даний момент завдання розробки нових додатків, як і раніше, залишається актуальним. Адже це найзатребуваніший сегмент інформаційного ринку. Це, безперечно, пов'язано з міцним входженням мобільних пристроїв у повсякденне життя людини. За допомогою мобільних пристроїв люди звикли отримувати інформацію та обмінюватися нею з іншими, робити покупки, спілкуватися та знайомитися. Їх практична користь у бізнес-середовищі очевидна - контроль за станом прибутків і справ, відстеження важливих повідомлень та можливість обробляти великі обсяги інформації, доступ до сервісів оплати та прочитання аналітичних звітів і т.п.

Функціонал програми для туристів може бути різним, і залежить від потреб бізнесу та сфери діяльності. Тому ми спочатку визначаємо завдання та вибудовуємо чітку стратегію розвитку програми. Нам важливо, щоб програма для туризму суворо виконувала поставлені завдання. І тому впроваджуються певні інструменти. Вони різняться залежно від цього, кому розробляється додаток. Адже користувачами можуть бути не лише мандрівники, а й працівники турагентства.

Актуальність дослідження передусім зумовлена необхідністю просування туристичних послуг до м. Париж.

Об'єктом дослідження - є засоби створення Андроїд-застосунків - Java, Kotlin.

Предмет дослідження – методи, алгоритми синтаксичного аналізу, відображення та візуалізації матеріалів на мобільному додатку.

Мета роботи - створення Андроїд-додатку для туристів м.Париж.

Практичне значення – створюється унікальний продукт за закордонними аналогами, адаптований під інфраструктуру м. Париж, що дає інформаційну зручність та комфорт діловим туристом міста.

1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1 Мобільний додаток для туризму

Туристична сфера бізнесу побудована на взаємодії з клієнтом, наданні якісних послуг та вирішенні завдань. Лояльність клієнта – основна "валюта" будь-якого туристичного бізнесу. Програми для туристів – це інструмент комфортної та ефективної взаємодії між турагентством та мандрівником.

Навіщо вашому туристичному бізнесу мобільний додаток? Мобільні програми для туристів приносять велику користь, а своїм власникам – зиск. Це інструмент комунікації із клієнтом. Завдання полягає в тому, щоб зробити програму для туризму, без якої користувач не зможе представити свій найкращий відпочинок. Це не просто інструмент додаткових продажів. Це персональний помічник, який допомагає вирішити питання, розібратися у конфліктних та проблемних ситуаціях, підказати та допомогти знайти потрібний об'єкт в іншій країні.

Рисунок 1.1. Користувачі мобільного застосунку.

Що це дає бізнесу? Насамперед, лояльного клієнта, який стає постійним. Підвищується впізнаваність бренду, збираються позитивні відгуки. Це можливість додаткової монетизації бізнесу. Навіть якщо клієнт користується послугами іншої компанії, додаток для подорожей стане йому

корисним. Ви можете продавати додаткові послуги, товари та взаємодіяти з мандрівником, навіть якщо він не є вашим клієнтом.

Основний функціонал туристичних мобільних додатків. Функціонал програми для туристів може бути різним, і залежить від потреб бізнесу та сфери діяльності. Тому ми спочатку визначаємо завдання та вибудовуємо чітку стратегію розвитку програми. Нам важливо, щоб програма для туризму суворо виконувала поставлені завдання. І тому впроваджуються певні інструменти. Вони різняться залежно від цього, кому розробляється додаток. Адже користувачами можуть бути не лише мандрівники, а й працівники турагентства.

Для компаній туристичної сфери бізнесу, як правило, впроваджується наступний функціонал:

- сторінка з інформацією про компанію;
- список турів;
- каталог додаткових послуг (страхування, гід, перекладач та інші);
- каталог закладів та визначних пам'яток під країни, в які є тури;
- сторінка акцій та гарячих пропозицій;
- розклад рейсів, інтеграція із сервісами замовлення квитків;
- робота з картами та геолокацією;
- фільтри, сортування та інструменти швидкого пошуку;
- інструменти оформлення замовлення, наповнення документів.

Залежно потреб компанії вибираються необхідні інструменти. Завдання полягає в тому, щоб спростити роботу співробітників, прискорити та покращити якість обслуговування клієнтів та зробити додаток для подорожей легким, доступним та зрозумілим.

Додатки для туристів впроваджуються, як правило:

- пошук готелів по країні з рейтингом та відгуками;
- картка з відображенням значних місць;

- інформаційні сторінки, що допомагають туристу зібратися у подорож;
- рейтинги ресторанів, готелів та інших закладів;
- інструменти взаємодії з об'єктами, можливість залишати відгуки, додавати фото;
- візуальний контент;
- фільтр та пошук потрібних об'єктів;
- інформаційні сторінки із цінним контентом;
- необхідні номери екстрених служб;
- кнопки зв'язку з представниками компанії.

Функціонал безпосередньо залежить від типу програми. Якщо це програма для перекладу – її можливості одні, якщо це путівник із картою місцевості – завдання ставляться зовсім інші. Саме тому перед розробкою ми вивчаємо потреби, щоб програма для подорожей була цільовою та максимально корисною.

Програми для туристів мають свої особливості, і кожен створюється із власним функціоналом. Головне, щоб така програма була корисна для мандрівників, а власникам приносило гроші. Ось кілька прикладів успішних реалізованих проектів:

- **AroundMe** . Додаток, який допомагає знайти туристу пам'ятки, заклади, готелі та інші місця у невідомому місті. За допомогою геолокації програма розуміє, де знаходиться турист, і показує всі важливі об'єкти, включаючи банкомати, кінотеатри, готелі та ресторани.
- **MAPS.ME.** _ Дуже корисна програма для подорожей, адже тут є карти, які працюють без інтернету. Досить заздалегідь завантажити їх. Відображено всі локації, можна побудувати маршрут, знайти потрібні об'єкти.

- **Uber** . Як не дивно, але це теж програма для подорожей, яка працює в 77 країнах. Замовлення таксі у незнайомому місті – чудова ідея для будь-якого туриста.
- **Wiffinity** . Додаток для туристів, які не можуть без інтернету, та не бажають платити за роумінг. Це велика база точок Wi-Fi у всьому світі з паролями. За геолокацією програма визначає місце розташування туриста і пропонує підключитися до доступного інтернету.
- **Packpoint** . Справжній турист помічник. Тут є список речей, які необхідно взяти в подорож, програма подивиться погоду, нагадає про заходи та місця, які слід відвідати. Це планувальник для мандрівників.

Туристичні програми приносять величезну користь для користувачів, а для власників – це інструмент додаткових продажів послуг та товарів, можливість взаємодіяти з клієнтом, залишатися в полі зору та бути затребуваним.

1.2 Огляд аналогових систем

Квитки та готелі в ПАРИЖІ. Цей довідник по Парижу є надійним і простим у використанні супутником у подорожі. Тут можна знаходити маршрути з детальними офлайн-картами, детальним туристичним вмістом, популярними пам’ятками та інсайдерськими порадами за допомогою цього довідника по Парижу.

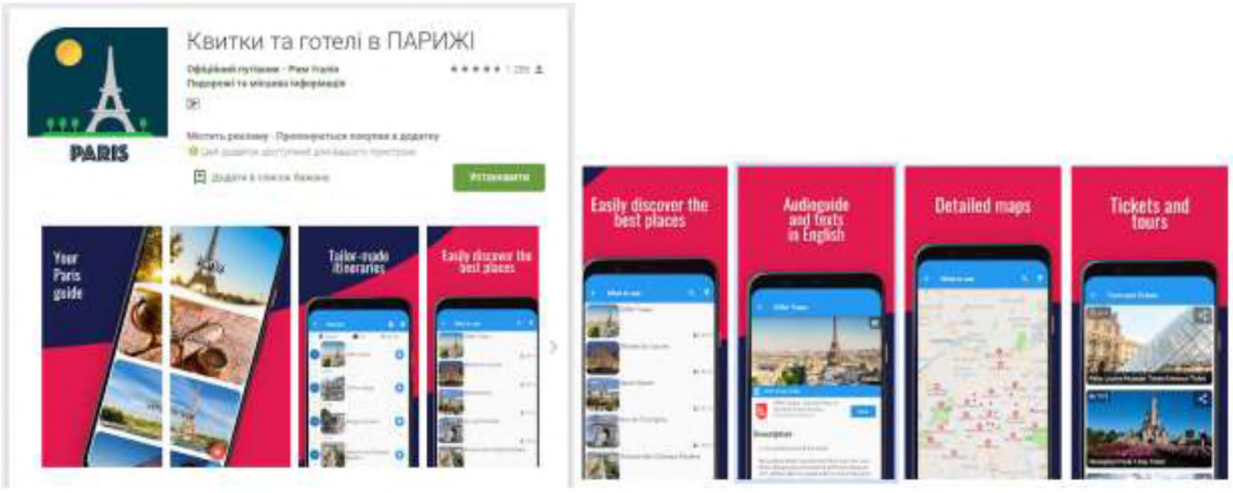


Рисунок 1.2. Додаток Квитки та готелі в Парижі.

Плануйте та влаштуйте ідеальну подорож! Забронюйте готель і насолоджуйтеся відгуками про ресторани та ділим вмістом користувачів.

Ось чому понад 15 мільйонів мандрівників люблять офлайн-путівники Парижа та міста:

Хіба вам не завжди хотілося мати легко портативного і компактного помічника, який дозволить заздалегідь планувати ваші поїздки в закордонні країни та міста? Тож перетворите свій смартфон або планшет на цифровий довідник і планувальник міста Парижа, який проведе вас через ВАШ вибір ресторанів, готелів та визначних пам'яток для відвідування. Насолоджуйтесь рекомендаціями та відгуками інших мандрівників і туристів-ентузіастів. Завжди зберігайте орієнтацію і знайдіть напрямок до наступного місця; повністю без роумінгу та офлайн.

З цією офлайновою картою Парижа та довідником по місту ви отримаєте широкий спектр переваг:

БЕЗКОШТОВНО. Немає абсолютно ніякого ризику, і ми впевнені, що вам це сподобається!

ДЕТАЛЬНІ МАПИ. Ніколи не заблукайте і зберігайте орієнтацію. Переглядайте своє місцезнаходження на офлайн-карті Парижа навіть без підключення до Інтернету. Знайдіть вулиці, визначні пам'ятки, ресторани, готелі, місцеве нічне життя та інші об'єкти – і отримуйте напрямок у напрямі прогулянок до місць, які ви хочете побачити.

ГЛИБИНИЙ ВМІСТ ПОДОРОЖІВ. Мати всю інформацію в автономному режимі та вільно переносити. Для кожного пункту призначення отримайте доступ до вичерпної та актуальної інформації про тисячі місць, визначних пам'яток, визначних місць та багато варіантів бронювання готелів у цьому путівнику по Парижу.

ШУКАЙТЕ ТА ВІДКРИВАЙТЕСЯ. Знайдіть найкращі ресторани, магазини, визначні пам'ятки, готелі, бари тощо. Шукайте за назвою,

переглядайте за категоріями або знаходьте місця поблизу за допомогою GPS свого пристрою – навіть офлайн і без роумінгу даних.

ОТРИМАТИ ПОРАДИ ТА РЕКОМЕНДАЦІЇ. Знайдіть поради та рекомендації від місцевих жителів та туристів. Переглядайте в режимі офлайн у цьому довіднику Парижа найпопулярніші визначні пам’ятки, ресторани, магазини, готелі, місця нічного життя тощо.

ПЛАНУЙТЕ ПОЇЗДКИ. Створіть списки місць, які ви хочете відвідати. Закріпіть наявні місця, як-от ваш готель чи рекомендований ресторан, на карті. Додайте власні шпильки на карту. Знайдіть і забронюйте готелі в цьому путівнику по Парижу.

ОФЛАЙН ДОСТУП. Офлайн-карта Парижа та вміст Путівника по місту Парижа повністю завантажуються та зберігаються на вашому пристрої. Усі функції, такі як пошук адреси та ваше місцезнаходження GPS, також працюють в автономному режимі та без роумінгу даних (звісно, для початкового завантаження даних або бронювання готелів потрібне підключення до Інтернету).

КВИТКИ. Знайдіть квитки на найважливіші місця Парижа: Лувр, Ейфелеву вежу, Нотр-Дам, катакомби та інші!

Paris travel guide. Цей додаток є надійним і простим у використанні супутником у подорожах. Знайти напрямках з докладними оффлайн карти, глибокий зміст подорожі, популярних пам'яток і інсайдерської поради з цим Парижі міський керівництва. Планувати і мають ідеальне подорож! Забронюйте готель і насолодитися огляди ресторанів і спільного користувальницький контент. Ось чому 15 + мільйонів пасажирів люблю Ulmon на форумі і путівники:

Не завжди хочеться мати легко портативний і компактний помічник в дорозі, що дозволяє планувати свої поїздки в зарубіжні країни та міст авансом? Так перетворити ваш смартфон або планшет в цифрову Путівник по Парижу міста і планувальник веде вас через ваш вибір ресторанів, готелів і які пам'ятки

відвідати. Насолоджуйтесь рекомендації та відгуки інших захоплених мандрівників і туристів. Завжди тримайте орієнтацію і знайти напрямок до наступного місця; абсолютно без роумінгу і в автономному режимі.

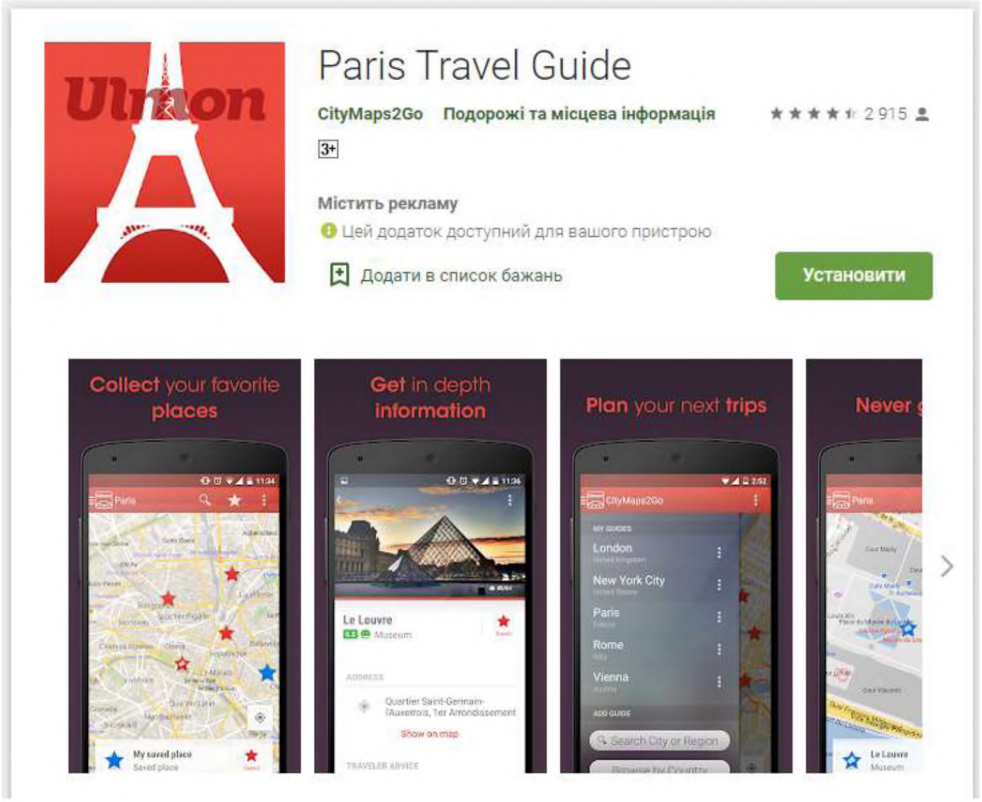


Рисунок 1.3. Додаток Paris travel guide.

З цією Паризької форумі карти і керівництва міста, вам сподобається широкий ряд переваг:

Безкоштовно. Просто скачайте і спробуйте це Париж путівник по місту безкоштовно. Там немає абсолютно ніякого ризику, і ми впевнені, що ви будете любити його!

Докладні карти. Ніколи не заблукаєте і зберегти вашу орієнтацію. Дивыться своє місце розташування на Паризькій карті, навіть без підключення до інтернету. Знайти вулиці, пам'ятки, ресторани, готелі, місцеві нічні клуби та інші об'єкти інфраструктури - і замовити екскурсії у хвилинах напрямку місця, де ви хочете побачити.

ПОГЛИБЛЕНОГО TRAVEL ЗМІСТ. Мати всю інформацію в автономному режимі і вільно портативний. Для кожного призначення, доступ всеосяжної та

актуальної інформації охоплює тисячі місць, пам'яток, пам'яток і багато варіантів бронювання готелів в Парижі путівник.

Пошук і вивчення. Знайти кращі ресторани, магазини, пам'ятки, готелі, бари і т.д. Пошук по імені, перегляд по категоріях або виявити довколишні місця, використовуючи GPS вашого пристрою - навіть в автономному режимі і без передачі даних в роумінгу.

Отримуйте поради та рекомендації. Знайти поради та рекомендації від місцевих жителів і туристів. Перегляд форуму в цьому Паризької керівництва для найбільш популярних пам'яток, ресторанів, магазинів, готелів, нічних клубів місцях, і т.д.

Планувати поїздки і налаштувати карти. Створення списків місць, де ви хочете відвідати. Pin існуючих місцях, як готелі або рекомендованою ресторані, до карти. Додати свої власні контакти на карту. Знайти і забронювати готелі зсередини цієї Паризької керівництва міста.

Автономний доступ. Париж форуму карта і зміст Париж путівник по місту повністю завантажуються і зберігаються на вашому пристрої. Всі функції, такі як адреса пошуків і вашого місця розташування GPS також працювати в автономному режимі і без передачі даних в роумінгу (підключення до інтернету, звичайно, необхідні для початкового завантаження даних або для бронювання готелів).

ЯКІСТЬ ДАНИХ: Дані карти надаються OpenStreetMap і регулярно оновлюється нас. Щоб перевірити рівень деталізації, перейдіть до www.openstreetmap.org. Те ж саме відноситься до Вікіпедії статей подорожі.

Paris Guide by Civitatis. Путівник по Парижу Civitatis.com включає всю необхідну та сучасну туристичну інформацію, щоб максимально використати поїздку до столиці Франції. Париж відомий як Місто світла, оскільки це було першим містом, яке прийняло газове вуличне освітлення. У нашому путівнику ви знайдете інформацію про найкращі пам'ятки міста, де поїсти, поради щодо економії грошей та багато іншого корисної інформації.

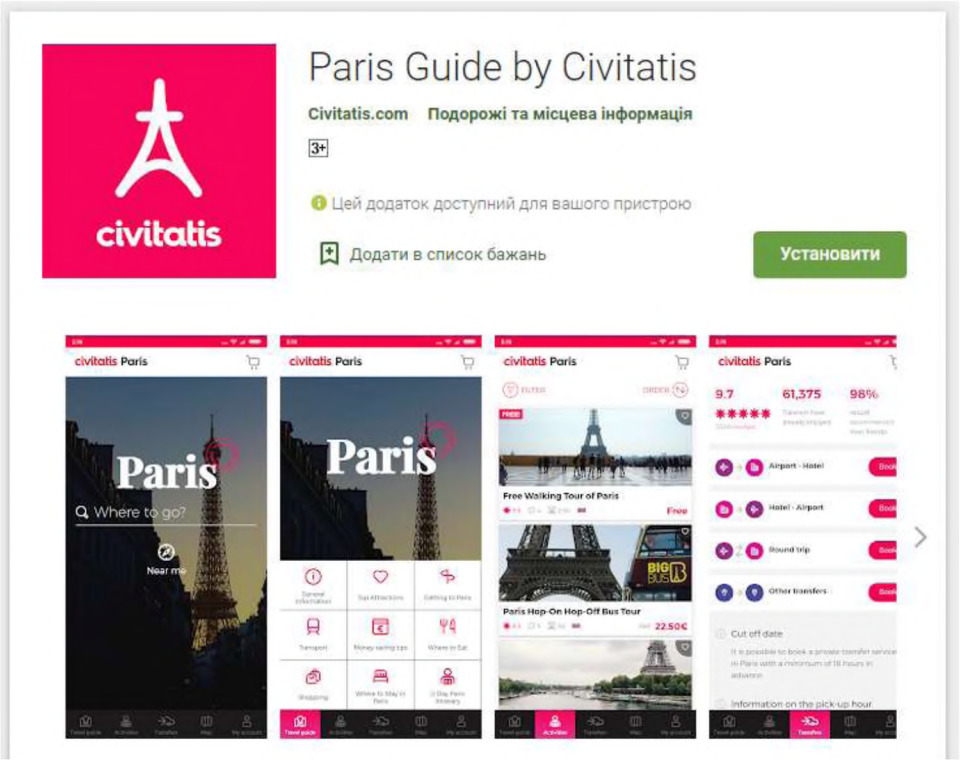


Рисунок 1.4. Додаток Paris Guide by Civitatis.

• Найпопулярніші пам'ятки Парижа: Відкрийте найкращі місця для відвідування та відвідування Парижа та дізнайтеся, як дістатися, години роботи, ціни та дні, коли пам’ятки закриті.

• Де поїсти: Відкрийте для себе французьку кухню та найкращі ресторани та бари, щоб спробувати її традиційні страви.

• Підказки щодо економії грошей: Відвідайте Паризький перевал, туристичну карту Парижа, де можна отримати найкращі знижки, атракціони з хорошим співвідношенням якість / ціна… у нашому довіднику є багато порад щодо заощадження грошей, які допоможуть вам у поїздці до Париж.

• Де зупинитися: Найкращі квартали для зупинки, райони, яких слід уникати, як знайти найкращі пропозиції щодо готелів та апартаментів та багато іншого корисної інформації.

• Інтерактивна карта: на нашій інтерактивній карті ви зможете планувати відвідування кращих музеїв та визначних пам'яток міста пішки або на машині.

Крім корисної туристичної інформації, ми також пропонуємо наступні послуги:

• Англомовні екскурсії: Прогулянки та екскурсії Парижем з англомовним путівником, включаючи екскурсію найкращими речами, які можна побачити в Парижі, прогулянку Монмартром та Сакре-Кер.

• Цілодобові подорожі: ми пропонуємо одноденні поїздки до Версаля, Шато долини Луари, Брюгге, Мон Сен Мішель та інших найпопулярніших напрямках, які завжди супроводжуються англомовним гідом.

• Круїзи по річці Сена: романтичний та розслабляючий круїз по річці Сена. Ми пропонуємо денні та нічні прогулянки на човні, обідом або вечерею на борту.

• Трансфер з аеропорту: якщо ви хочете комфортну, дешеву та безпроблемну подорож від аеропорту до вашого готелю, наші шофери чекають на вас із табличкою з вашим ім'ям, і вони якнайшвидше доставлять вас у ваш готель. як можна.

• Проживання: У нашій пошуковій системі ви знайдете тисячі готелів, обслуговувані квартири, хостели, все з найкращою гарантованою ціною.

Інші корсні програми для телефону (смартфону, планшета) у поїздці до Парижу:

RATP - Для переміщень Парижем – офіційний додаток від транспортної компанії RATP. Тут зібрано різноманітну інформацію: розклад в реальному часі електричок, метро, автобусів, трамваїв. Місцезнаходження найближчих зупинок та велостоянок, де ще є вільні велосипеди. Програма допоможе прокласти найкращий маршрут з однієї точки в іншу. Попередить про складну ситуацію на дорогах. Тут можна побачити плани всіх ліній громадського транспорту Парижа. Потрібне підключення до інтернету!

CityMapper - Тут Ви знайдете інформацію про всі види транспорту Парижа та Паризького регіону (автобуси, метро, RER, трамваї, поїзди) і навіть за розташуванням точок, де можна взяти напрокат велосипеди та автомобілі. Попереджає про проблеми на дорогах. Показує виходи з метро,

радить у якийсь вагон сісти. Розклад транспорту як реального часу. Працює не лише в Парижі, а й із транспортними системами великих міст світу.

Moovit - Ще одна програма-планувальник маршрутів з актуальним розкладом усіх видів транспорту. Вважається одним із найкращих із усіх подібних додатків, ним користується понад 300 млн. осіб. Можна використовувати у 86 країнах, 2700 містах, 44 мовами.

SNCF - Додаток для тих, хто вирішив поїхати з Парижа. Це офіційне застосування французьких залізниць з розкладом поїздів по Франції та можливістю купити онлайн квиток на поїзд. Все про квитки на поїзди до Франції.

Maps.Me - Найкраще додаток із планами міст. Працює навіть без інтернету, якщо Ви заздалегідь завантажите картку потрібного міста. З цією програмою Ви не загубитеся.

Секрети Парижа - Безкоштовний додаток допоможе знайти секретні та незвичайні місця Парижа. Французькою та англійською мовами. Може працювати оф-лайн.

Аеропорти Парижа - Офіційний додаток аеропортів Парижа (Шарль де Голль та Орлі) з відомостями про рейси та практичну інформацію. Є російська мова.

Wi-Fi Finder - Допоможе знайти вай-фай у місті.

Лафуршет - Додаток для пошуку та вибору ресторану/кафе/піцерії з можливістю зарезервувати столик без телефонного дзвінка. Також тут роздають бонуси, що конвертуються в знижки, та інформують про знижки та спеціальні пропозиції самих кафе.

Google Translate - Може стати в нагоді для перекладу; працює без інтернету, якщо ви заздалегідь завантажите потрібні мовні пакети. Можливе як введення з клавіатури, так і голосове введення слів та цілих речень. Є функція "бесіда" та завантаження інформації через фотокамеру гаджета.

Повізе - Програма вкаже громадське місце, де є можливість підзарядити телефон/гаджет.

Програми музеїв. На жаль, українською мовою офіційних програм французьких музеїв немає. Але якщо ви володієте англійською або французькою, то програми можуть допомогти в плануванні візиту і дадуть багато цікавої інформації.

Додаток Лувру - Вся практична інформація для планування відвідування, план будівель, відомості про експонати.

Додаток музею д'Орсе - Інформація про музей, його експонати та представлені в музеї артистичні напрямки; практична інформація про роботу музею; гід Парижа та його пам'яток.

Додаток центру Жоржа Помпіду - Практична інформація та відомості про виставки, експонати. План відвідин.

Додаток Версальського палацу - Практична інформація, аудіогід, геолокація на карті. Працює оф-лайн.

Додаток Діснейленду - Практична інформація, геолокація на карті, план парку, купівля квитків через програму.

2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Паттерн MVP.

На початкових етапах розробки Android учні пишуть коди таким чином, що в кінцевому підсумку створюється клас MainActivity, який містить всю логіку реалізації (бізнес-логіку реального світу) програми. Такий підхід до розробки додатків призводить до того, що діяльність Android тісно пов'язана як з інтерфейсом користувача, так і з механізмом обробки даних програми. Крім того, це викликає труднощі в обслуговуванні та масштабуванні таких мобільних додатків. Щоб уникнути подібних проблем з ремонтопридатністю, читабельністю, масштабованістю та рефакторингом додатків, розробники вважають за краще визначати добре розділені шари коду. Застосовуючи шаблони архітектури програмного забезпечення, можна організувати код програми, щоб розділити проблеми.Архітектура MVP (Model — View — Presenter) є одним із найпопулярніших архітектурних шаблонів і діє при організації проекту.

MVP (Model — View — Presenter) з'являється як альтернатива традиційному шаблону архітектури MVC (Model — View — Controller). Використовуючи MVC як архітектуру програмного забезпечення, розробники стикаються з наступними труднощами:

- Більшість основної бізнес-логіки знаходиться в Controller. Під час існування програми цей файл збільшується, і стає важко підтримувати код.
- Через тісно пов'язаний інтерфейс користувача та механізми доступу до даних і рівень контролера, і шар перегляду підпадають під одну дію або фрагмент. Це викликає проблеми з внесенням змін до функцій програми.
- Проводити модульне тестування різного рівня стає складно, оскільки більшість частини, яка тестується, потребує компонентів Android SDK.

Шаблон MVP (рисунок 2.1.) долає ці проблеми MVC і забезпечує простий спосіб структурувати коди проекту. Причина широкого визнання MVP полягає в

тому, що він забезпечує модульність, можливість тестування та більш чисту та зручну базу коду. Він складається з наступних трьох компонентів:

- *Моделі*: моделі містять відображувані дані. Зазвичай ці дані витягуються з мережі або локальної бази даних. Потім дані поміщаються в невеликі прості класи, які можуть використовувати інші компоненти.
- *Перегляди*: Перегляди – це те, що відображається користувачеві. Вони також обробляють будь-яку взаємодію користувача з екраном — наприклад, слухачі кліків. Подання має відповідати лише за відображення речей і не повинно містити жодної бізнес-логіки. Таким чином, представлення, як правило, є полегшеним компонентом порівняно з контролером і зазвичай не містить багато коду. В Android відповідальність за перегляд часто лягає на *дії* та *фрагменти* .
- *Контролери*: контролери – це спосіб з'єднувати моделі та види. Контролер оновлює модель, коли щось відбувається в поданні. Контролер також оновить подання, коли модель зміниться. Дуже часто обов'язки контролера також містяться в діяльності та фрагментах.

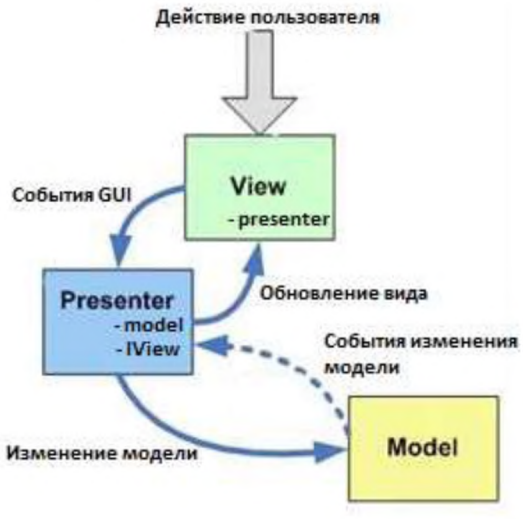


Рисунок 2.1. Модель MVP.

Ключові моменти архітектури MVP

1. Зв'язок між View-Presenter і Presenter-Model відбувається через інтерфейс (також званий контрактом) .

2. Один клас Presenter керує одним представленням за раз, тобто між доповідачем і представленням існує взаємозв'язок один до одного.

3. Класи Model і View не знають про існування один одного.

Переваги архітектури MVP

- Немає концептуальних зв'язків у компонентах Android
- Легке обслуговування та тестування коду, оскільки рівень моделі, представлення та презентатора програми розділені.

Недоліки архітектури MVP

- Якщо розробник не дотримується принципу єдиної відповідальності, щоб зламати код, тоді рівень Presenter має тенденцію розширюватися до величезного всезнаючого класу.

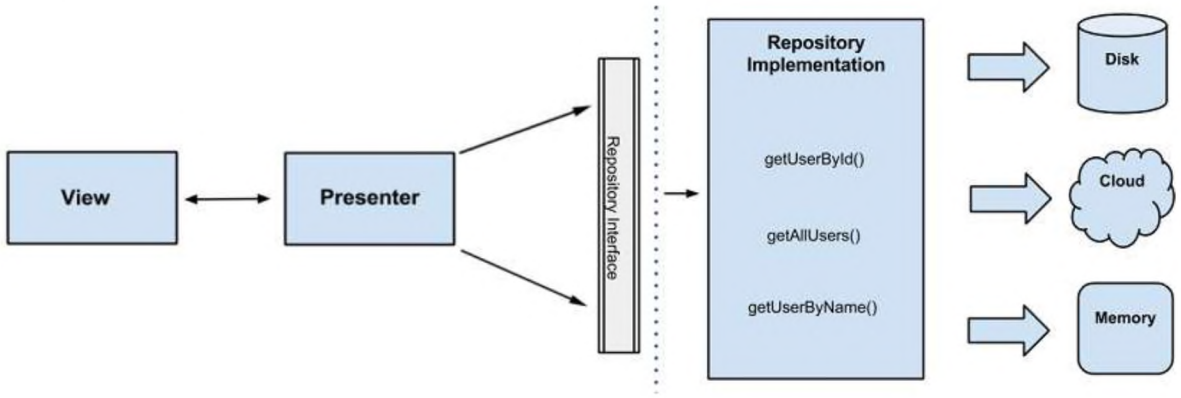
Коли справа доходить до впровадження MVC на платформі Android, все стає складніше. Для початку велика частина логіки потрапляє в контролер. Поширеною проблемою Android є активність контролера, яка містить всю логіку. Тоді контролер несе всю відповідальність за те, що відображається на екрані. Для простого екрана цим можна керувати, але в міру додавання функцій цей файл буде рости.

Крім того, діяльність Android тісно пов'язана як з інтерфейсом користувача, так і з механізмами доступу до даних. Таким чином, легко потрапити в пастку розміщення як контролера, так і логіки перегляду в дії або фрагменті. Однак це створює тісно пов'язані компоненти, що ускладнює реорганізацію та зміни.

Рівень перегляду повинен займатися лише видами. Він не повинен знати про бази даних або мережеві виклики. Крім того, важко перевірити компоненти, коли вони щільно з'єднані. Ви повинні перевірити код, який існує в контролері. Більшій частині тестованого коду в кінцевому підсумку знадобиться запустити компоненти Android SDK, такі як дії, для яких потрібен повноцінний пристрій або емулятор. Базове модульне тестування не допоможе; вам потрібно буде

використовувати дорогі інструментальні модульні тести, щоб перевірити основні частини програми.

Тому підсумкова схема може виглядати так:



Риунок 2.2. Схема патерну

2.2. Процес компіляції програми для Android з кодом Java/Kotlin

Чи замислювалися ви коли-небудь, як виглядає процес компіляції вашого коду Java/Kotlin і запуску програми на пристрої Android? Що відбувається під капотом? Що ж, мета цього розділу — описати процес компіляції програми для Android.

Як відомо, програми для Android можуть бути написані мовами програмування Java та Kotlin. Отже, процес компіляції додатків для Android базується на процесі компіляції коду Java та Kotlin (окрім середовища Android). Тому розглянемо ці два процеси.

Компіляція коду Java.

Щоб код Java запрацював, потрібно виконати кілька кроків. З метою демонстрації скажімо, що у нас є файл *TestClass.java* , який ми хочемо запустити. Кроки для компіляції даного файлу будуть такими:

1. ***TestClass.java*** компілюється за допомогою **javac (компілятор Java).**
2. Javac компілює вихідний файл **Java у файл байт-коду Java** як ***TestClass.class.***

3. Файл байт-коду Java (*TestClass.class*) потрапляє у **JVM (віртуальна машина Java).**

4. JVM розуміє байт-код і перетворює його в машинний код, використовуючи **компілятор JIT (Just-In-Time)** .

5. Потім машинний код надходить у пам'ять і виконується центральним процесором комп'ютера.

- **Байт-код Java** — представлення збірки Java.

```
// Java regular code
public int addTwoNumbers(int a, int b) {
    return a + b;
}

// Java byte-code equivalent
public int addTwoNumbers(int, int);
    Code:
       0: iload_1
       1: iload_2
       2: iadd
       3: ireturn
```

- **JVM (Java Virtual Machine)** — двигун, який забезпечує середовище виконання для виконання коду Java.

- **JIT (Just-In-Time) компілятор** — тип компілятора, який виконує компіляцію під час виконання програми (компілює програму, коли користувач її відкриває).

Компіляція коду Kotlin

За визначенням, Kotlin — це статично типізована мова програмування з відкритим вихідним кодом, яка генерує код, який може працювати на JVM.

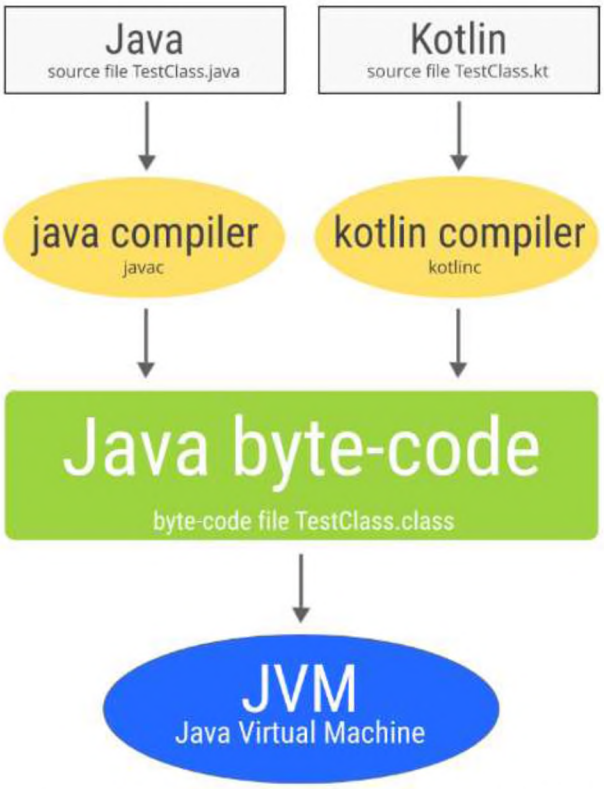


Рисунок 2.3. Процеси компіляції Java та Kotlin

Щоб мати можливість працювати на JVM, її потрібно скомпілювати в байт-код Java. Якщо ми перевіримо FAQ Kotlin , це саме те, що робить компілятор Kotlin.

Подивившись на зображення рис. ККК, ми можемо зробити висновок, що процеси компіляції Java і Kotlin майже однакові.

Процес компіляції Android

Основна відмінність між звичайною компіляцією коду Java/Kotlin і процесом компіляції Android полягає в тому, що Android не працює з JVM (якщо вам цікаво чому, ви можете знайти відповідь у кінці статті).

Замість цього Android вирішив створити дві віртуальні машини спеціально для Android:

1. **DVM (віртуальна машина Dalvik)**
2. **ART (Android Runtime) —** представлений з випуском Android 4.4 (Kitkat), а до цього середовищем виконання для додатків Android був DVM.

Що стосується їх функціональних можливостей, ART і Dalvik є сумісними середовищами виконання, які виконують **байт-код Dex,** який можна знайти у файлі **.dex** .

Тепер виникає питання, звідки ми беремо файл .dex? Виявляється, між байт-кодом Java (файлом .class) і DVM знаходиться ще один компілятор. Його назва — **DX (компілятор DEX), і його мета — перетворити байт-код Java в байт-код Dalvik.**

З огляду на це, у нас є всі частини процесу компіляції Android, і ми можемо представити його так:

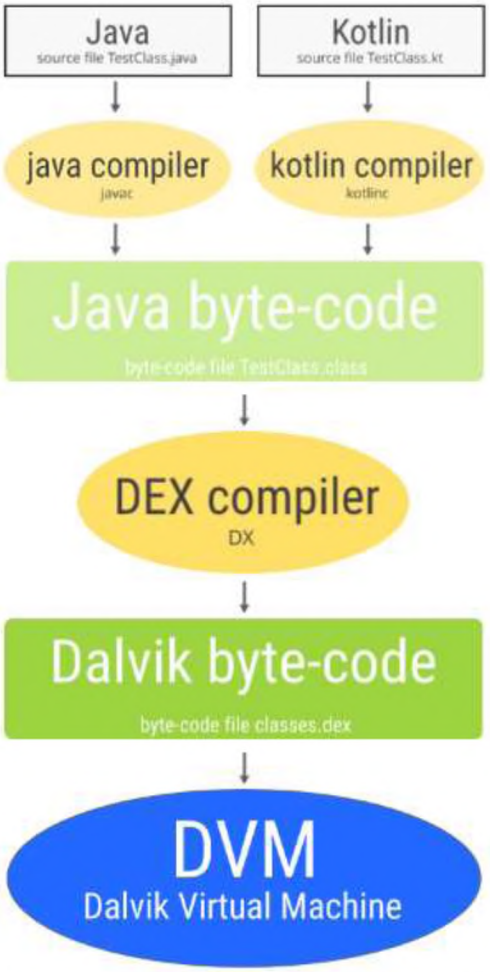


Рисунок 2.4. Процес компіляції Java та Kotlin на Android.

Також було б цікаво перевірити, як виглядає байт-код Dalvik. Ось приклад раніше згаданої функції *addTwoNumbers* як звичайного коду, її еквівалент байтового коду Java та його еквівалент байтового коду Dalvik.

```
// Java regular code
public int addTwoNumbers(int a, int b) {
    return a+ b;
}


// Java byte-code equivalent
public int addTwoNumbers(int, int);
    Code:
       0: iload_1
       1: iload_2
       2: iadd
       3: ireturn


// Dalvik byte-code equivalent
.method public addTwoNumbers(II)I
    .registers 4
    .param p1, "a"    # I
    .param p2, "b"    # I

    .line 6
    add-int v0, p1, p2

    return v0
.end method
```

3. ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Етапи розробки мобільного додатку для туризму.

Мобільні програми для туристів несуть практичну цінність. Вони є додатковим інструментом, помічником, який завжди є під рукою. Розробляючи програми для туристів, ми намагаємося зробити його максимально доступним, зрозумілим, простим та лаконічним. Щоб користувач одразу отримав потрібний результат.

Рисунок 3.1. Етапи розробки проекту

Етапи розробки:

1. Щоб створити програму для подорожей, необхідно зрозуміти основні завдання. Обговорюється ідея, формується бачення та визначається, як може використовуватися програма, яку роль виконуватиме і яка користь для клієнта та бізнесу.

2. Далі слідує аналітика. Збираються дані, що допоможуть сформувати концепцію. Вивчаються додатки конкурентів, аналізується ринок, потреби клієнтів, прогнозується просування. Аналітика допомагає поставити чіткі завдання та скласти план робіт.

3. Фахівці створюють прототип, готується первинний дизайн, визначаються інструменти реалізації програми, який стек технологій буде використаний для створення програми для подорожей. Прототип допомагає зрозуміти, як виглядатиме програма в результаті. Після його затвердження на роботу приступають технічні фахівці.

4. Створюється дизайн, всі елементи промальовуються, а програмісти пишуть код. Це займає значну частину часу розробки. Усі модулі затверджуються. Проводиться тестування та програма збирається в робочу систему.

5. Запуск. Фахівці навчають співробітників клієнта використовувати програму для туризму, розміщують на майданчиках і допомагають запустити. Рробота вважається закінченою, коли програма повноцінно працює.

Мобільний додаток для туризму це інструмент взаємодії з клієнтом, що підвищує довіру, впізнаваність бренду та покращує комунікацію з туристом. Для співробітників додаток дає можливість оптимізувати та прискорити роботу, відстежувати якість обслуговування, збирати статистику та звіти. Це допомагає покращити діяльність компанії та вивести бізнес на новий рівень.

Функціонал мобільного додатку для туризму залежить від поставлених завдань. Можна впровадити будь-які інструменти, починаючи від перекладача та конвертера валют, закінчуючи інструментами бронювання готелів, переглядом статистики та відгуків та підключенням карток з відображенням локацій. Ми можемо впровадити будь-які інструменти, необхідні для бізнесу.

Вартість мобільного додатку для туризму залежить від низки факторів, серед яких: дизайн, кількість інструментів, що впроваджуються, можливості навантаження, кількість адміністраторів, наявність особистих кабінетів, підключення сторонніх сервісів. Чим складніше додаток, тим дорожче воно обходиться, але тоді й користі від нього більше. Детальну ціну ми зможемо сказати лише після попередньої консультації та обговорення проекту.

3.2. Створення Android додатку

Android - операційна система для смартфонів, планшетів, електронних книг, цифрових програвачів, наручного годинника, фітнесбраслетів, ігрових приставок, ноутбуків, окулярів Google Glass, телевізорів та інших пристроїв [1]. Вона заснована на ядрі Linux та власної реалізації віртуальної машини Java від Google. Спочатку розроблялася компанією Android Inc. (тепер Google).

Кожна версія системи, починаючи з версії 1.5, отримує власне кодове ім'я на тему солодощів. Кодові імена надаються у порядку латинського алфавіту

крок – встановлення версії API (рис. 1.2). При виборі нового API надається

- ***Розробка інтерфейсу програми***

У студії має бути відкрито файл activity main.xml, який містить визначення графічного інтерфейсу програми.

Якщо файл відкритий у режимі дизайнера, а центрі відображається дизайн програми, треба переключити вигляд файлу в текстовий. Для перемикання режиму з текстового в графічний і назад є дві кнопки Design і Text (рис. 3.2.).

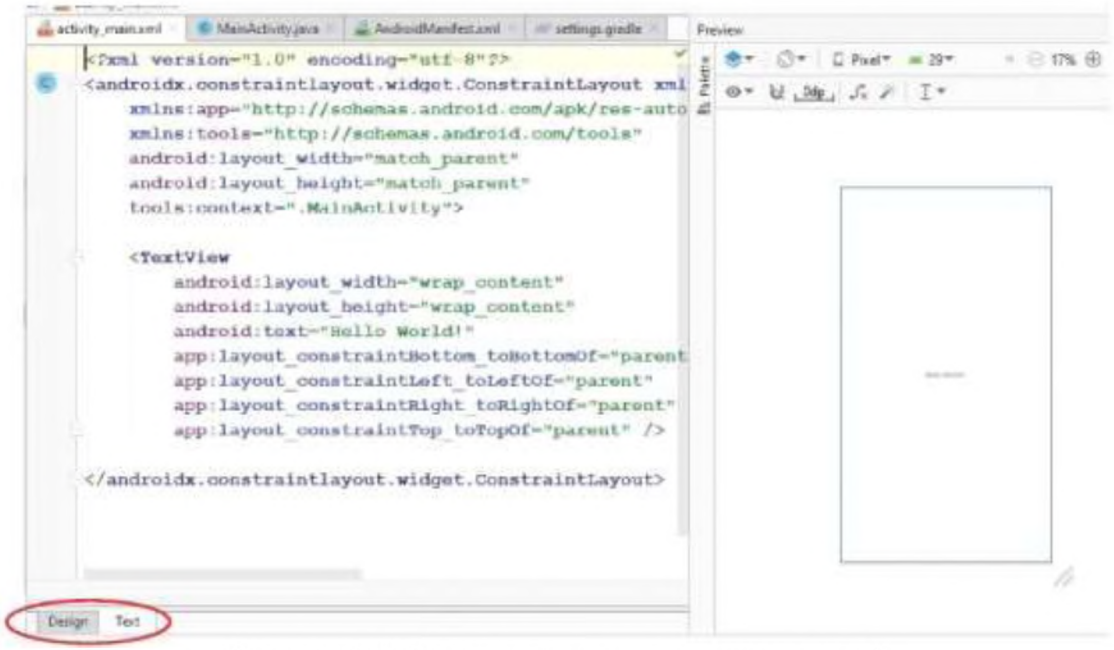


Рисунок 3.2. Режими редагування у середовищі.

Для реалізації програми створено наступні класи:

- Клас **HomeFragment.** Відповідає за відображення домашнього екрану.
- Клас **MainActivity.** Відповідає за відображення вмісту програми та реалізації деякої бізнес-логіки.
- Клас **Alarm Receiver.** Відповідає за появлення та відображення пуш-сповіщень.
- Клас **HotelActivity**. Відповідає за відображення екрану детального представлення для бронювання житла.
- Клас **NoInternetDialog**. Клас, який описує діалог, що відображається користувачу при відсутності доступу до інтернету.

- Клас **SplashActivity**. Описує вікно, яке відображається в момент запуску аплікації.
- Клас **SharedPrefsManager**. Відповідає за збереження даних програми та відновлення їх при повторному використанні аплікації.
- Клас **WebViewActivity**. Відповідає за відображенян вмісту допоміжних веб-сторінок аплікації.

Напеведемо лістинг клас **HomeFragment** та методів які відповідають за відображення домашнього екрану.

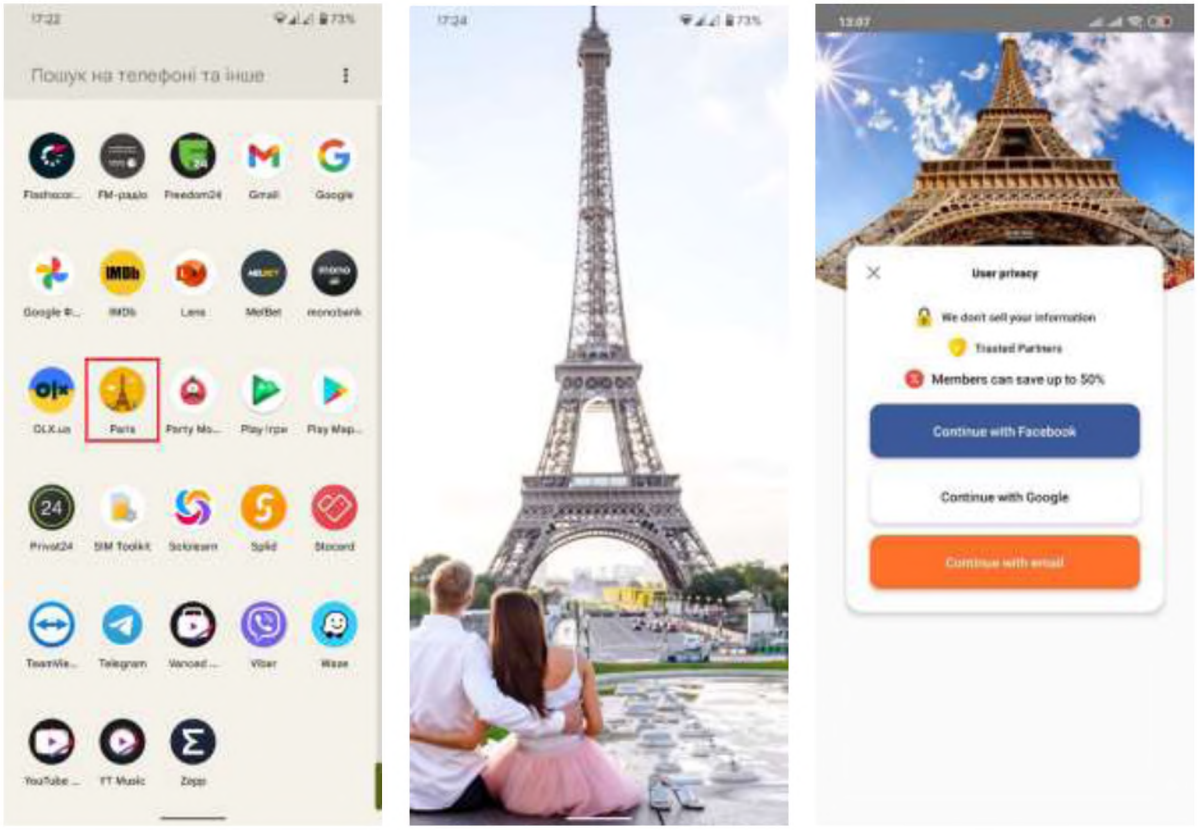
```
class HomeFragment : Fragment() {

    private val itemsList: List<TourModel> = listOf(
        TourModel(
            "Paris",
            R.drawable.img_1,
            "https://www.getyourguide.com/paris-l16/?partner_id=VGTGP5R&cmp=Paris"),
...
    TourModel(
        "Weather",
        R.drawable.img_4,
        "https://www.google.com/search?client=android&q=paris+weather"
    ),
    TourModel(
        "City Cards",
        R.drawable.img_14,
        "https://www.getyourguide.com/paris-l16/city-cards-tc101/?partner_id=VGTGP5R&cmp=ParisCityCards"
    )
)
override fun onCreateView(
    inflater: LayoutInflater,
    container: ViewGroup?,
    savedInstanceState: Bundle?
): View? {

    return inflater.inflate(R.layout.fragment_home, container, false)
}

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)

    list.adapter = TourAdapter(itemsList)

    btn_car_rent.setOnClickListener {
        startActivity(Intent(context, WebViewActivity::class.java).apply {
            putExtra(
                WebViewActivity.LINK_INTENT,
            )
        })
    }

    btn_hotels.setOnClickListener {
        startActivity(Intent(context, HotelActivity::class.java))
    }

    btn_emergency.setOnClickListener {
        startActivity(Intent(context, HelpActivity::class.java))
    }
}
override fun onResume() {
    super.onResume()
    if ((activity as? AppCompatActivity)?.verifyAvailableNetwork() == false) {
        val inetDialog =
            NoInternetDialog()
        inetDialog.show(childFragmentManager, "no_internet_dialog")
    }
```

```
    }
    private fun openWebViewByLink(link: String) {
        val intent = Intent(
            Intent.ACTION_VIEW,
            Uri.parse(link)
        )
        startActivity(intent)
    }
}
```

Лістинг класів та їх методів представлено в додатках А-В.

3.3. Функціонал розробленого додатку

Щоб почати користуватися програмою слід запустити її. Для запуску використовуємо іконку застосунку із меню програм. *Рисунок 3.3. а*. Після кліку на іконку зявиться Splash-вікно із логотипом програми *Рисунок 3.3.б* і після цього користувачу відкриється екран програми де потрібно авторизуватися. *Рисунок 3.3.в.* Авторизуватися пропонується трьомаваріантами: за допомогою облікового запису Facebook, Google або за допомогою e-email.



а. *б.* *в.*

Рисунок 3.2.. Запуск програми

а. Іконка серед програм на телефоні. *б.* Вікно запуску з логотипом.

в. Вибір методу авторизації

Після запуску аплікації користувачу відображається головний екран із переліком сервісів та екскурсій, які можна замовити. У програмі передбачена категоризація сервісів – Проживання, Оренда автомобілів, Харчування та ін. Можна обрати атракціони, музеї, туристичні тури, транспортні послуги, харчування, ерегляд погоди і багато іншого.

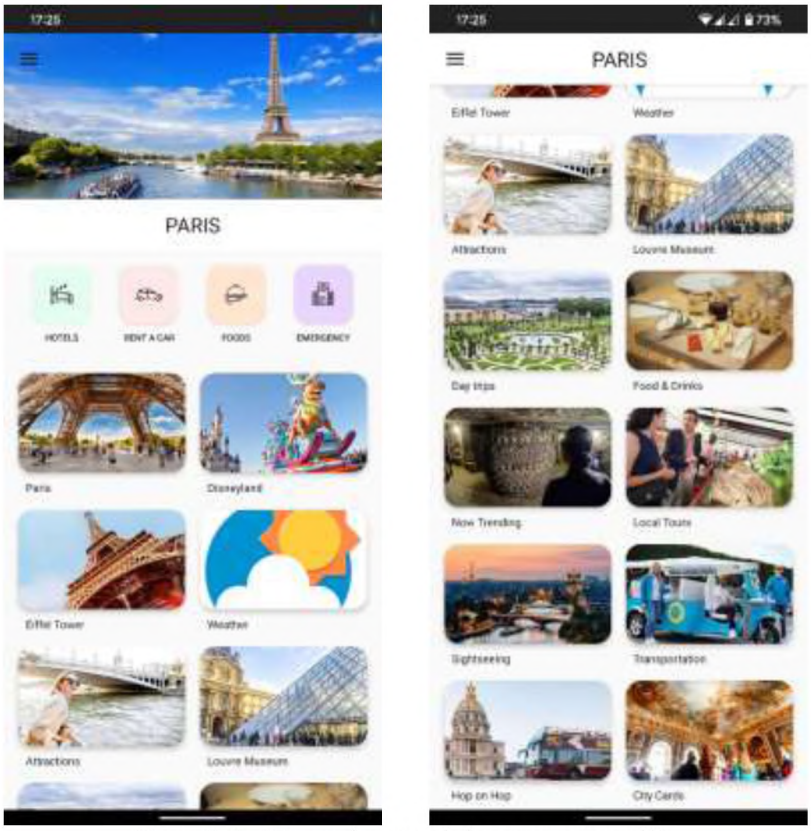


Рис. 3.4. Перелік сервісів та екскурсій.

Меню програми представлене шторкою, що випливає потягнувши екраш зліва. Меню складається із таких пунктів:

Home – Головний екран застосунку.

Hotels – Готелі – перехід у розділ з вибору та готелів.

About us – Про застосунок.

FAQ – Допомога, або часто задавані питання.

Contact us – Контакти.

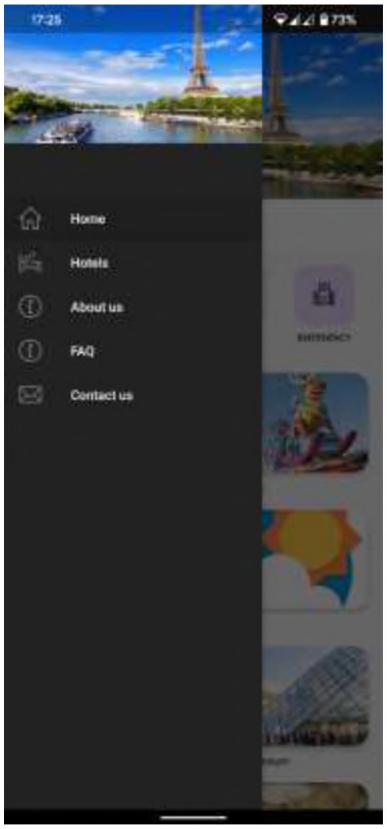


Рисунок 3.5. Меню.

Обирати тури для відвідування користувач може із відповідного списку. Вигляд турів подається по перенаправленню на відповідні екрани додатку з детальним представленням турів. Кожен екран на якому подається інформація про певний тур відображає його особливості та характеристики, до який відносять (Рис.3.6.-3.7.):

- Назву, Опис;
- Тривалість;
- Вартість;
- Наявність синхронного перекладу;
- Наявність ліфта чи іншої інфраструктури;
- Оцінка користувачів;

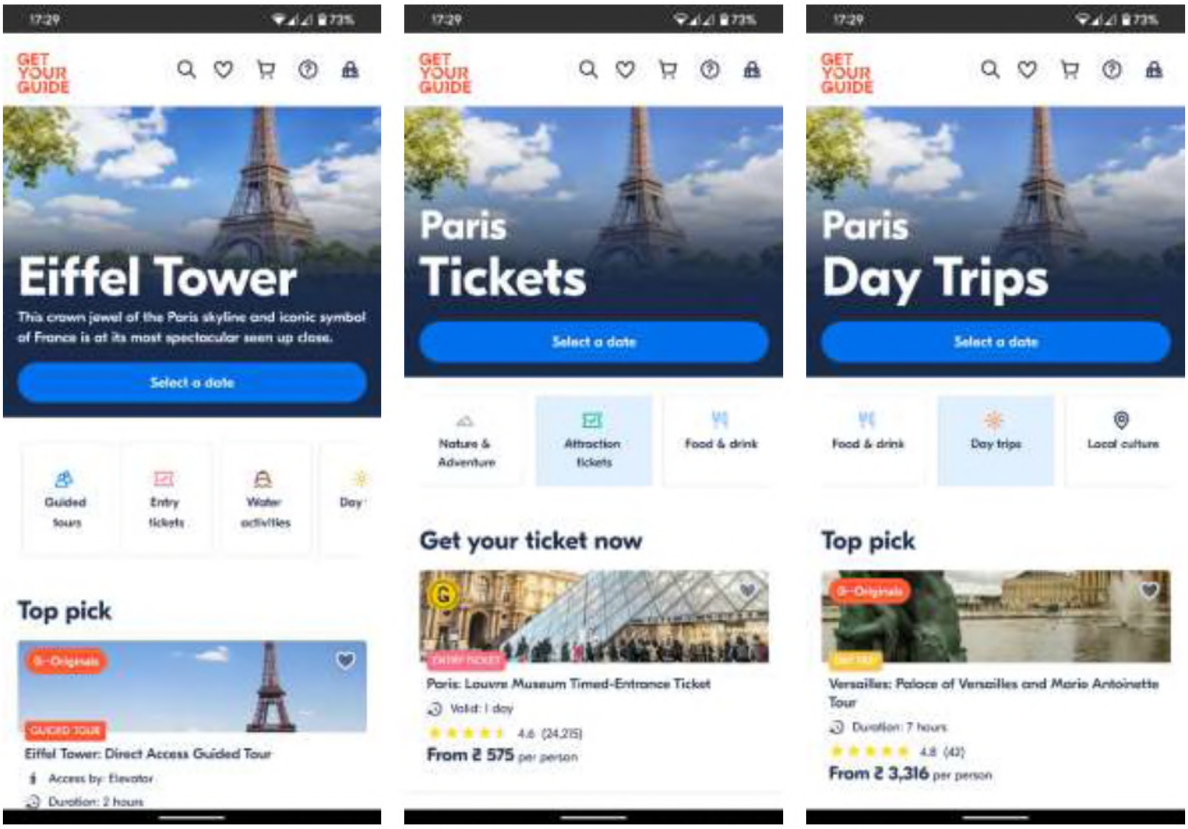


Рисунок 3.6.. Представлення турів в додатку.

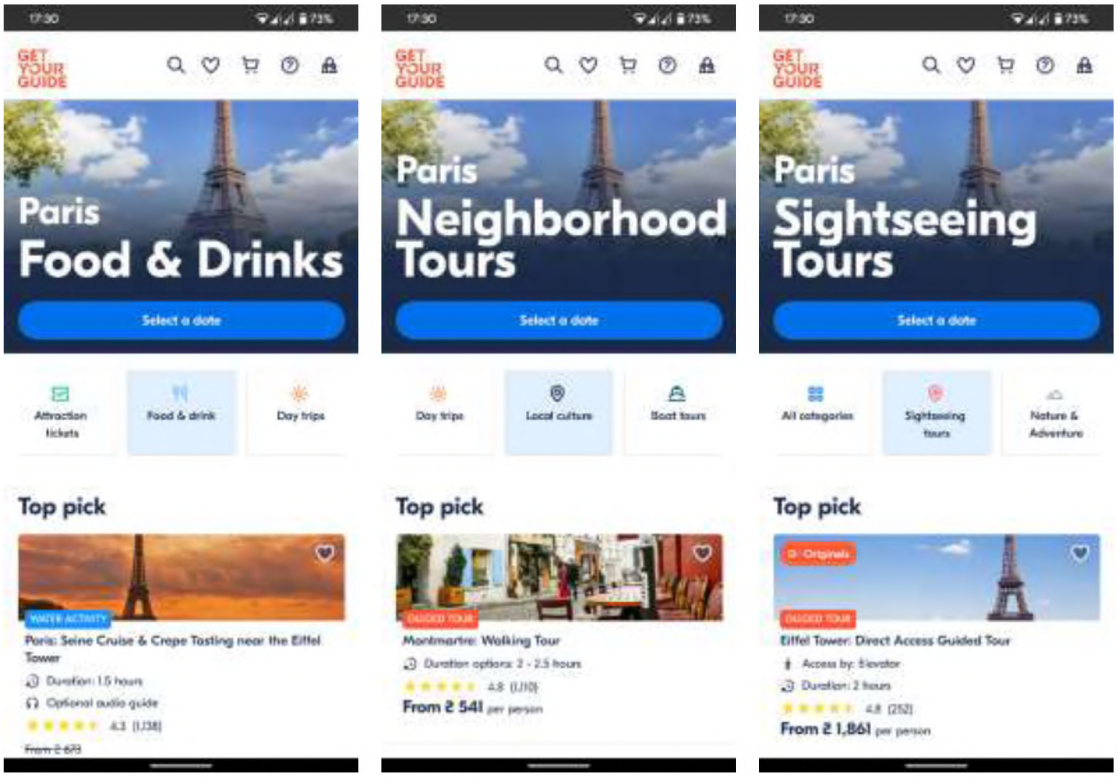
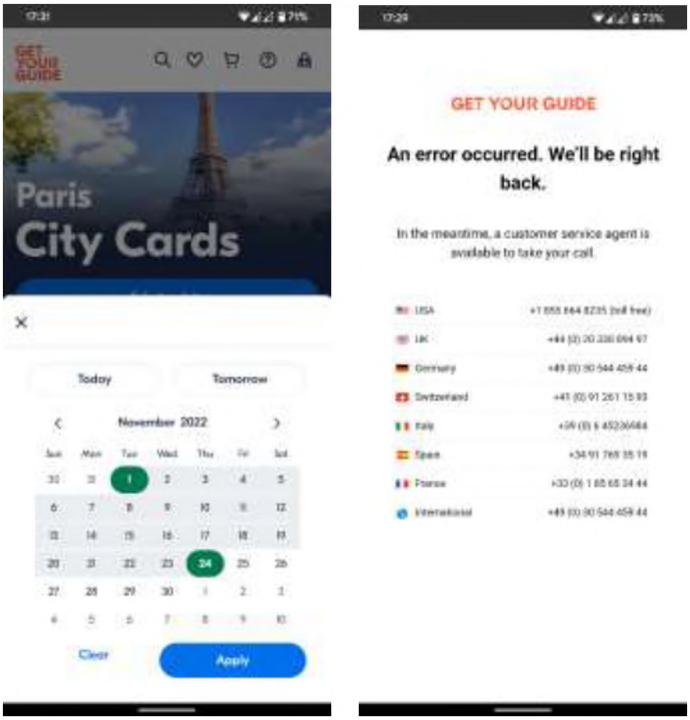


Рисунок 3.7. Представлення турів в додатку.

Після того як турист - користувач додатку обрав тур йому пропонується вказати дату бронювання туру (Рис. 3.8.). у випадку відсутності або тимчасової

несправності вибраного туру відображається Еран, що представлений на рисунку 3.8.



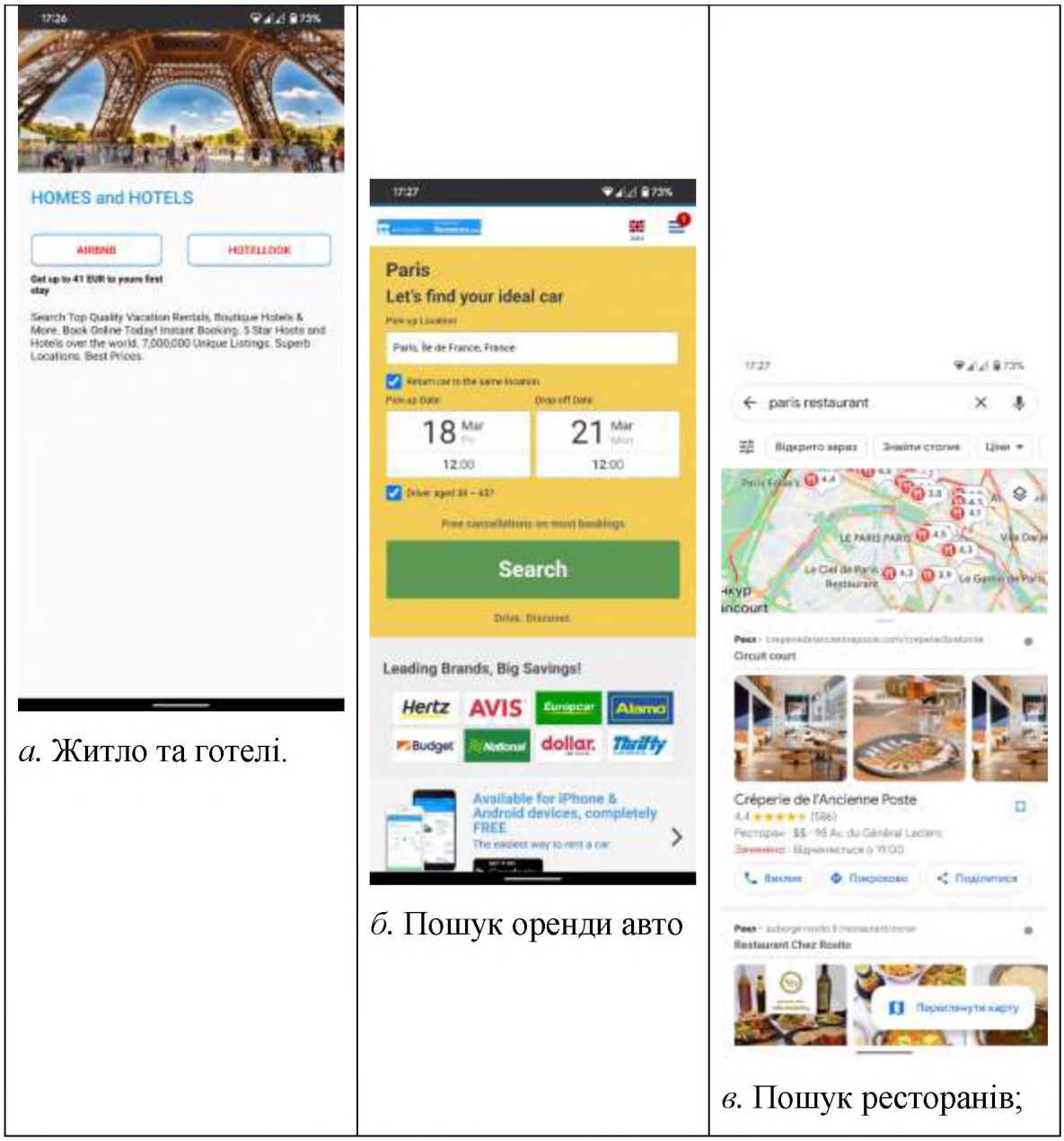
а. Вибір часу. *б.* Вікно з помилкою.

Рисунок 3.8. Вікна застосунку.

Крім цього функціонал додатку розширений послугами (рис. 3.10.) компанії Google:

Це такі сервіси:

- Житло та готелі;
- Пошук оренди авто;
- Пошук ресторанів;
- Погода;
- Служби екстерної допомоги.



а. Житло та готелі.

б. Пошук оренди авто

в. Пошук ресторанів;

Рисунок. 3.9. Сервіси доступні в додатку.

г. Погода. *д.Екстерні служби*

Рис. 3.10. Сервіси доступні в додатку.

При при втраті інтернет-з'єднання на екран мобільного телефону висвітлиться повідомлення Рис. 3.11. Користувач відреагує на нього і виправить цб ситуацію.

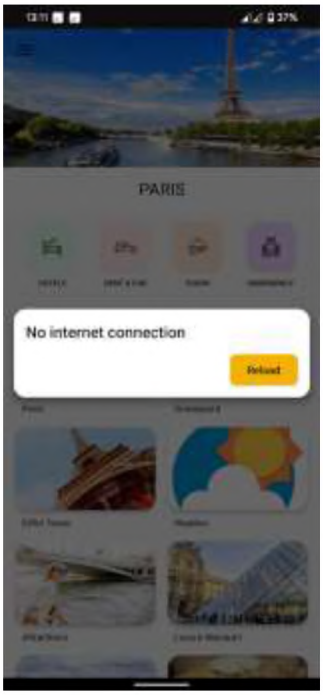


Рисунок 3.11. Діалог втрати інтернет зєднання.

Екрани із допоміжною інформацією про програму, про розробників та контакти.

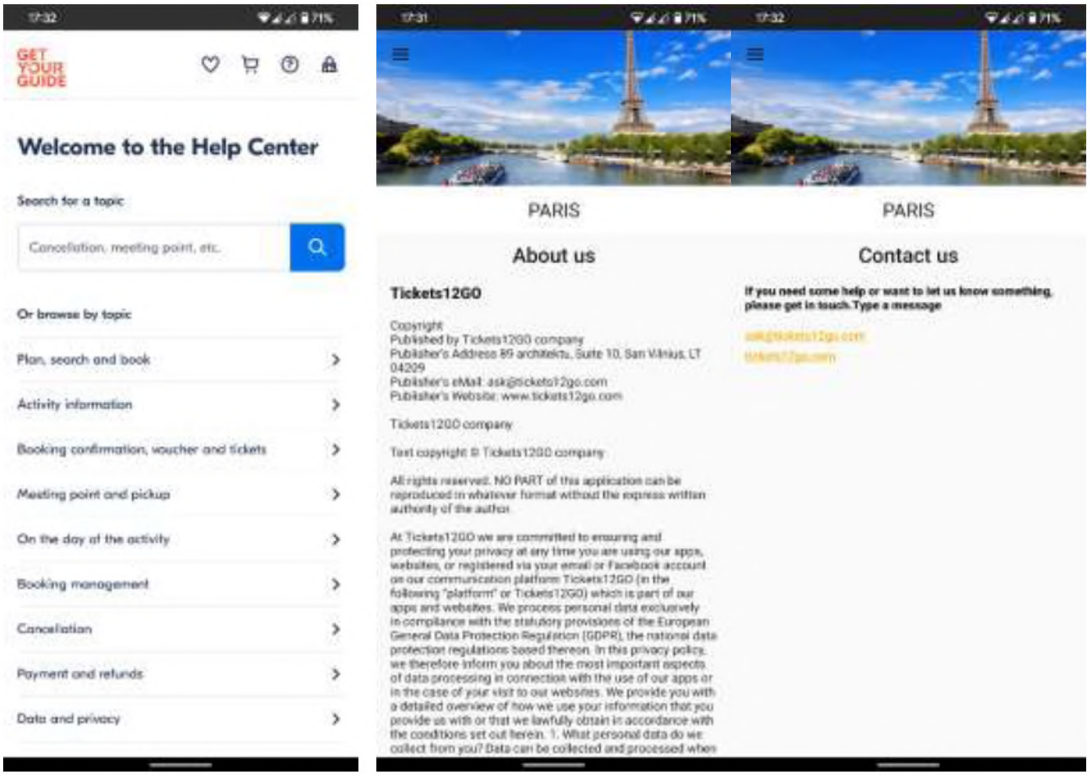


Рисунок 3.12. Службова інформація про застосунок.

Крім вище описаного функціоналу у програмі підключена автоматизована система відображення сповіщень. Сповіщення, які відображаються (рис. 3.13.) користувачеві дають можливістю скористатись пропонованою екскурсією. Це свого роду нагадування користувачеві про заходи які у нього заплановані на майбутній екскурсійний день.

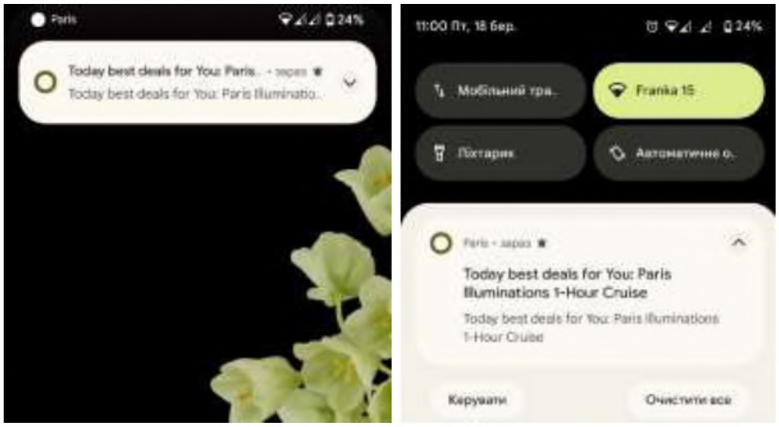


Рисунок 3.13. Сповіщення від застосунку.

Маніфіест та список каталогів програми на Рисунку 3.14.

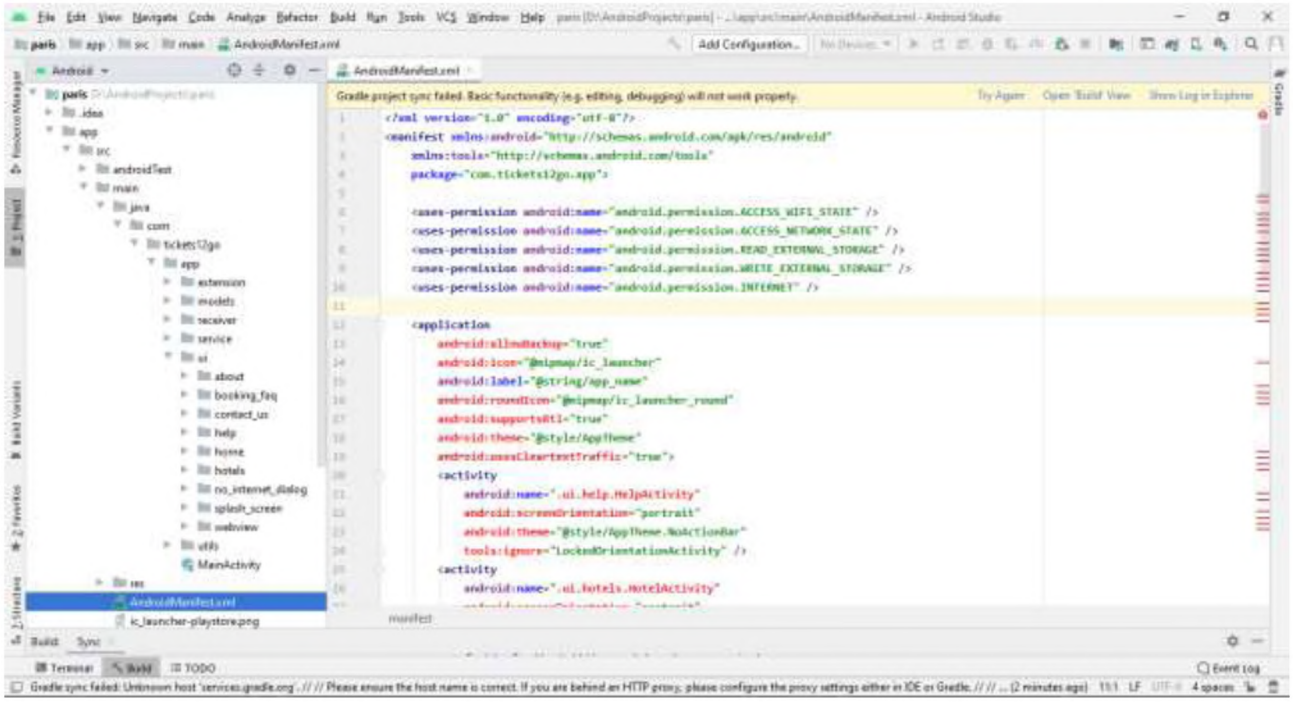


Рисунок 3.14. Екран Контактів.

ВИСНОВОК

Бурхливий розвиток технологій не тільки штовхає діловий туризм загалом до серйозних метаморфоз і змін, які потрібні щоб підлаштуватися під кінцевого клієнта, але й змінює оснащення всієї системи ділового туризму. Так з'являються в інтернеті портали для ділових подій, мобільні додатки від компаній-відправників або організацій, що займаються дозвіллям, самі люди починають користуватися електронними мобільними гаджетами, які рятують їх коли нехватає часу. IT-розвиток накриває гігантською хвилею всі сфери послуг та туризм у цьому випадку не варто бути осторонь, необхідно не просто прийняти умови, які диктує нова економіка, потрібно думати на випередження та створювати інновації.

Мобільна програма для туристів є ефективним інструментом, який допомагає просувати бізнес-послуги. Вона приносить велику користь для клієнтів і стане затребуваною. Впровадження програми для туристів у діяльність компанії допоможе залучити більше клієнтів, покращити взаємини з ними, підвищити впізнаваність бренду та розширити можливості компанії. Завдяки якісному просуванню проекту в інтернет-середовищі розробленні маркетингового плану, збільшиться кількість нових клієнтів. А відповідно, все це негайно спричинить підвищення прибутку підприємства замовника.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Android Cookbook: Problems and Solutions for Android Developers by Ian F. Darwin / Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472. 2017.
2. Дейтел, П. Android для разработчиков / П. Дейтел, Х. Дейтел, А. Уолд. – 3-е изд. – СПб.: Питер, 2016. – 512 с.
3. Android. Программирование для профессионалов / Б. Харди [и др.]. – 2-е изд. – СПб.: Питер, 2016. – 640 с.
4. Гриффитс, Д. Head First. Программирование для Android / Д. Гриффитс. – СПб.: Питер, 2016. – 704 с.
5. Start Android: учебник по Android для начинающих и продвину тых // STARTANDROID [Электронный ресурс]. – Режим доступа: https://startandroid.ru/ru/. – Дата доступа: 01.01.2020.
6. Уроки по разработке андроид-приложений // Fandroid.info [Электронный ресурс]. – Режим доступа: https://www.fandroid.info. – Дата доступа: 11.12.2019.
7. Murphy, Mark L. The Busy Coder's Guide to Android Develop ment 8.11 / Mark L. Murphy. – CommonsWare, 2018. – 417 с.
8. Darwin, Ian F. Android Cookbook: Problems and Solutions for Android Developers / Ian F. Drawin. –2nd Edition. – Sebastopol: O'Reilly, 2017.– 838 с.
9. Майер, Р. Android 4. Программирование приложений для планшетных компьютеров и смартфонов / Р. Майер. – М.: Эксмо, 2013. – 814 с.
10. Professional Android 4th Edition, Kindle Edition by Reto Meier, Ian Lake
11. Features in SQL Server Management Studio. Режим доступа: https://docs.microsoft.com/en-us/sql/ssms/features-in-sql-server-management-studio
12. Clean Code: A Handbook of Agile Software Craftsmanship. By Robert C. Martin. 2017 O'Reilly Media, Inc. All rights reserved.

13. Head First Kotlin: A Brain-Friendly Guide - February 13, 2019 - Dawn Griffiths.
14. Kotlin Programming: The Big Nerd Ranch Guide by Josh Skeen, David Greenhalgh.
15. Model-View-Controller. Режим доступа: https://msdn.microsoft.com/ en-us/library/ff649643.aspx (дата обращения: 18.02.2017)
16. RSS: A View to the Inside by Walter K. Andersen, Shridhar D. Damle.
17. [Електронний ресурс]. – Доступний з: https://developer.android.com/
18. [Електронний ресурс]. – Доступний з: https://gradle.org/guides/
19. [Електронний ресурс]. – Доступний з: https://support.dj-extensions.com/ portal/en/kb/extensions
20. [Електронний ресурс]. – Доступний з: https://docs.microsoft.com/en-us/sql/ssms/use-sql-server-management-studio
21. [Електронний ресурс]. Основы создания приложений // Developers – Режим доступа: https://developer.android.com/guide.

ДОДАТОК А

Клас MainActivity. Відповідає за відображення вмісту програми та реалізації деякої бізнес-логіки.

```
class MainActivity : AppCompatActivity() {

    private var rateTimer: CountDownTimer? = null

    private lateinit var appBarConfiguration: AppBarConfiguration
    private lateinit var sharPrefsManager: SharedPrefsManager

    private var link: String = ""

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        openWebViewIfNeed(null)

        setSupportActionBar(toolbar)

        collapse_toolbar.title = " "


        sharPrefsManager = SharedPrefsManager(this)


        val navController = findNavController(R.id.nav_host_fragment)

        appBarConfiguration = AppBarConfiguration(
            setOf(
                R.id.nav_home,
                R.id.nav_hotels,
                R.id.nav_about,
                R.id.nav_booking,
                R.id.nav_contact_us
            ), drawer_layout
        )

        setupActionBarWithNavController(navController, appBarConfiguration)
        nav_view.setupWithNavController(navController)

        toolbar.title = null
        actionBar?.title = null
        actionBar?.setDisplayShowTitleEnabled(true)

        nav_view.setNavigationItemSelectedListener {
            when (it.itemId) {
                R.id.nav_booking -> {
                    openWebViewByLink("https://www.getyourguide.com/contact/?referrer_source=site_header")
                }

                R.id.nav_hotels -> {
                    startActivity(Intent(this, HotelActivity::class.java))
                }
            }

            NavigationUI.onNavDestinationSelected(it, navController)
            drawer_layout.closeDrawer(GravityCompat.START)

            return@setNavigationItemSelectedListener true
        }

        appbar_layout.addOnOffsetChangedListener(AppBarLayout.OnOffsetChangedListener { appBarLayout,
verticalOffset ->
            if (abs(verticalOffset) - appBarLayout.totalScrollRange == 0) {
                title_toolbar.setVisibilityOption(true)
            } else {
                title_toolbar.visibility = View.INVISIBLE
            }
        })
```

```
    startRateTimer()

    val startServiceIntent = Intent(this@MainActivity, NotificationService::class.java)

    ContextCompat.startForegroundService(this@MainActivity, startServiceIntent)
}

override fun onNewIntent(intent: Intent?) {
    super.onNewIntent(intent)
    openWebViewIfNeed(intent)
}


private fun openWebViewIfNeed(intent: Intent?) {
    lifecycleScope.launchWhenResumed {
        val data = intent ?: getIntent()
        try {
            link = data!!.getStringExtra("url") ?: ""

            if (link.isNotBlank()) {
                val intentWebView = Intent(this@MainActivity, WebViewActivity::class.java)

                intentWebView.putExtra(
                    WebViewActivity.LINK_INTENT,
                    link
                )

                startActivity(intentWebView)
            }
        } catch (e: Exception) {
            e.printStackTrace()
        }
    }
}


override fun onResume() {
    super.onResume()
    if (!verifyAvailableNetwork()) {
        val inetDialog =
            NoInternetDialog()
        inetDialog.show(supportFragmentManager, "no_internet_dialog")
    }
}

override fun onSupportNavigateUp(): Boolean {
    val navController = findNavController(R.id.nav_host_fragment)
    return navController.navigateUp(appBarConfiguration) || super.onSupportNavigateUp()
}

private fun startRateTimer() {
    if (sharPrefsManager.getRateDialogStatus()) {
        return
    }

    rateTimer = object : CountDownTimer(120_000, 1000) {
        override fun onFinish() {
            showRateDialog()
        }

        override fun onTick(millisUntilFinished: Long) {}
    }.start()
}

private fun showRateDialog() {
    sharPrefsManager.saveRateDialogIsShowed(true)

    val dialogBuilder = AlertDialog.Builder(this)
    val dialogView = layoutInflater.inflate(R.layout.dialog_rate_app, null)
    dialogBuilder.setView(dialogView)

    val alertDialog = dialogBuilder.create()
    alertDialog.show()
```

```
        alertDialog.window?.setBackgroundDrawable(ColorDrawable(Color.TRANSPARENT))

        dialogView.btn_rate_no.setOnClickListener {
            alertDialog.dismiss()
        }

        dialogView.btn_rate_yes.setOnClickListener {
            val uri = Uri.parse("market://details?id=$packageName")
            val intent = Intent(Intent.ACTION_VIEW, uri)

            intent.addFlags(
                Intent.FLAG_ACTIVITY_NO_HISTORY or
                        Intent.FLAG_ACTIVITY_NEW_DOCUMENT or
                        Intent.FLAG_ACTIVITY_MULTIPLE_TASK
            )

            try {
                startActivity(intent)
            } catch (e: ActivityNotFoundException) {
                startActivity(
                    Intent(
                        Intent.ACTION_VIEW,
                        Uri.parse("http://play.google.com/store/apps/details?id=$packageName")
                    )
                )
            }

            alertDialog.dismiss()
        }
    }

    private fun openWebViewByLink(link: String) {
        val intent = Intent(this, WebViewActivity::class.java)

        intent.putExtra(WebViewActivity.LINK_INTENT, link)

        startActivity(intent)
    }

    companion object {
        const val NOTIFY_LINK = "main_notify_link"
    }
}
```

ДОДАТОК Б

Клас Alarm Receiver. Відповідає за появлення та відображення пуш-сповіщень.

```
public class AlarmReceiver extends BroadcastReceiver {

    public static String ACTION_NOTIFY = "com.tenerife.tenerifeadvisor.notify";
    private final String CHANEL = "paris";
    private static String[][] NOTIFICATIONS;

    void initNotificationMessages() {
        String[][] r0 = new String[5][];
        r0[0] = new String[]{"https://www.getyourguide.com/paris-l16/skip-the-line-eiffel-tower-tour-with-
summit-access-t62484/?partner_id=VGTGP5R&cmp=EIFFEL", "Today best deals for You: Eiffel Tower Tickets: Summit-
Level Access "};
        r0[1] = new String[]{"https://www.getyourguide.com/paris-l16/night-cruise-through-the-illuminated-
paris-t57315/?partner_id=VGTGP5R&cmp=ILLUM", "Today best deals for You: Paris Illuminations 1-Hour Cruise"};
        r0[2] = new String[]{"https://www.getyourguide.com/paris-l16/louvre-museum-skip-the-line-entrance-
ticket-t63741/?partner_id=VGTGP5R&cmp=LOUVRE", "Today best deals for You: Louvre Museum: Skip-The-Line
Entrance Ticket"};
        r0[3] = new String[]{"https://www.getyourguide.com/paris-l16/paris-catacombs-skip-the-line-ticket-
t66985/?partner_id=VGTGP5R&cmp=CATACOMB", "Today best deals for You: Paris Catacombs Skip-the-Line Ticket"};
        r0[4] = new String[]{"https://www.getyourguide.com/paris-l16/moulin-rouge-show-with-champagne-
t54034/?partner_id=VGTGP5R&cmp=MOULIN", "Today best deals for You: Moulin Rouge Show with Champagne"};
        NOTIFICATIONS = r0;
    }


    @Override
    public void onReceive(Context context, Intent intent) {
        initNotificationMessages();
        int notifShowCount = ShPrefs.loadIntPrefs("notifCount", 0, context);
        if (notifShowCount < NOTIFICATIONS.length) {
            ShPrefs.saveIntPrefs("notifCount", notifShowCount + 1, context);
            PowerManager pm = (PowerManager) context.getSystemService(Context.POWER_SERVICE);
            PowerManager.WakeLock wl = pm.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK, "");
            wl.acquire();


            int last = context.getApplicationContext().getSharedPreferences(ACTION_NOTIFY,
Context.MODE_PRIVATE).getInt("last-notification", -1);
            if (last >= NOTIFICATIONS.length - 1) {
                last = 0;
            } else {
                last++;
            }

            Log.v("notifications", "Last: " + last);

            context.getApplicationContext().getSharedPreferences(ACTION_NOTIFY,
Context.MODE_PRIVATE).edit().putInt("last-notification", last).apply();
            Intent notificationIntent = new Intent(context, MainActivity.class);
            notificationIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_SINGLE_TOP |
Intent.FLAG_ACTIVITY_CLEAR_TOP);
            notificationIntent.putExtra("url", NOTIFICATIONS[last][0]);
            PendingIntent pendingIntent = PendingIntent.getActivity(context, 12345,
                    notificationIntent, PendingIntent.FLAG_CANCEL_CURRENT);
            NotificationCompat.Builder builder = new NotificationCompat.Builder(context, CHANEL);
            builder.setContentIntent(pendingIntent);
            builder.setContentTitle(NOTIFICATIONS[last][1]);
            builder.setContentText(NOTIFICATIONS[last][1]);
            builder.setSmallIcon(R.mipmap.ic_launcher);
            builder.setDefaults(Notification.DEFAULT_SOUND);
            builder.setAutoCancel(true);
            NotificationManager mNotificationManager = (NotificationManager)
context.getSystemService(Context.NOTIFICATION_SERVICE);
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
                CharSequence name = context.getResources().getString(R.string.app_name);// The user-visible
name of the channel.
                int importance = NotificationManager.IMPORTANCE_HIGH;
                NotificationChannel mChannel = new NotificationChannel(CHANEL, name, importance);
```

```
                mNotificationManager.createNotificationChannel(mChannel);
            }
            mNotificationManager.notify(12345, builder.build());

            wl.release();
        }

    }

    public void setAlarm(Context context) {
        Calendar alarmStartTime = Calendar.getInstance();
        Calendar now = Calendar.getInstance();
        alarmStartTime.set(Calendar.HOUR_OF_DAY, 11);
        alarmStartTime.set(Calendar.MINUTE, 0);
        alarmStartTime.set(Calendar.SECOND, 0);
        if (now.after(alarmStartTime)) {
            alarmStartTime.add(Calendar.DATE, 1);
        }
        Intent intent1 = new Intent(context, AlarmReceiver.class);
        PendingIntent pendingIntent = PendingIntent.getBroadcast(context, 12345, intent1,
PendingIntent.FLAG_UPDATE_CURRENT);
        AlarmManager am = (AlarmManager) context.getSystemService(context.ALARM_SERVICE);
        am.setRepeating(AlarmManager.RTC_WAKEUP, alarmStartTime.getTimeInMillis(), AlarmManager.INTERVAL_DAY,
pendingIntent);
    }

    public void cancelAlarm(Context context) {
        Intent intent = new Intent(context, AlarmReceiver.class);
        PendingIntent sender = PendingIntent.getBroadcast(context, 0, intent, 0);
        AlarmManager alarmManager = (AlarmManager) context.getSystemService(Context.ALARM_SERVICE);
        alarmManager.cancel(sender);
    }
}
```

ДОДАТОК В

Клас HomeFragment. Відповідає за відображення домашнього екрану.

```
class HomeFragment : Fragment() {

    private val itemsList: List<TourModel> = listOf(
        TourModel(
            "Paris",
            R.drawable.img_1,
            "https://www.getyourguide.com/paris-l16/?partner_id=VGTGP5R&cmp=Paris"
        ),
        TourModel(
            "Disneyland",
            R.drawable.img_2,

"https://www.getyourguide.com/s/?q=Disneyland%20Paris&lc=l2603/?partner_id=VGTGP5R&cmp=ParisDisneyland"
        ),
        TourModel(
            "Eiffel Tower",
            R.drawable.img_3,
            "https://www.getyourguide.com/eiffel-tower-l2600/?partner_id=VGTGP5R&cmp=EiffelTower"
        ),
        TourModel(
            "Weather",
            R.drawable.img_4,
            "https://www.google.com/search?client=android&q=paris+weather"
        ),
        TourModel(
            "Attractions",
            R.drawable.img_5,
            "https://www.getyourguide.com/paris-l16/tickets-tc123/?partner_id=VGTGP5R&cmp=ParisAttraction"
        ),
        TourModel(
            "Louvre Museum",
            R.drawable.img_6,
            "https://www.getyourguide.com/s/?q=Louvre%20Museum&lc=l3224/?partner_id=VGTGP5R&cmp=ParisLouvre"
        ),
        TourModel(
            "Day trips",
            R.drawable.img_7,
            "https://www.getyourguide.com/paris-l16/day-trips-tc172/?partner_id=VGTGP5R&cmp=Parisdaytrips"
        ),
        TourModel(
            "Food & Drinks",
            R.drawable.img_8,
            "https://www.getyourguide.com/paris-l16/food-drinks-
tc103/?partner_id=VGTGP5R&cmp=ParisFood&Drinks"
        ),
        TourModel(
            "Now Trending",
            R.drawable.img_9,
            "https://www.getyourguide.com/paris-l16/tickets-tc123/?partner_id=VGTGP5R&cmp=ParisNowTrending"
        ),
        TourModel(
            "Local Tours",
            R.drawable.img_10,
            "https://www.getyourguide.com/paris-l16/local-flavor-neighborhood-tours-
tc21/?partner_id=VGTGP5R&cmp=ParislocalTours"
        ),
        TourModel(
            "Sightseeing",
            R.drawable.img_11,
            "https://www.getyourguide.com/paris-l16/sightseeing-tours-
tc2/?partner_id=VGTGP5R&cmp=ParisSightseeing"
        ),
        TourModel(
            "Transportation",
            R.drawable.img_12,
            "https://www.getyourguide.com/paris-l16/transportation-
tc152/?partner_id=VGTGP5R&cmp=ParisTransportation"
        ),
        TourModel(
```

```
            "Hop on Hop",
            R.drawable.img_13,
            "https://www.getyourguide.com/paris-l16/hop-on-hop-off-tours-
tc9/?partner_id=VGTGP5R&cmp=ParisHoponHop"
        ),
        TourModel(
            "City Cards",
            R.drawable.img_14,
            "https://www.getyourguide.com/paris-l16/city-cards-tc101/?partner_id=VGTGP5R&cmp=ParisCityCards"
        )
    )

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {

        return inflater.inflate(R.layout.fragment_home, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        list.adapter = TourAdapter(itemsList)

        btn_foods.setOnClickListener {

openWebViewByLink("https://www.google.com/maps/search/paris+restaurant/@48.8567823,2.3159511,13z/data=!4m4!2m3
!5m1!4e9!6e5")
        }

        btn_car_rent.setOnClickListener {
            startActivity(Intent(context, WebViewActivity::class.java).apply {
                putExtra(
                    WebViewActivity.LINK_INTENT,

"https://c130.travelpayouts.com/click?shmarker=278320.paris&promo_id=4004&source_type=link&type=click"
                )
            })
        }

        btn_hotels.setOnClickListener {
            startActivity(Intent(context, HotelActivity::class.java))
        }

        btn_emergency.setOnClickListener {
            startActivity(Intent(context, HelpActivity::class.java))
        }
    }

    override fun onResume() {
        super.onResume()
        if ((activity as? AppCompatActivity)?.verifyAvailableNetwork() == false) {
            val inetDialog =
                NoInternetDialog()
            inetDialog.show(childFragmentManager, "no_internet_dialog")
        }
    }

    private fun openWebViewByLink(link: String) {
        val intent = Intent(
            Intent.ACTION_VIEW,
            Uri.parse(link)
        )
        startActivity(intent)
    }
}
```

Клас HotelActivity. Відповідає за відображення екрану детального представлення для бронювання житла.

```
class HotelActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_hotel)

        setSupportActionBar(toolbar)

        collapse_toolbar.title = " "

        supportActionBar?.setHomeAsUpIndicator(R.drawable.ic_arrow_left)
        supportActionBar?.setDisplayHomeAsUpEnabled(true)
        supportActionBar?.setDisplayShowHomeEnabled(true)
        toolbar.setNavigationOnClickListener {
            onBackPressed()
        }

        window.statusBarColor = ContextCompat.getColor(this, R.color.colorPrimary)

        btn_airbnb.setOnClickListener {
            openWebViewByLink("https://abnb.me/e/xltJ2LKtD4?suuid=ffb70be2-391f-43b9-9d85-
fafc3cd4471d&slevel=0")
        }

        btn_hotellook.setOnClickListener {

openWebViewByLink("http://search.hotellook.com/?locationId=15542&marker=278320&language=en&currency=eur")
        }
    }

    private fun openWebViewByLink(link: String) {
        val intent = Intent(this, WebViewActivity::class.java)

        intent.putExtra(WebViewActivity.LINK_INTENT, link)

        startActivity(intent)
    }
}
```

Клас NoInternetDialog. Клас, який описує діалог, що відображається користувачу при відсутності доступу до інтернету.

```
class NoInternetDialog : DialogFragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        return inflater.inflate(R.layout.dialog_no_internet_connection, container, false)
    }

    override fun onStart() {
        super.onStart()

        dialog?.window?.run {
            setBackgroundDrawable(ColorDrawable(Color.TRANSPARENT))

            setLayout(
                ViewGroup.LayoutParams.MATCH_PARENT,
                ViewGroup.LayoutParams.WRAP_CONTENT
            )
        }

        dialog?.setCancelable(true)
        dialog?.setCanceledOnTouchOutside(false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        view.btn_retry.setOnClickListener {
            if ((activity as? AppCompatActivity)?.verifyAvailableNetwork() == true) {
                dismiss()
            }
        }
    }
}
```

Клас SplashActivity. Описує вікно, яке відображається в момент запуску аплікації.

```
class SplashActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)

        window.setFlags(
            WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS,
            WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS
        )

        lifecycleScope.launchWhenStarted {
            progress_bar.setVisibilityOption(true)
            delay(2000)

            val intent = Intent(this@SplashActivity, MainActivity::class.java)

            intent.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
            startActivity(intent)
            overridePendingTransition(0, 0)
        }

    }
}
```

Клас SharedPrefsManager. Відповідає за збереження даних програми та відновлення їх при повторному використанні аплікації.

```
class SharedPrefsManager(context: Context) {
    companion object {
        const val SP_NAME = "paris_app_prefs"

        const val RATE_DIALOG_SHOWED = "rate_dialog_is_showed"
    }

    private val sharedPrefs: SharedPreferences =
        context.getSharedPreferences(SP_NAME, Context.MODE_PRIVATE)

    fun saveRateDialogIsShowed(isShowed: Boolean) {
        sharedPrefs.edit {
            putBoolean(RATE_DIALOG_SHOWED, isShowed)
        }
    }

    fun getRateDialogStatus(): Boolean = sharedPrefs.getBoolean(RATE_DIALOG_SHOWED, false)

    fun clearAll() {
        sharedPrefs.edit().clear().apply()
    }
}
```

Клас WebViewActivity. Відповідає за відображенян вмісту допоміжних веб-сторінок аплікації.

```
class WebViewActivity : AppCompatActivity() {

    companion object {
        const val LINK_INTENT = "link_intent"
    }

    private var dialogShowed = false

    @SuppressLint("SetJavaScriptEnabled")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_web_view)

        window.statusBarColor = ContextCompat.getColor(this, R.color.colorPrimary)

        val link = intent.extras?.getString(LINK_INTENT, null)

        if (link == null) {
            finish()
        }

        webview.run {
            settings.javaScriptEnabled = true
            settings.javaScriptCanOpenWindowsAutomatically = true
            settings.allowContentAccess = true
            settings.domStorageEnabled = true
            settings.allowFileAccess = false
            settings.setAppCacheEnabled(false)
            settings.setAppCacheMaxSize(0)

            settings.userAgentString =
                "Mozilla/5.0 (Linux; Android 4.4; Nexus 4 Build/KRT16H) AppleWebKit/537.36 (KHTML, like Gecko)
Version/4.0 Chrome/30.0.0.0 Mobile Safari/537.36"

            scrollBarStyle = WebView.SCROLLBARS_OUTSIDE_OVERLAY
            webChromeClient = object : WebChromeClient() {
                override fun onProgressChanged(view: WebView?, newProgress: Int) {
                    progress_bar.progress = newProgress
                }
            }
            webViewClient = SSLTolerentWebViewClient()

            loadUrl(link)
        }

        swipe_refresh_layout.setOnRefreshListener {
            webview.reload()
        }

    }

    override fun applyOverrideConfiguration(overrideConfiguration: Configuration) {
        if (Build.VERSION.SDK_INT in 21..25 && (resources.configuration.uiMode ==
resources.configuration.uiMode)) {
            return
        }
        super.applyOverrideConfiguration(overrideConfiguration)
    }

    private inner class SSLTolerentWebViewClient : WebViewClient() {
        override fun onReceivedSslError(
            view: WebView,
            handler: SslErrorHandler,
            error: SslError
        ) {
            if (!dialogShowed) {
                val builder = AlertDialog.Builder(this@WebViewActivity)
                builder.setMessage("SSL certificate invalid")
                builder.setPositiveButton("Continue") { _, _ ->
```

```
                handler.proceed()
            }

            builder.setNegativeButton("Cancel") { _, _ ->
                handler.cancel()
                finish()
            }

            val dialog = builder.create()
            dialog.show()
            dialogShowed = true
        }
    }

    override fun onPageStarted(view: WebView?, url: String?, favicon: Bitmap?) {
        super.onPageStarted(view, url, favicon)

        progress_bar.setVisibilityOption(true)
    }

    override fun onPageFinished(view: WebView?, url: String?) {
        super.onPageFinished(view, url)

        progress_bar.setVisibilityOption(false)
        swipe_refresh_layout.isRefreshing = false
    }
}

override fun onResume() {
    super.onResume()
    if (!verifyAvailableNetwork()) {
        finish()
    }
}

override fun onKeyDown(keyCode: Int, event: KeyEvent): Boolean {
    if (event.action == KeyEvent.ACTION_DOWN) {
        when (keyCode) {
            KeyEvent.KEYCODE_BACK -> {
                if (webview.canGoBack()) {
                    webview.goBack()
                } else {
                    finish()
                }

                return true
            }
        }
    }

    return super.onKeyDown(keyCode, event)
}
}
```