ЛИСОВСКИЙ МАРТИН ВИКТОРОВИЧ

Разработка android-приложения для поиска и хранения авиабилетов

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

API - Специальный интерфейс программы или приложения с помощью которого одна программа может взаимодействовать с другой.

АККАУНТ - Учетная запись пользователя.

IATA-код - Индивидуальные идентификаторы объектов, имеющих значение для индустрии пассажирских авиаперевозок и присваиваемые международной ассоциацией воздушного транспорта.

СКРИНШОТ – Изображение, полученное устройством и показывающее в точности то, что видит пользователь на экране монитора или другого визуального устройства вывода

ВВЕДЕНИЕ

Актуальность проблемы

На данный момент пользователи воздушного транспорта для поиска самых дешевых авиабилетов вынуждены самостоятельно сравнивать цены у перевозчиков и агрегаторов дешевых билетов. При этом они сталкиваются с проблемами: не самая дешевая цена, наценки перевозчиков, устаревшая и недостоверная информация, а также заспамленность электронной почты. При этом, чтобы не потерять актуальность и информацию об интересующих авиабилетах необходимо устанавливать «тяжелые» приложения.

Разрабатываемое приложение призвано решить все эти проблемы, а именно: избавить пользователя от необходимости анализировать цены у разных перевозчиков, авиакомпаний и агрегаторов, предоставить ему наиболее дешевый авиабилет без пересадок, а также возможность сохранять авиабилеты, отслеживать в фоновом режиме их стоимость и получать уведомления при достижении указанной цены.

Постановка задачи и предлагаемые методы решения

Требуется найти самый дешевый билет без пересадок по заданным пользователем параметрам. По желанию пользователя сохранить билет с желаемой ценой и обновлять в фоновом режиме актуальную цену, уведомляя при достижении указанной пользователем ценны.

Для функционирования приложения потребуется зарегистрироваться, указав электронную почту, пароль, имя и телефон. Для поиска авиабилета необходимо задать следующую информацию: город отправления, город назначения, дата вылета, желаемая пользователем цена.

Приложение будем разрабатывать на Java. При разработке разобьем его на следующие модули:

1. API агрегатора.
2. Модуль обращения к API.
3. Базы данных.
4. Пользовательский интерфейс.

2. РАЗРАБОТКА КОМПОНЕНТОВ ПРИЛОЖЕНИЯ

2.1 API агрегатора

Для использования в разрабатываемом приложении мною среди многочисленных API, позволяющих получать информацию об авиабилетах, был выбран Aviasales API доступа к данным. Данный выбор был основан на основании наиболее точных и актуальных цен на авиабилеты, отсутствии платы за использование и отсутствии ограничения в количестве запросов с одного устройства.

2.2 Модуль обращения к API

С целью оптимизации скорости работы приложения и повышения комфорта использования используем поиск со всплывающим списком возможных городов, получаемых в формате JSON от API доступа к данным (рис. 2.2.1).


```
        "type": "city",
        "state_code": null,
        "country_name": "Беларусь",
        "cases": {
            "pr": "Минске",
            "da": "Минску",
            "ro": "Минска",
            "tv": "Минском",
            "vi": "в Минск"
        },
        "weight": 1911991,
        "country_code": "BY",
        "name": "Минск",
        "main_airport_name": "Минск Национальный аэропорт",
        "code": "MSQ",
        "country_cases": null
    }
]
```


Рис. 2.2.1 – Фрагмент JSON-файла, содержащего возможные города

Данный JSON-файл получаем в ответе на следующий запрос к API:
https://places.aviasales.ru/v2/places.json?max=7&term=Минск&types[]=city&locale=ru,
где max – параметр, обозначающий число выводимых возможных городов;
term – параметр, обозначающий вводимый текст пользователем;
types – параметр, обозначающий тип объектов поиска (города/аэропорты);
locale – параметр, обозначающий язык ввода текста пользователем.

Данные об авиабилете получаем, выполняя десериализацию получаемых в формате JSON данных от API. (рис. 2.2.2).


```
{
    "prices": [
        {
            "value": 2963.0,
            "return_date": null,
            "number_of_changes": 0,
            "gate": "Tickets.ru",
            "depart_date": "2020-02-18"
        }
    ],
    "errors": {}
}
```


Рис. 2.2.2 – Фрагмент JSON-файл с найденным авиабилетом

Данный JSON-файл получаем в ответе на следующий запрос к API:
https://lyssa.aviasales.ru/price_matrix?origin_iata=MSQ&destination_iata=MOW&depart_start=2020-02-18&depart_range=0&affiliate=false,
где origin_iata - параметр, обозначающий IATA-код города отправления;
destination_iata – параметр, обозначающий IATA-код города назначения;
depart_start - параметр, обозначающий дату отправления;

depart_range – параметр, обозначающий число дней от даты отправления;

affiliate – параметр, обозначающий использование партнерского поиска;

Исходный код данного модуля находится в приложении А.

2.3 Пользовательский интерфейс

Для создания пользовательского интерфейса используем XML.Приложение состоит из 3 основных экранов (рис. 2.3.1).

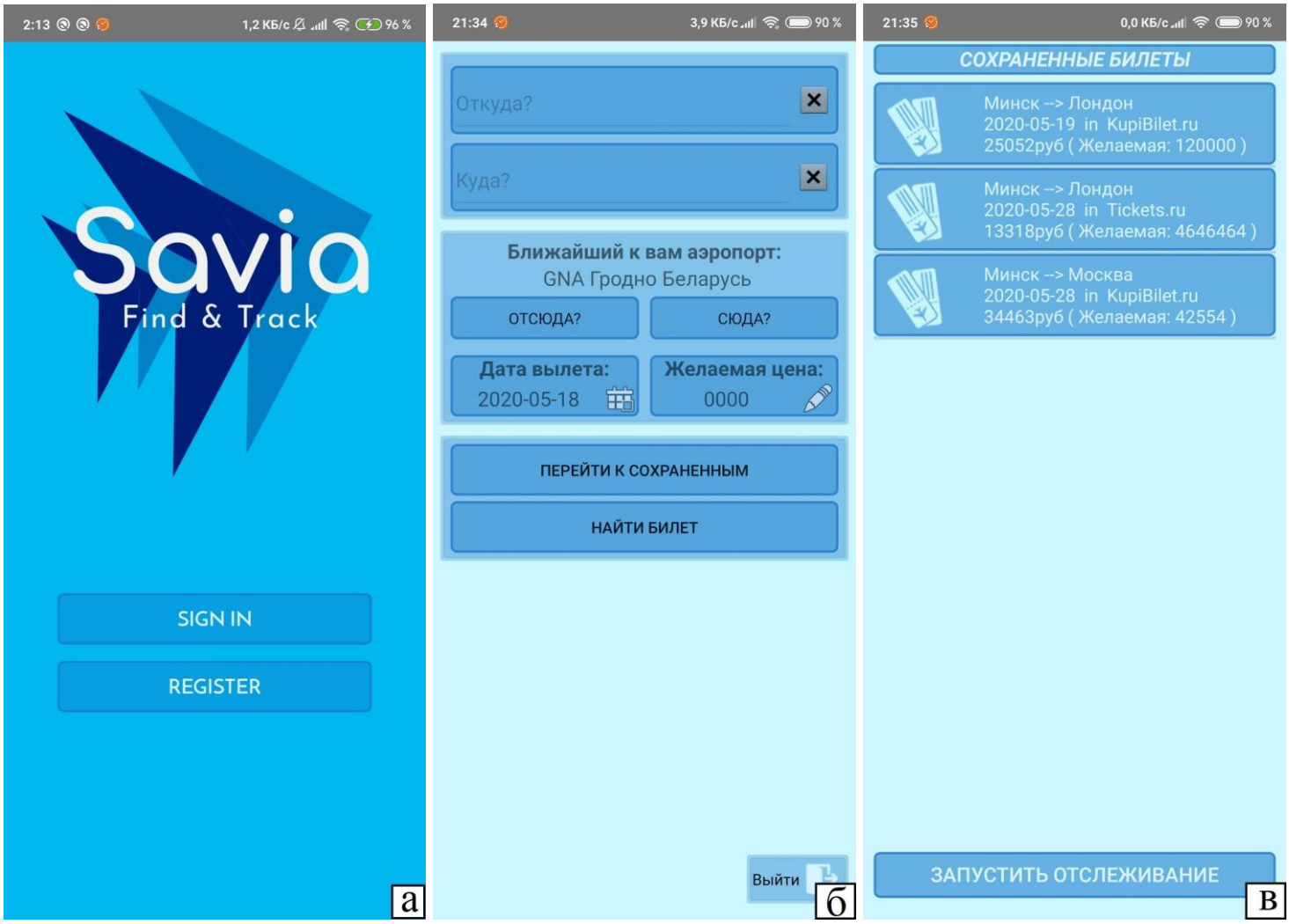

Рис. 2.3.1 - Основные экраны: экран для регистрации и входа (а), экран для поиска авиабилетов (б), экран управления сохраненными авиабилетами (в).

2.4 Базы данных

С целью упрощения и ускорения приложения используем для хранения информации об авиабилетах внутреннюю реляционную базу данных SQLite (рис. 2.4.1).

ID	FROMPLACE	TOPLACE	DATE	GATE	PRICE	WANTPRICE	ID_USER
1	MSQ Минск	LNA Лондон	2020-05-19	KupiBilet.ru	25052.0	120000.0	3

Рис. 2.4.1 – Условное представление таблицы SQLite

Для хранения данных учетных записей пользователей используем внешнюю реляционную базу данных Firebase (рис. 2.4.2).

Рис. 2.4.2– Содержание талицы People базы данных Firebise.

Исходный код данного модуля находится в приложении Б.

3. ОПИСАНИЕ РАБОТЫ ПРИЛОЖЕНИЯ

При открытии приложения пользователь видит экран для регистрации и входа (рис. 2.3.1а), содержащий кнопки для открытия карточек для ввода данных (рис. 3.1).

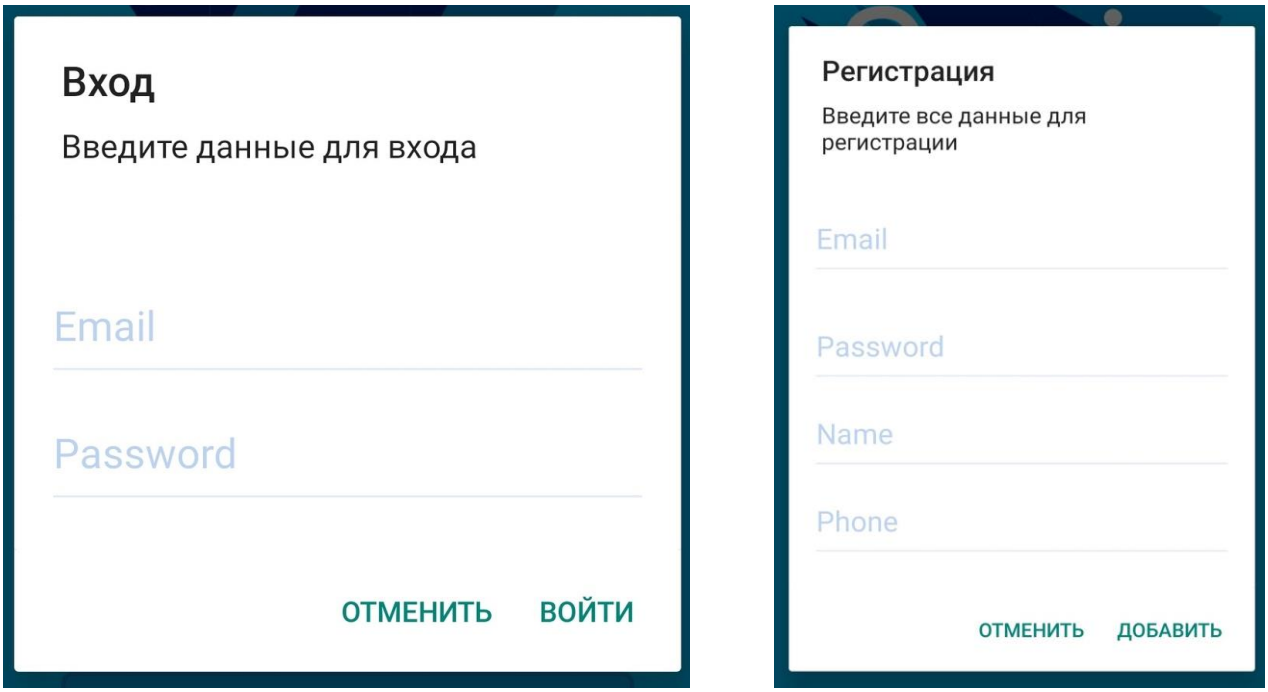

Рис 3.1 – Карточки для ввода данных при входе и регистрации

После успешного входа пользователь видит основной экран приложения (рис. 2.3.1б), служащий для поиска и сохранения найденного авиабилета. На данном этапе пользователь уже может перейти к списку сохраненных билетов. Основываясь на геоданных полученных через GPS и мобильный интернет пользователю предлагается ближайший по расстоянию аэропорт как точка вылета или прибытия. Для поиска авиабилета необходимо ввести данные для поиска (рис. 3.2).

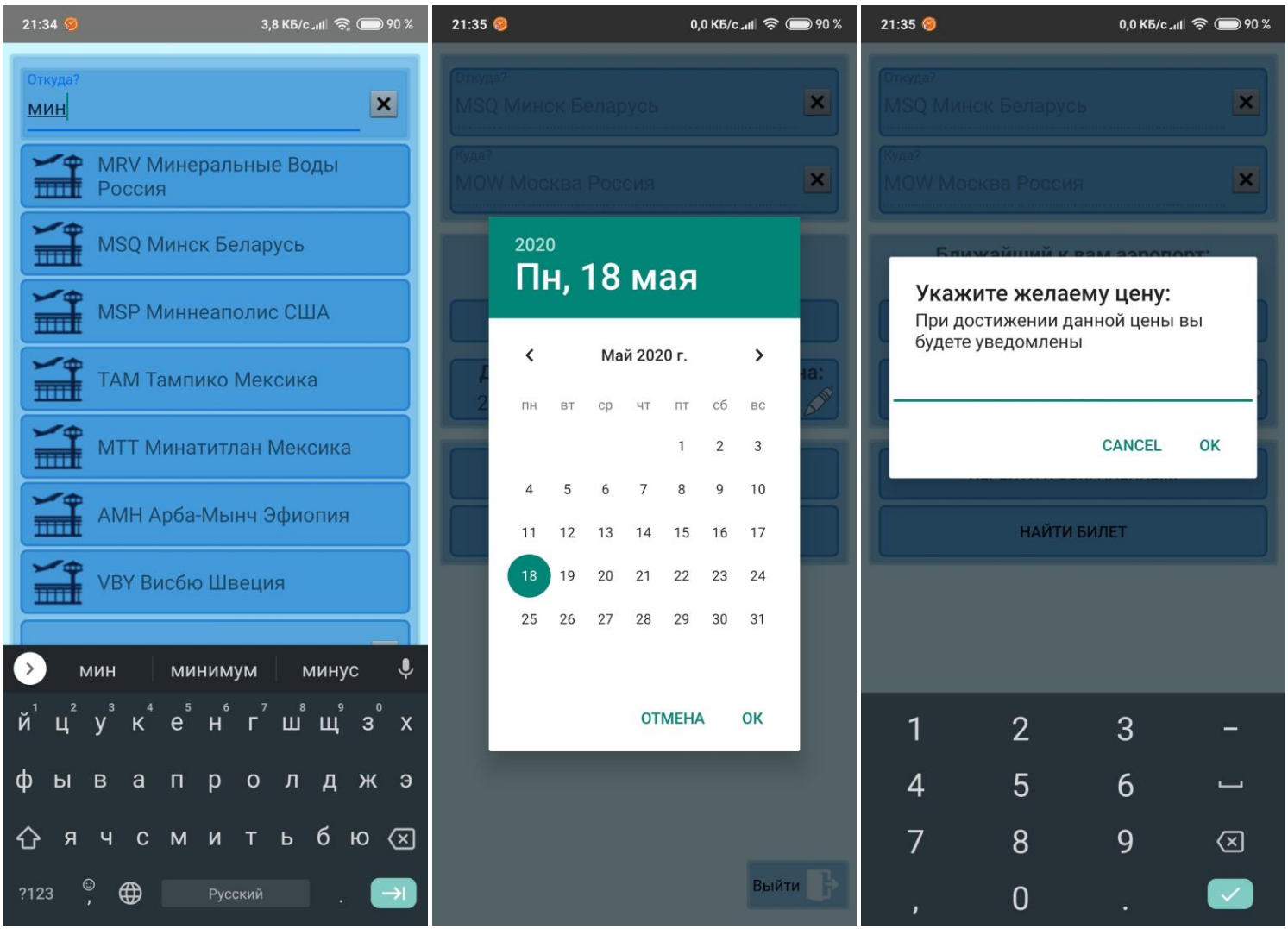

Рис 3.2 – Ввод необходимых данных для поиска авиабилета.

Результат поиска авиабилета отображается непосредственно на главном экране окном с данными авиабилета при успешном поиске (рис 3.3) и всплывающей подсказкой красного цвета при неуспешном (рис. 3.3).

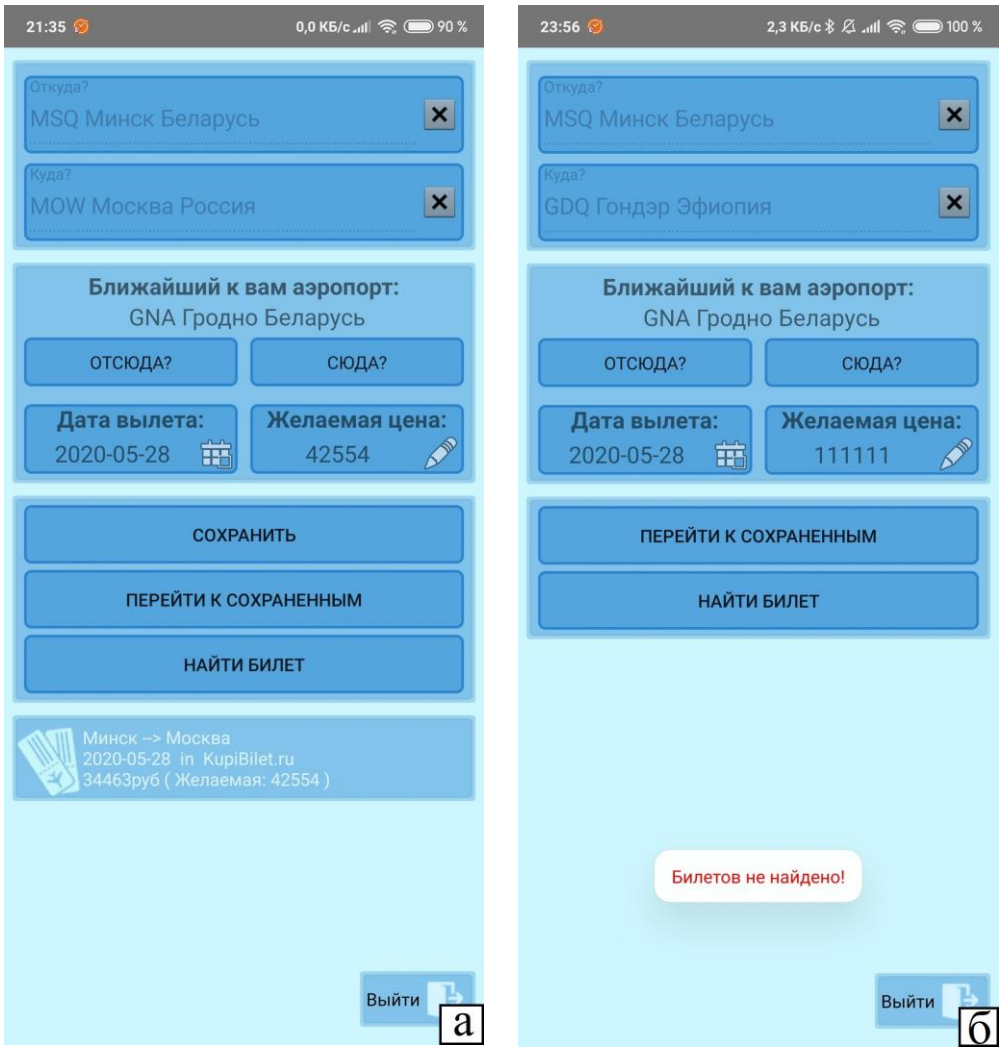

Рис 3.3 – Отображение успешного поиска (а) и неуспешного поиска авиабилета (б).

В приложении реализована система уведомлений и отслеживания. Для включения системы отслеживания актуальной стоимости и уведомления пользователя при достижении указанной желаемой цены ему необходимо на экране управления сохраненными авиабилетами (рис. 2.3.1в) нажать кнопку «отслеживать». Проверка стоимости и создание уведомлений происходит каждые 6 часов, поскольку на мой взгляд этот временной промежуток является оптимальным: достаточный для поддержки актуальной стоимости авиабилетов и недостаточный для надоедания и раздражения пользователя уведомлениями (рис 3.4).

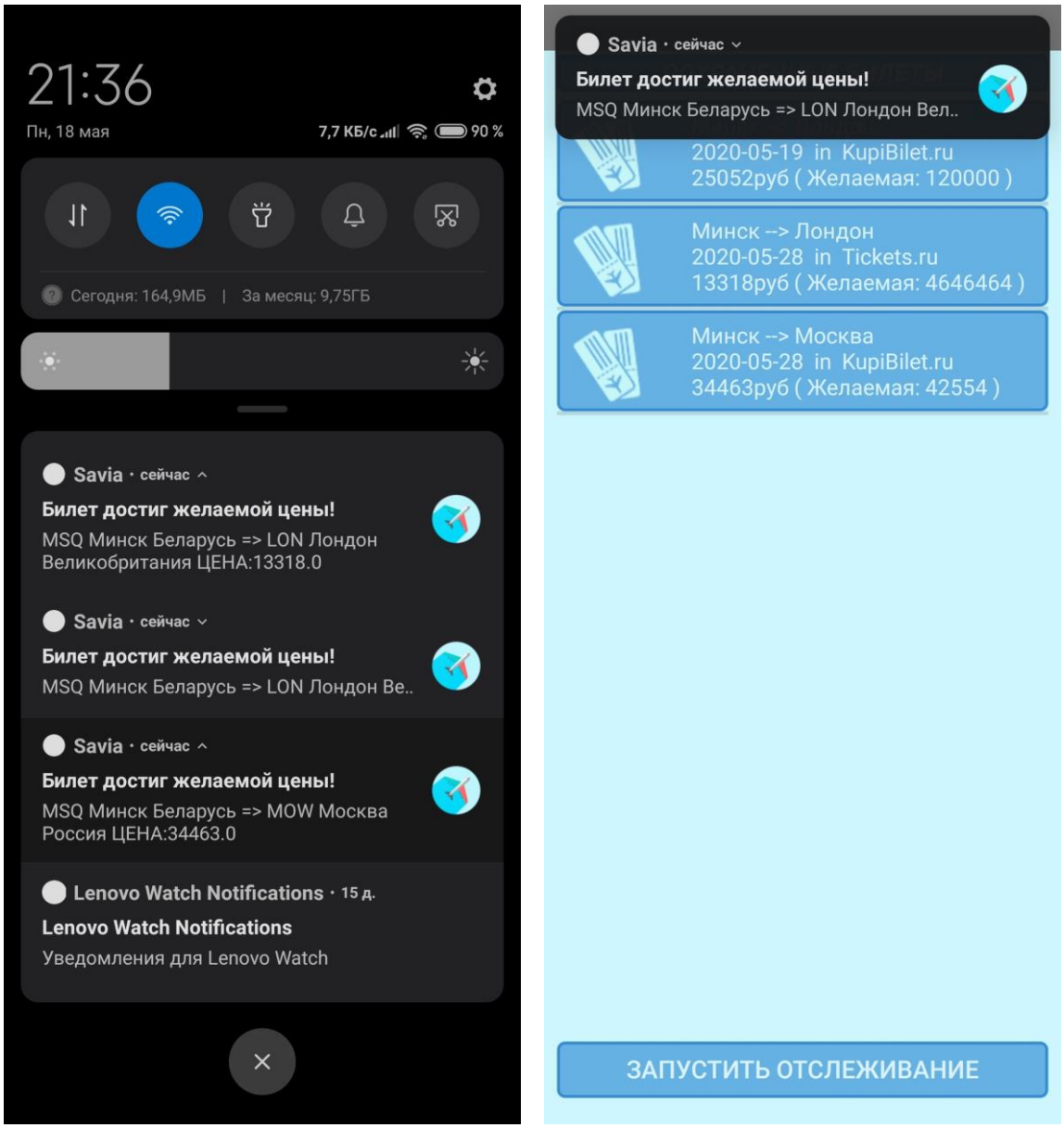

Рис 3.4 – Уведомления при достижении желаемой цены.

Система уведомлений создана с использованием многопоточности в целях ускорения работы приложения и независимости работоспособности остальных модулей от данной системы.

Исходный код системы уведомлений находится в приложении В.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы было разработано приложение, находящее наиболее дешевые билеты, позволяющее их хранить, актуализировать цены, а также уведомлять.

Ранее пользователи воздушного транспорта были вынуждены самостоятельно анализировать цены от различных перевозчиков и авиакомпаний, загружать «тяжелые» приложения, пропускать момент снижения цены. Теперь же достаточно указать немного данных и приложение находит самый дешевый билет и необходимое агентство. Сохраняет авиабилеты и уведомляет при достижении определенной цены. Приложение сохраняет частичную функциональность в offline-режиме: при отсутствии подключения к сети пользователь может просматривать и удалять сохраненные билеты.

Дальнейшие пути развития и улучшения приложения видны мною в следующем:

1. Создание графиков изменения цен за любой возможный временной промежуток.
2. Переход на страницу покупки авиабилета.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Travelpayouts: [сайт]. URL: https://support.travelpayouts.com/hc/ru/articles/203956163
2. Habr: [сайт]. URL: https://habr.com/ru/post/428736/
3. Metanit: [сайт]. URL: https://metanit.com/java/android/5.10.php