Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский**

**технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

Допускаю к защите

|  |  |
| --- | --- |
| Руководитель |  |
|  | подпись |
|  | В.Л. Аршинский |
|  | И.О. Фамилия |

|  |
| --- |
|  |
| Разработка приложения с использованием объектно-ориентированного подхода |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

|  |
| --- |
| Объектно-ориентированное программирование |

|  |
| --- |
| 1.011.00.00 – ПЗ |
| обозначение документа |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИСТб-19-2 |  |  | |  | Ю.В. Комогорцева |
|  |  | шифр |  | подпись | |  | И.О. Фамилия |
| Нормоконтроль |  |  |  |  | |  | В.Л. Аршинский |
|  |  |  |  | подпись | |  | И.О. Фамилия |
| Курсовой проект защищен с оценкой | | | | |  | | |

Иркутск 2021 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

ЗАДАНИЕ

на курсовое проектирование

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| По курсу | Объектно-ориентированное программирование | | | | | | | | | | | | | | | | | |
| Студенту | Комогорцевой Ю.В. | | | | | | | | | | | | | | | | | |
|  | (фамилия, инициалы) | | | | | | | | | | | | | | | | | |
| Тема проекта | | Разработка приложения с использованием объектно-ориентированного подхода | | | | | | | | | | | | | | | | |
| Исходные данные | | | Разработать Desktop-приложение для салонов красоты, предназначенное для автоматизации стратегий взаимодействия с клиентами, путём сохранения информации о клиентах и истории взаимоотношений с ним. | | | | | | | | | | | | | | | |
|  | | | | | | | | | |  |  |  |  |  |  |  |  |  |
| Графическая часть на | | | | листах. | | | | | | | | | | | | | | |
| Дата выдачи задания | | | | « | 1 | » | марта | | | | | 2021 г. | | | | |  |  |
|  | | | | | | | | | | | | | | | | | | |
| Задание получил | | | | | | | |  | | | | |  | | Ю.В. Комогорцева | | | |
|  | | | | | | | | подпись | | | | |  | | И.О. Фамилия | | | |
|  | | | | | | | | |  | |  | | |  | |  | |  |
| Дата представления проекта руководителю | | | | | | | | | « | | 29 | | | » | | мая | | 2021 г. |
|  | | | | | | | | | | | | | | | | | | |
| Руководитель курсового проектирования | | | | | | | |  | | | | |  | | В.Л. Аршинский | | | |
|  | | | | | | | | подпись | | | | |  | | И.О. Фамилия | | | |

**Содержание**

[Введение 4](#_Toc73015374)

[1 Анализ задания и описание предметной области 5](#_Toc73015375)

[1.1 Постановка задач 5](#_Toc73015376)

[1.2 Функциональные требования 6](#_Toc73015377)

[1.3 Выбор средств реализации 7](#_Toc73015378)

[2 Проектирование приложения 8](#_Toc73015379)

[2.1 Проектирование интерфейса 8](#_Toc73015380)

[2.2 Проектирование классов 10](#_Toc73015381)

[2.3 Проектирование базы данных 12](#_Toc73015382)

[3 Реализация приложения 14](#_Toc73015383)

[3.1 Спецификация реализованных классов 14](#_Toc73015384)

[4 Тестирование 16](#_Toc73015385)

[4.1 Методика тестирования 16](#_Toc73015386)

[4.2 Разработанные тесты 17](#_Toc73015387)

[4.3 Результаты тестирования 22](#_Toc73015388)

[Заключение 42](#_Toc73015389)

[Список использованных источников 43](#_Toc73015390)

# Введение

Целью курсового проекта является закрепление теоретических знаний полученных в процессе изучения курса «Объектно-ориентированное программирование» и развитие практических навыков применения объектного подхода в ходе решения конкретной практической задачи – разработки приложения с использованием объектно-ориентированного подхода, а также развитие способностей к обоснованному принятию самостоятельных решений в ходе проектной деятельности и получение базового опыта разработки программного обеспечения.

В рамках курсового проекта решается задача создания Desktop-приложения «EasyBeauty» - это информационная система для автоматизированного взаимодействия с клиентами и с возможностью хранить и редактировать данные об услугах, клиентах, сотрудниках, выполненных работах и записях. Вся информация хранится в локальной базе данных.

1. Анализ задания и описание предметной области

## Постановка задач

В ходе разработки необходимо решить следующие задачи:

1. Выработать функциональные требования к приложению
2. Выполнить проектирование интерфейса
3. Осуществить проектирование классов
4. Произвести проектирование базы данных с помощью DAO классов
5. Разработать приложение, основываясь на результатах проделанной работы вышеперечисленных пунктов
6. Определить методику тестирования и проверить работу приложения
7. Резюмировать результаты курсового проекта
   1. Функциональные требования

Приложение должно выполнять следующие функции:

1. Логирование всех действий пользователя
2. Просмотр отчетов о работе салона красоты
3. Изменение процентов начисления и списания бонусной системы
4. Работа с клиентами
   1. Вывод информации о клиентах в таблицу
      1. ФИО
      2. Номер телефона
      3. Дата рождения
      4. Количество бонусов
   2. Добавление информации о клиенте
   3. Редактирование информации о клиенте
   4. Удаление информации о клиенте
   5. Уведомление пользователя о дне рождения клиента
5. Работа с услугами
   1. Вывод информации об услугах таблицу
      1. Название услуги
      2. Стоимость
   2. Добавление информации об услуге
   3. Редактирование информации об услуге
   4. Удаление информации об услуге
6. Работа с сотрудниками
   1. Вывод информации о сотрудниках в таблицу
      1. ФИО
      2. Номер телефона
      3. Дата рождения
      4. Количество бонусов
   2. Добавление информации о сотруднике
   3. Редактирование информации о сотруднике
   4. Удаление информации о сотруднике
   5. Уведомление пользователя о дне рождения сотрудника
7. Работа с информацией о записях
   1. Вывод информации о записях в таблицу
      1. Данные о клиенте
      2. Данные о выбранной услуге
      3. Данные о сотруднике
      4. Дата и время записи
   2. Добавление новой записи
   3. Редактирование существующей записи
   4. Удаление информации о записи
   5. Автоматическое удаление прошедших по времени записей
   6. Возможность просмотра записей на день
   7. Уведомление пользователя о предстоящей записи
8. Работа с информацией о выполненных работах
   1. Вывод информации о выполненных работах в таблицу
      1. Данные о клиенте
      2. Данные о сотруднике
      3. Данные о выбранной услуге
      4. Сумма дохода и использованные баллы
      5. Дата и время выполнения
   2. Добавление новой выполненной работы
   3. Выполнение возврата средств
   4. Выбор средств реализации

Были использованы следующие технологии:

* + - 1. IntelliJ IDEA – IDE для разработки приложения
      2. Драйвер JDBC для SQLite — JDBC (Java Data Base Connectivity)— это API для работы с базой данных на Java для SQLite

# Проектирование приложения

* 1. Проектирование интерфейса

На основе анализа функциональных требований были выделены следующие окна приложения (см. рис. 1):

1. Главное окно – на нем находятся такие группы кнопок:

Записи: «Сделать запись», «Записи на день», «Все записи»;

Услуги: «Оказание услуги», «Выполненные работы», «Услуги»;

Клиенты: «Клиенты», «Новый клиент»;

Сотрудники: «Сотрудники», «Новый сотрудник»;

Так же главное окно содержит панели для уведомления пользователя о днях рождениях и предстоящих записях. Так же в правом верхнем углу есть 3 кнопки: «Бонусная программа», «Отчеты», «Log»

1. Окно «Сотрудники» содержит таблицу с информацией о сотрудниках и кнопки которые позволяют добавлять, редактировать ,удалять запись и вернуться на главное окно.
2. Окно «Клиенты» содержит таблицу с информацией о клиентах и кнопки которые позволяют добавлять, редактировать и удалять запись и вернуться на главное окно.
3. Окно «Все записи» содержит таблицу с информацией о всех записях и кнопки которые позволяют добавлять, редактировать и удалять запись и вернуться на главное окно.
4. Окно «Записи на день» содержит таблицу с информацией о записях на день и кнопки которые позволяют добавлять, редактировать и удалять запись и вернуться на главное окно.
5. Окно «Выполненные работы» содержит таблицу с информацией о сотрудниках и кнопки которые позволяют добавлять ,делать возврат средств и вернуться на главное окно
6. Окно «Записи на день» содержит таблицу с информацией о сотрудниках и кнопки которые позволяют добавлять, редактировать и удалять запись.

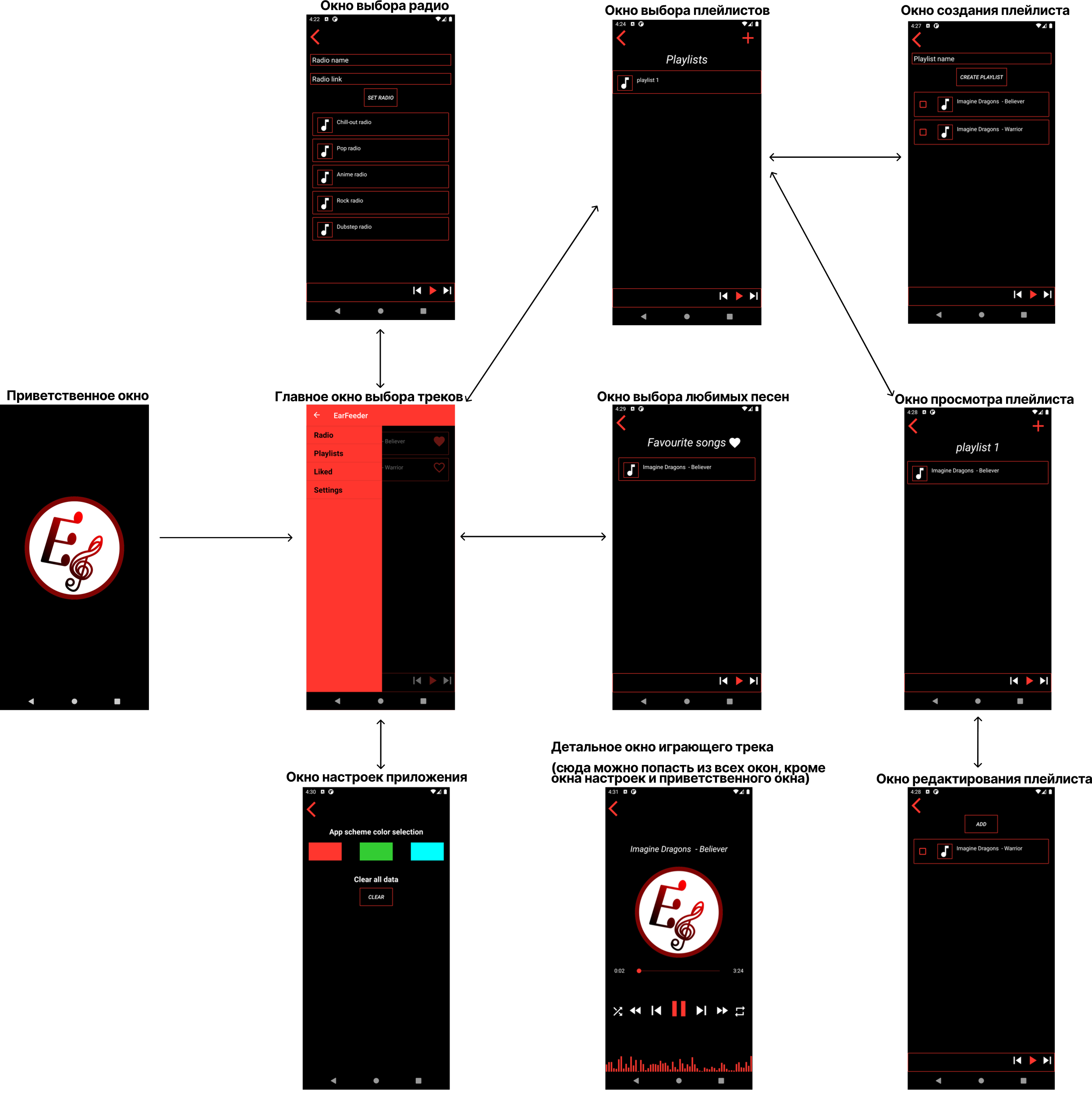


Рисунок 1 – Схема переходов между окнами приложения

* 1. Проектирование классов

Для реализации работы приложения с базой данных были определены следующие классы-сущности:

1. Track – музыкальная композиция, взятая из хранилища пользователя.
2. Radio – интернет-радиоканал
3. Playlist – список музыкальных композиций, собранных лично пользователем
4. TrackPlaylist – принадлежность музыкальных композиций к плейлистам

Были выделены следующие группы классов:

1. Database – на основе принадлежности к базе данных (сущности, запросы и абстрактный класс, соединяющий все классы-запросы)
2. Service – на основе наследование от класса Service
3. Player – отдельный класс для управления музыкальным плеером
4. App – запускающий класс, содержащий все важные переменные и методы, позволяющие приложению функционировать без лишних переменных, храня единственный экземпляр нужных значений

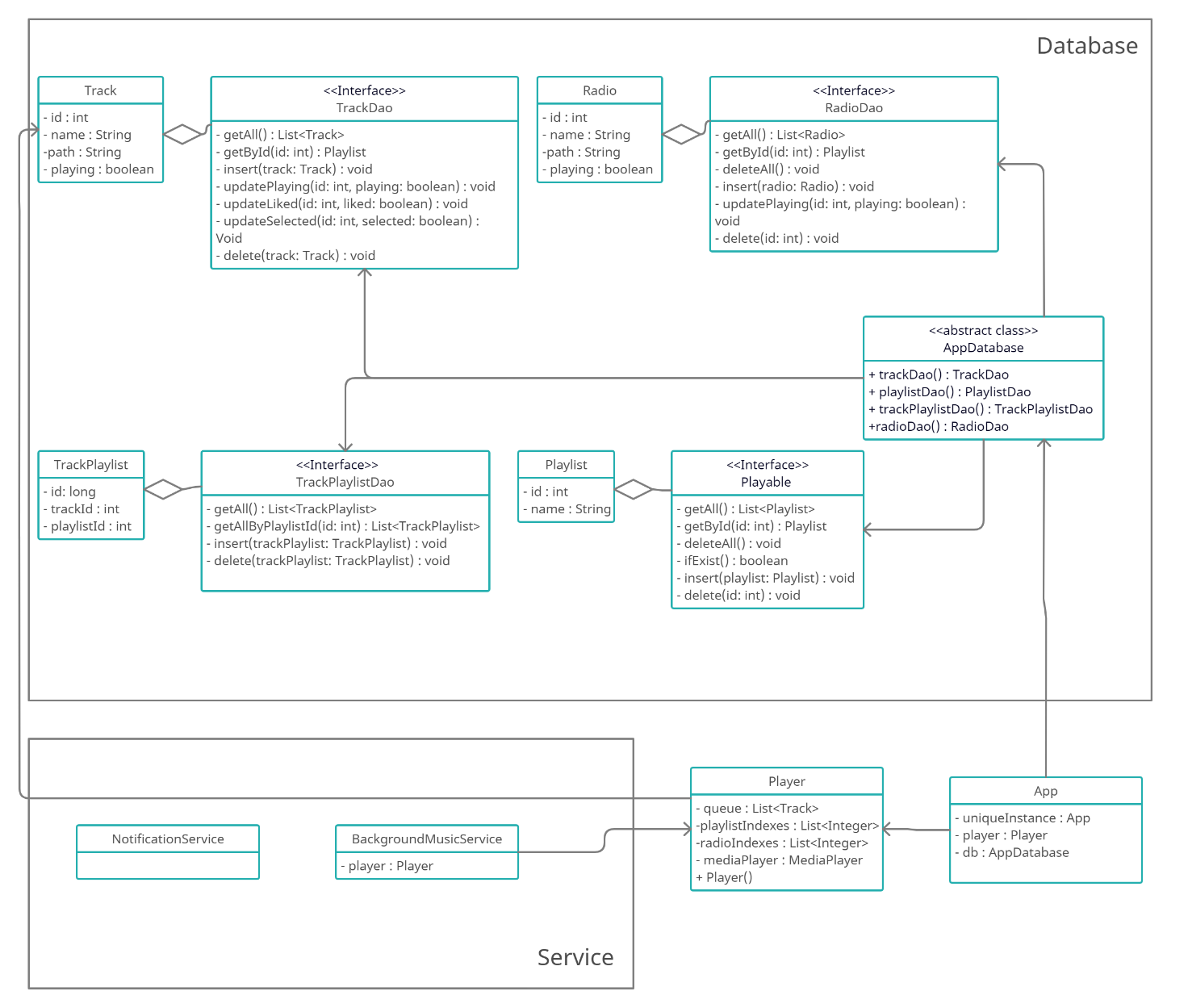


Рисунок 2 – Диаграмма UML, первый фрагмент

Для взаимодействия пользователя с приложением были определены следующие группы классов:

1. Activity – группа экранов приложения
2. Playable – интерфейс, реализующий взаимодействие между плеером и меню уведомлений

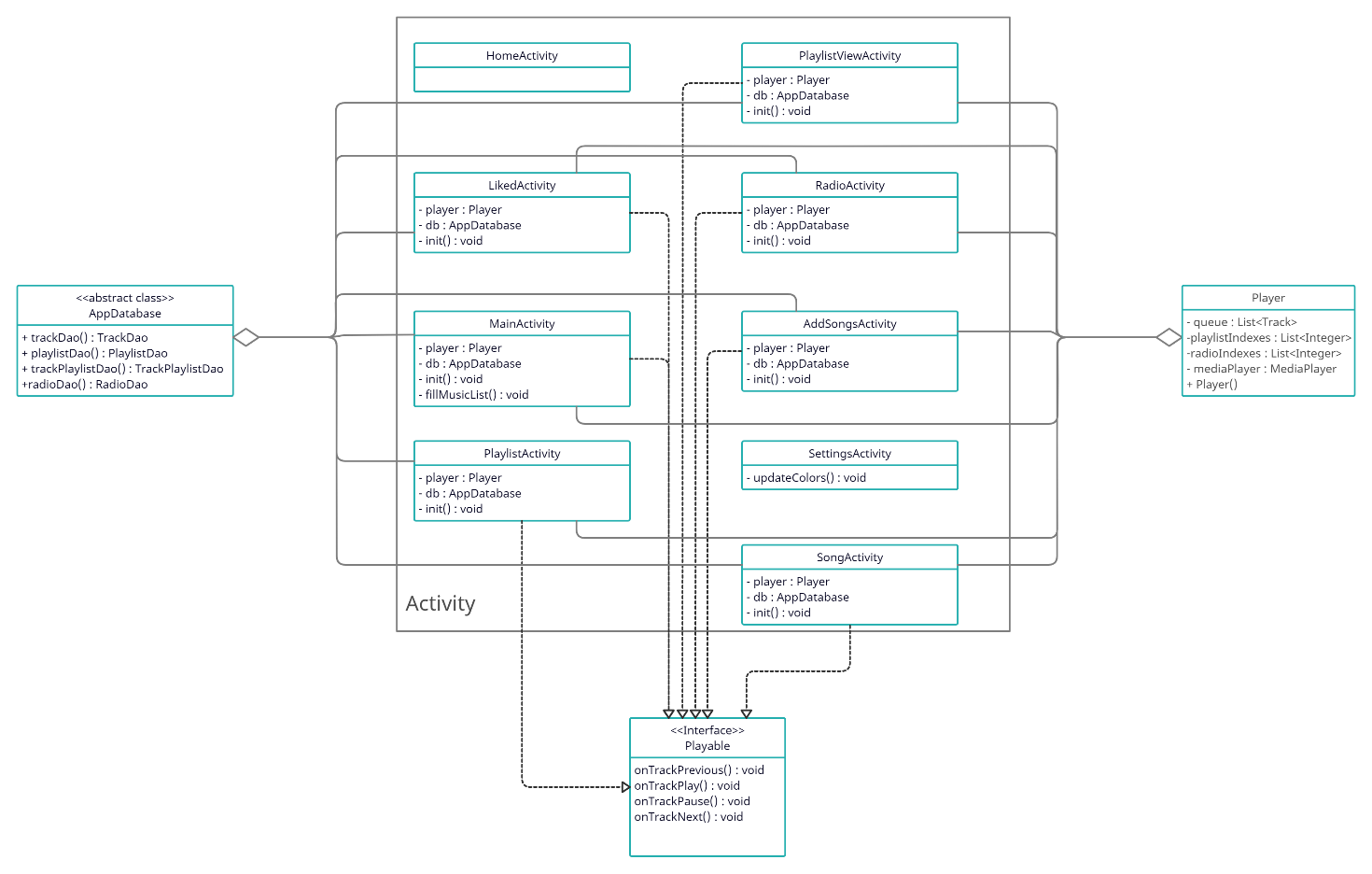


Рисунок 3 – Диаграмма классов UML, второй фрагмент

* 1. Проектирование базы данных

Для работы с базой данных на основе упомянутых сущностей в проектировании классов (см. рис. 2 и рис. 3) были созданы следующие таблицы (см. рис. 4).

Первичным ключом во всех четырех таблицах является их уникальный идентификатор id, по которому таблицы Track, Playlist и TrackPlaylist могут «общаться» друг с другом.

В таблице TrackPlaylist хранятся id треков и плейлистов, к которым они принадлежат. В то время, как таблица Radio существует сама по себе, хранит только записи о интернет-радиоканалах и не взаимодействует ни с какой

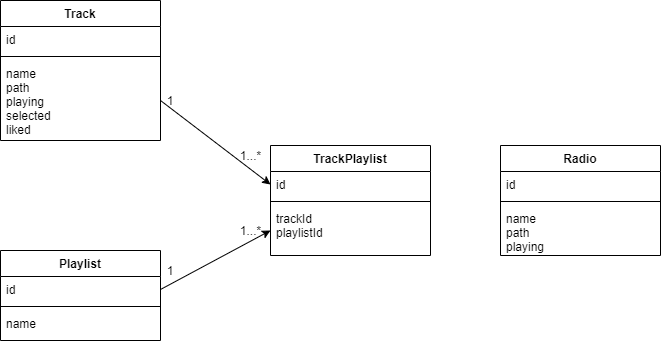


Рисунок 4 - диаграмма классов базы данных

Таблица 1 – Описание таблиц БД

|  |  |  |  |
| --- | --- | --- | --- |
| Имя таблицы | Имя атрибута | Тип | Описание |
| services | name | text | Идентификатор услуги в виде её названия |
| price | float | Стоимость услуги |
| clients | id\_clients | integer autoincrement | Идентификатор клиента |
| surname\_clients | text | Фамилия |
| name\_clients | text | Имя |
| middle\_clients | text | Отчество |
| birhdate | text | День рождения |
| email | text | Почта |
| phone | text not null | Телефон клиента |
| point | float | Количество баллов |

Таблица 1 - Продолжение

|  |  |  |  |
| --- | --- | --- | --- |
| employee | id\_employee | integer autoincrement | Идентификатор сотрудника |
| surname\_employee | text | Фамилия |
| name\_employee | text | Имя |
| middle\_employee | text | Отчество |
| birhdate | text | День рождения |
| post | text | Должность |
| phone | text not null | Телефон сотрудника |
| performedWork | id\_work | integer autoincrement | Идентификатор записи о выполненной работе |
| name | text | Название услуги |
| price | float | Стоимость услуги |
| point | float | Сколько баллов было использовано |
| income | float | Доход=price-point |
| id\_clients | integer | Идентификатор клиента |
| surname\_clients | text | Фамилия |
| name\_clients | text | Имя |
| middle\_clients | text | Отчество |
| id\_employee | integer | Идентификатор сотрудника |
| surname\_employee | text | Фамилия |
| name\_employee | text | Имя |
| middle\_employee | text | Отчество |
| date | text | Дата выполнения услуги |
| time | text | Время выполнения услуги |
| comment | text | Комментарий |
| record | id\_record | integer autoincrement | Идентификатор записи |
| name | text | Название услуги |
| price | float | Стоимость услуги |
| id\_clients | integer | Идентификатор клиента |
| surname\_clients | text | Фамилия |
| name\_clients | text | Имя |
| middle\_clients | text | Отчество |
| id\_employee | integer | Идентификатор сотрудника |
| surname\_employee | text | Фамилия |
| name\_employee | text | Имя |
| middle\_employee | text | Отчество |
| date | text | Дата записи |
| time | text | Время записи |
| comment | text | Комментарий |
| rewards | transfersPoint | int not null | Количество зачисляющихся бонусов клиенту с каждой покупки в процентах |
| discountPoint | int not null | Количество скидки из имеющихся у клиента бонусов при каждой покупки в процентах |

Реализованные SQL-запросы к базе данных:

services :

1. CREATE TABLE if not exists 'services' ('name' text PRIMARY KEY NOT NULL, ‘price' float NOT NULL);
2. INSERT INTO services ('name', 'price') VALUES(?,?) – Добавляет новую услугу в БД

clients :

1. CREATE TABLE if not exists 'clients' ('id\_clients' INTEGER PRIMARY KEY AUTOINCREMENT, 'surname\_clients' text, 'name\_clients' text NOT NULL, 'middle\_clients' text, 'birhdate' text, 'email' text, 'phone' text NOT NULL, 'point' float );

employee :

1. CREATE TABLE if not exists 'employee' ( 'id\_employee' INTEGER PRIMARY KEY AUTOINCREMENT, 'surname\_employee' text, 'name\_employee' text NOT NULL, 'middle\_employee' text, 'birhdate' text, 'post' text, 'phone' text NOT NULL);

performedWork :

1. **CREATE TABLE if not exists 'performedWork' ("**   
    **" 'id\_work' INTEGER PRIMARY KEY AUTOINCREMENT,"** +  
    **" 'name' text NOT NULL,"** +  
    **" 'price' float NOT NULL,"**+  
    **" 'point' float ,"**+*//сколько баллов спишется* **" 'income' float ,"**+*//доход=прайс-пойнт* **" 'id\_clients' INTEGER ,"** +  
    **" 'surname\_clients' text,"** +  
    **" 'name\_clients' text NOT NULL,"** +  
    **" 'middle\_clients' text,"** +  
    **" 'id\_employee' INTEGER ,"** +  
    **" 'surname\_employee' text,"** +  
    **" 'name\_employee' text NOT NULL,"** +  
    **" 'middle\_employee' text,"** +  
    **" 'date' text,"** +  
    **" 'time' text,"** +  
    **" 'comment' text );**

# Реализация приложения

* 1. Спецификация реализованных классов

Пакет Activity:

1. HomeActivity – приветственный экран с логотипом приложения
2. LikedActivity –экран с любимыми песнями пользователя
3. MainActivity – главный экран со всеми треками с устройства
4. PlaylistActivity –экран с просмотром и возможностью добавления плейлистов
5. PlaylistViewActivity – экран с просмотром песен в выбранном плейлисте
6. RadioActivity – экран с возможностью прослушивания и добавления радио
7. AddSongsActivity – экран с возможность добавления песен в выбранный плейлист
8. CreatingPlaylistActivity – экран с созданием плейлиста
9. SettingsActivity – экран с настройками приложения
10. SongActivity – экран текущей песни с расширенными возможностями управления

Пакет Adapter:

1. LikeTrackAdapter – редактирование показа любимых песен в списке
2. PlaylistAdapter – редактирование показа плейлистов в списке
3. TrackAdapter - редактирование показа песен в списке
4. RadioAdapter - редактирование показа радио в списке
5. TrackAdapterSelect – редактирования списка песен с возможностью выбора (checkbox)

ORM (Object-Relational Mapping) – техника, создающая виртуальную объектную БД и добавляющая дополнительный слой абстракции на работу с ней.

DAO (Data Access Object) - интерфейс, отвечающий за передачу запросов в БД и обработку полученных от нее ответов.

Пакет Database:

1. Playlist – сущность «плейлист»
2. PlaylistDao – запросы к сущности «плейлист»
3. Radio – сущность «радио»
4. RadioDao – запросы к сущности «радио»
5. Track – сущность «трек»
6. TrackDao – запросы к сущности «трек»
7. TrackPlaylist – таблица связей «треков» и «плейлистов»
8. AppDatabase – набор абстрактных ссылок на классы запросов

Пакет Service:

1. BackgroundMusicService – сервис, который организует воспроизведение музыки
2. NotificationService – сервис, который позволяет отображать уведомления
3. NotificationActionService – сервис, который организует отправку действий, которые происходят на уведомлении

Пакет Notification:

1. CreateNotification – создание уведомления
2. Playable – интерфейс, позволяющий управлять уведомлением

Пакет Fragment:

1. BottomFragment – нижняя панель приложения для управления треками
2. ClearAllFragment – кнопка очистки данных приложения
3. ColorSelectionFragment – панель управления цветовой схемой приложения

Singleton – паттерн, позволяющий иметь лишь единственный экземпляр некоторого класса через предоставление глобального доступа к этому экземпляру [9].

LoadingDialog – управление панелью загрузки

RecyclerItemListener – переопределение событий нажатия на RecyclerView

AppColor – организация управления цветовой схемой приложения

Player – все переменные и методы, нужные для воспроизведения музыки

App – запускающий класс для управления и хранения главных состояний приложения, основан на паттерне Singleton

# Тестирование

## Методика тестирования

В качестве метода тестирования было выбрано приемочное тестирование.

Целью данного метода является определение готовности продукта, что достигается путем прохода тестовых сценариев и случаев, которые построены на основе спецификации требований к разрабатываемому ПО.

## Разработанные тесты

Таблица 2 – Тесты в главном меню MainFrame

|  |  |  |
| --- | --- | --- |
| № | Совершенное действие | Ожидаемый результат |
| 1 | Запуск приложения. |  |
| 2 | Нажатие кнопки «Клиенты» на главном окне . |  |
| 3 | Нажатие кнопки «Новый клиент» на главном окне. |  |
| 4 | Нажатие кнопки «Сотрудники» на главном окне. |  |
| 5 | Нажатие кнопки «Новый сотрудник» на главном окне. |  |
| 6 | Нажатие кнопки «Все записи» на главном окне. |  |
| 7 | Нажатие кнопки «Записи на день» на главном окне. |  |
| 8 | Нажатие кнопки «Сделать запись» на главном окне. |  |
| 9 | Нажатие кнопки «Услуги» на главном окне. |  |
| 10 | Нажатие кнопки «Выполненные работы» на главном окне. |  |
| 11 | Нажатие кнопки «Оказание услуги» на главном окне. |  |
| 12 | Нажатие кнопки «Бонусная программа» на главном окне в верхнем левом углу. |  |
| 13 | Нажатие кнопки «Отчеты» на главном окне в верхнем левом углу. |  |
| Клиенты | | |
| 14 | Нажатие кнопки «Новый клиент» на окне «Клиенты» |  |
| 15 | Нажатие кнопки «Удалить клиента» на окне «Клиенты» |  |
| 16 | Нажатие кнопки «Редактировать» на окне «Клиенты» |  |
| 17 | Нажатие кнопки «На главную» на окне «Клиенты» |  |
| Услуги | | | |
| 18 | Нажатие кнопки «Новый клиент» на окне «Услуги» |  | |
| 19 | Нажатие кнопки «Удалить клиента» на окне «Услуги» |  | |
| 20 | Нажатие кнопки «Редактировать» на окне «Услуги» |  | |
| 21 | Нажатие кнопки «На главную» на окне «Услуги» |  | |
| 22 | Сотрудники | | |
| 23 | Нажатие кнопки «Новый клиент» на окне «Сотрудники» |  | |
| 24 | Нажатие кнопки «Удалить клиента» на окне «Сотрудники» |  | |
| 25 | Нажатие кнопки «Редактировать» на окне «Сотрудники» |  | |
| 26 | Нажатие кнопки «На главную» на окне «Сотрудники» |  | |
| Все записи | | | |
| 28 | Нажатие кнопки «Новый клиент» на окне «Сотрудники» |  | |
| 29 | Нажатие кнопки «Удалить клиента» на окне «Сотрудники» |  | |
| 30 | Нажатие кнопки «Редактировать» на окне «Сотрудники» |  | |
| № | Нажатие кнопки «На главную» на окне «Сотрудники» |  | |
| Записи на день | | | |
| 32 | Нажатие кнопки «Новый клиент» на окне «Сотрудники» |  | |
| № | Нажатие кнопки «Удалить клиента» на окне «Сотрудники» |  | |
| 33 | Нажатие кнопки «Редактировать» на окне «Сотрудники» |  | |
| 34 | Нажатие кнопки «На главную» на окне «Сотрудники» |  | |
| Выполненные работы | | | |
| 36 | Нажатие кнопки «Новый клиент» на окне «Сотрудники» |  | |
| 37 | Нажатие кнопки «Удалить клиента» на окне «Сотрудники» |  | |
| 38 | Нажатие кнопки «Редактировать» на окне «Сотрудники» |  | |
| 39 | Нажатие кнопки «На главную» на окне «Сотрудники» |  | |
| 40 | Нажатие на checkbox у трека.  Checkbox нажмется, либо отожмется. | | |
| 41 | Нажатие на кнопку «создать плейлист». | Все выбранные (checkbox’ом) треки образуют новый плейлист на PlaylistActivity. | |
| 42 | Нажатие на кнопку «назад». | Действие вернется на PlaylistActivity. | |

Таблица 8 – Тесты на PlaylistViewActivity

|  |  |  |
| --- | --- | --- |
| № | Совершенное действие | Ожидаемый результат |
| 43 | Нажатие на трек. | См пункт 1. |
| 44 | Нажатие на кнопку «назад». | Действие вернется на PlaylistActivity. |
| 45 | Нажатие на кнопку «добавить». | Откроется AddSongsActivity, где можно будет добавить треки в данный плейлист. |

Таблица 9 – Тесты на AddSongsActivity

|  |  |  |
| --- | --- | --- |
| № | Совершенное действие | Ожидаемый результат |
| 46 | Создание activity. | Появятся все песни, не принадлежащие выбранному плейлисту. |
| 47 | Нажатие на трек. | См пункт 1. |
| 48 | Нажатие на checkbox у трека. | Checkbox нажмется, либо отожмется. |
| 49 | Нажатие на кнопку «добавить». | Выбранные треки добавятся в выбранный плейлист. |
| 50 | Нажатие на кнопку «назад». | Действие вернется на PlaylistViewActivity. |

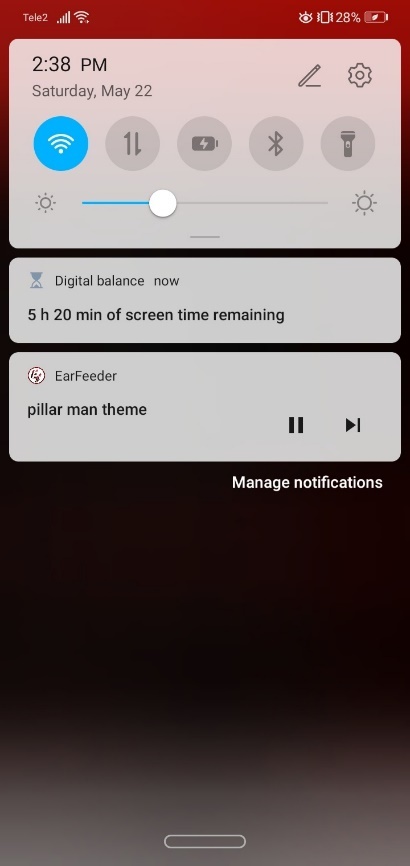
Таблица 10 – Тесты на RadioAcitivty

|  |  |  |
| --- | --- | --- |
| № | Совершенное действие | Ожидаемый результат |
| 51 | Нажатие на радио. | На экране появится колесо загрузки, которое нельзя прервать. Радио начнет воспроизводиться. |
| 52 | Нажатие на радио, когда нет интернета. | Всплывет сообщение «Нет интернет подключения». |
| 53 | Нажатие на кнопку «назад». | См пункт 20. |
| 54 | Нажатие на кнопку «добавить». | На экране появится колесо загрузки, которое нельзя прервать. Радио добавится в список и начнет воспроизводиться. |
| 55 | Нажатие на кнопку «добавить», когда нет интернета. | Всплывет сообщение «Нет интернет подключения». |
| 56 | Нажатие на кнопку «добавить», когда ссылка неверная. | Всплывет сообщение «Неверная ссылка». |

Таблица 10 - Продолжение

|  |  |  |
| --- | --- | --- |
| № | Совершенное действие | Ожидаемый результат |
| 57 | Нажатие на кнопку «старт» / «пауза» на нижней панели приложения, когда играет радио. | Появится загрузка, пока не включится радио. Поменяется иконка воспроизведения. |
| 58 | Нажатие на кнопку «следующий» на нижней панели приложения, когда играет радио. | Появится загрузка, пока радио не переключится на последующее. |
| 59 | Нажатие на кнопку «следующий» на нижней панели приложения, когда дальше нет радио. | Ничего не произойдет. |
| 60 | Нажатие на кнопку «предыдущий» на нижней панели приложения, когда играет радио. | Появится загрузка, пока радио не переключится на предыдущее. |
| 61 | Нажатие на кнопку «предыдущий» на нижней панели приложения, когда перед текущим больше нет радио. | Ничего не произойдет. |

## Результаты тестирования



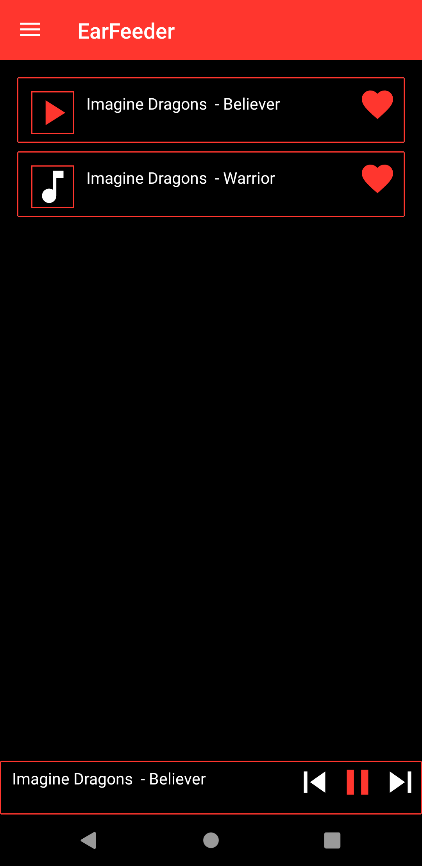


Рисунок 5 - результат теста 1

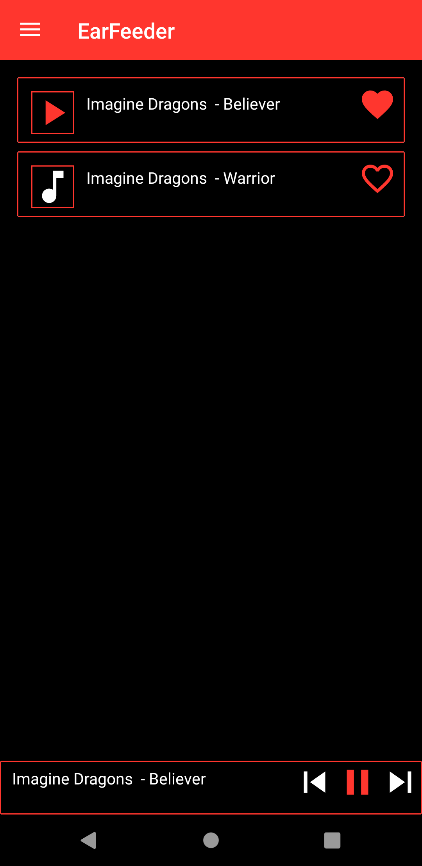
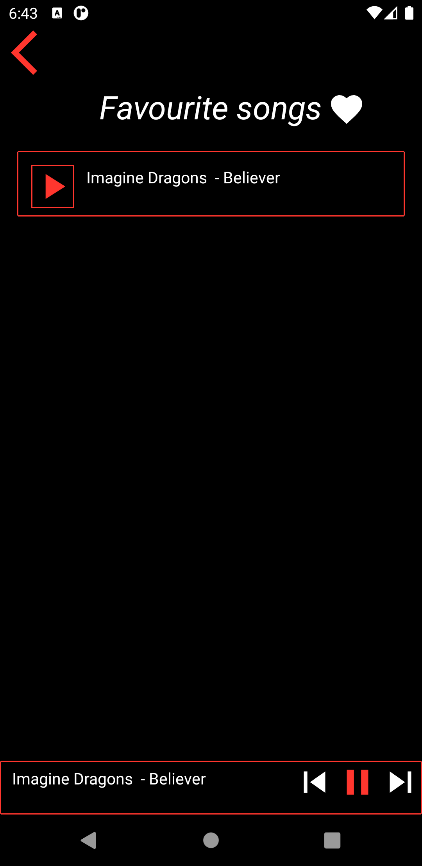


Рисунок 6 – результат теста 2

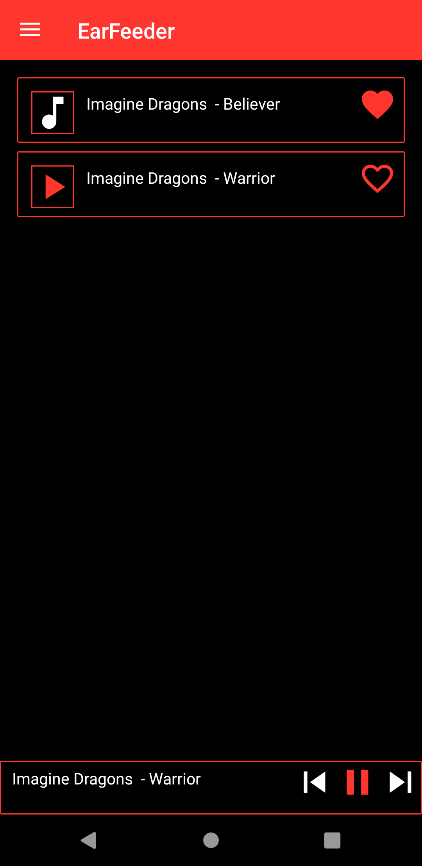
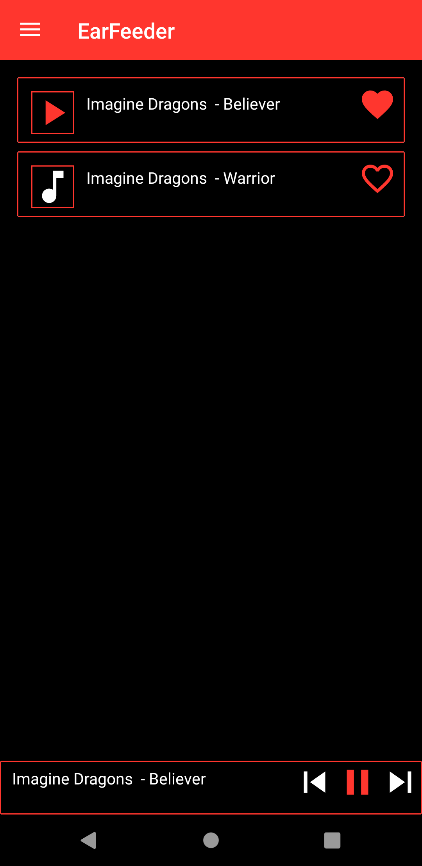


Рисунок 7 – а) результат теста 3; б) результат теста 5

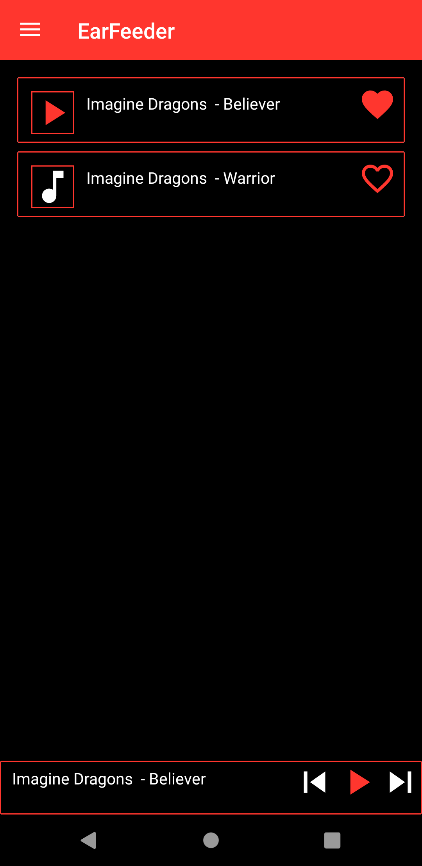
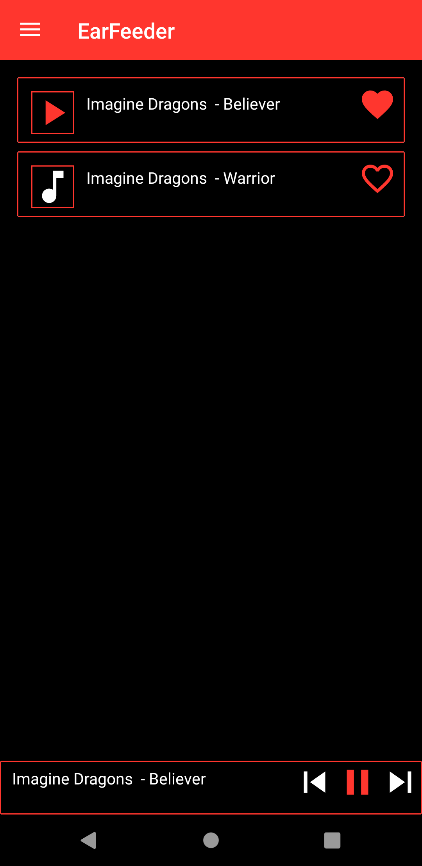


Рисунок 8– результат теста 7

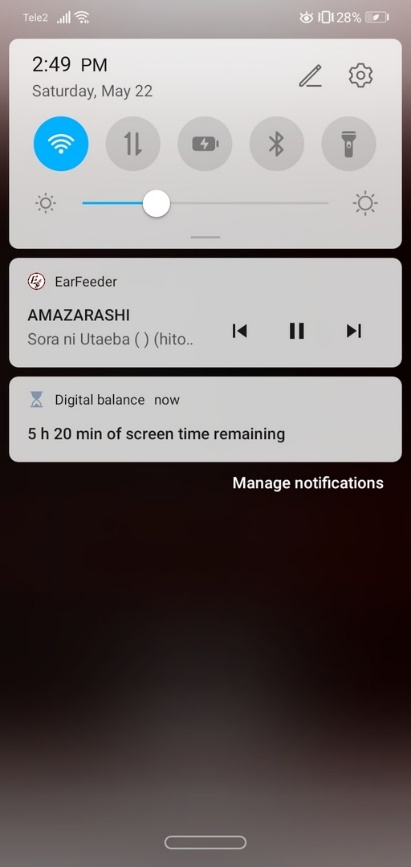
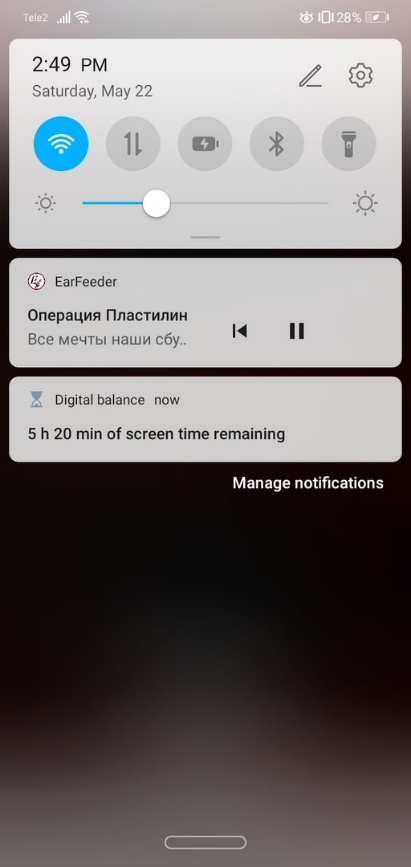


Рисунок 9 – а) результат теста 9; б) результат теста 10

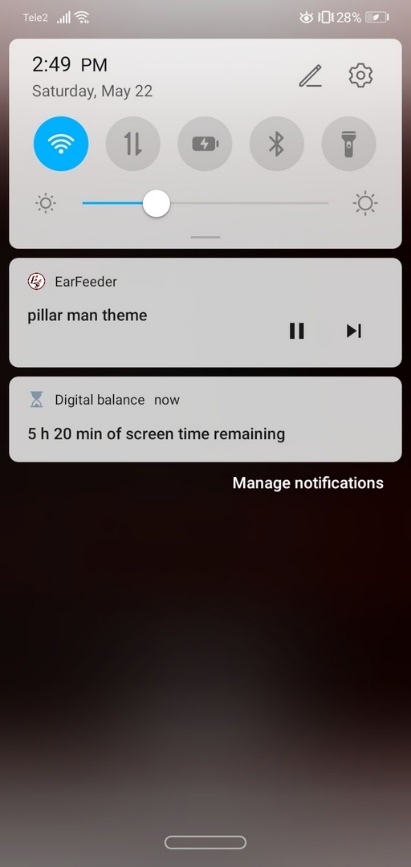
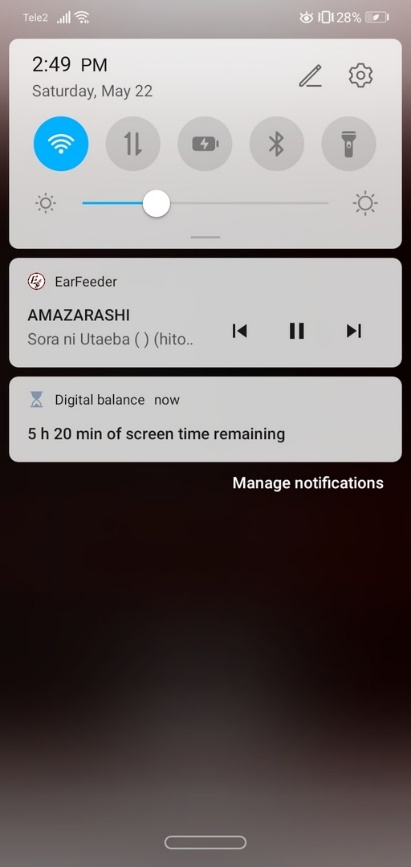


Рисунок 10 – результат тестов 11 и 12

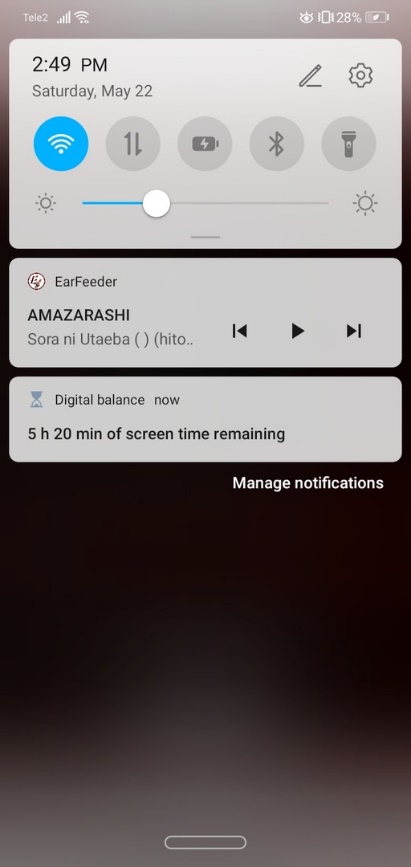
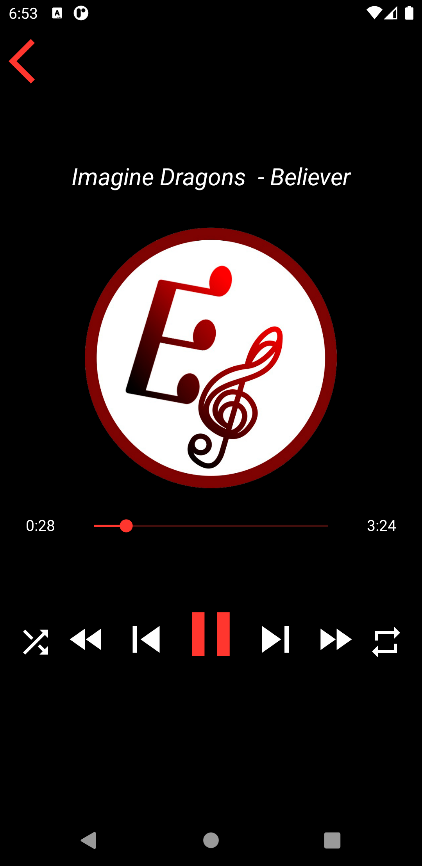


Рисунок 11 – а) результат теста 13; б) результат теста 15

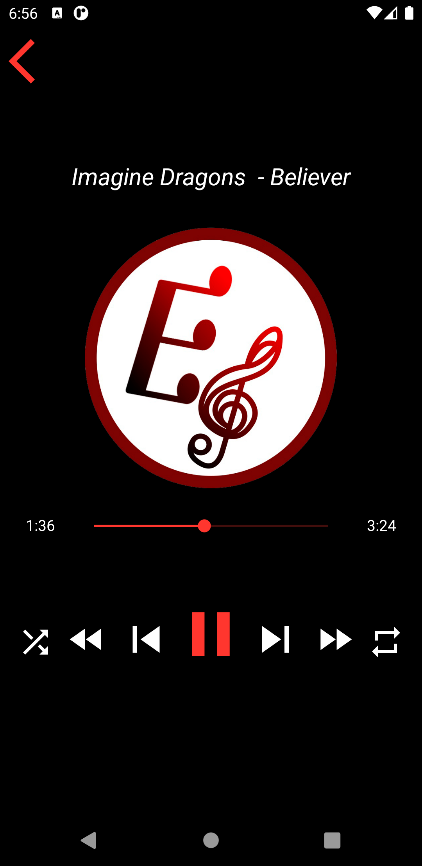


Рисунок 12 – а) результат теста 17; б) результат тестов 18 и 19

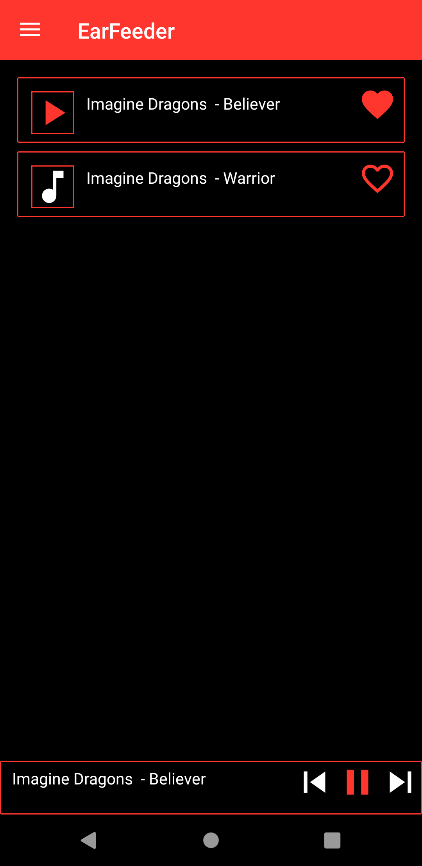
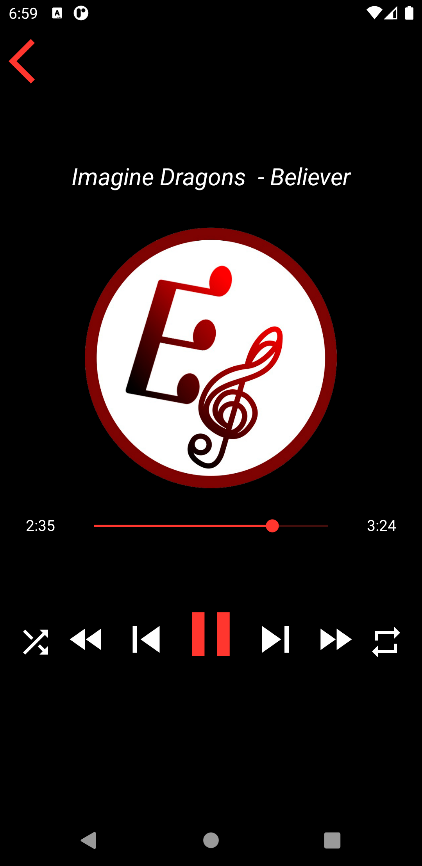


Рисунок 13 – результат теста 20



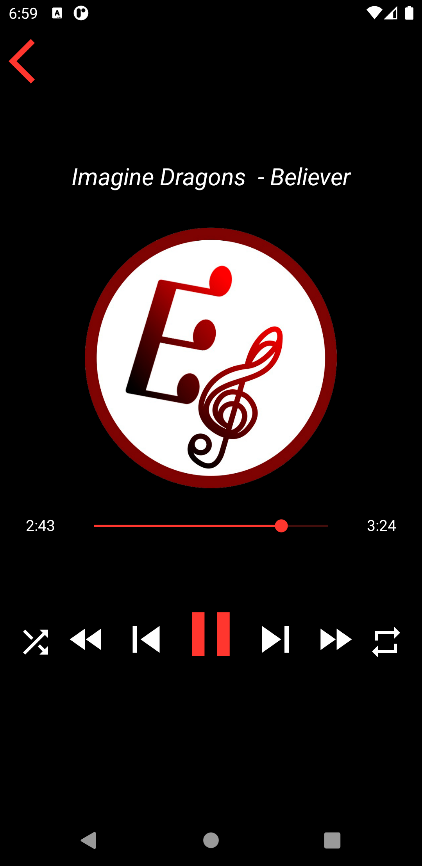


Рисунок 14 – результаты теста 21

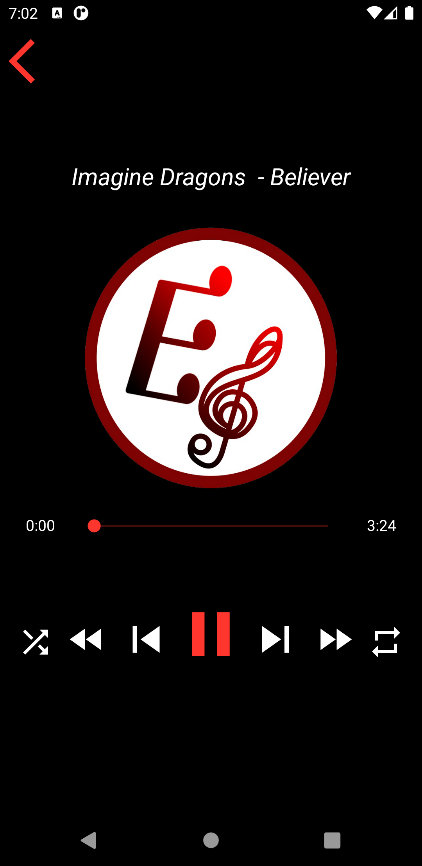
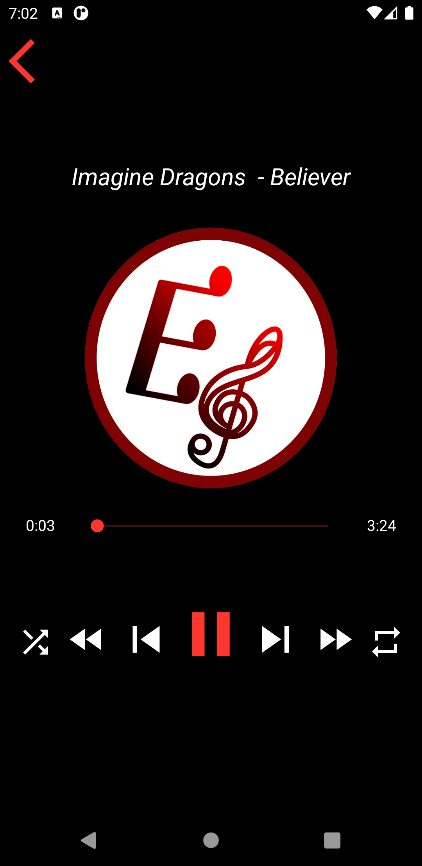


Рисунок 15 – результаты теста 22

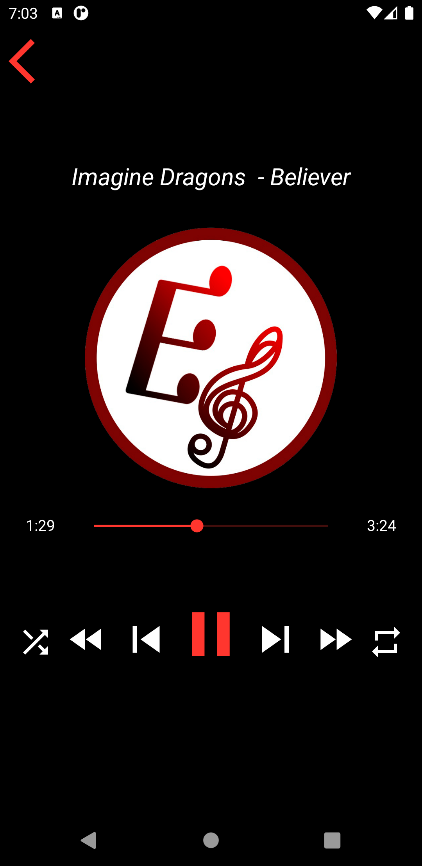
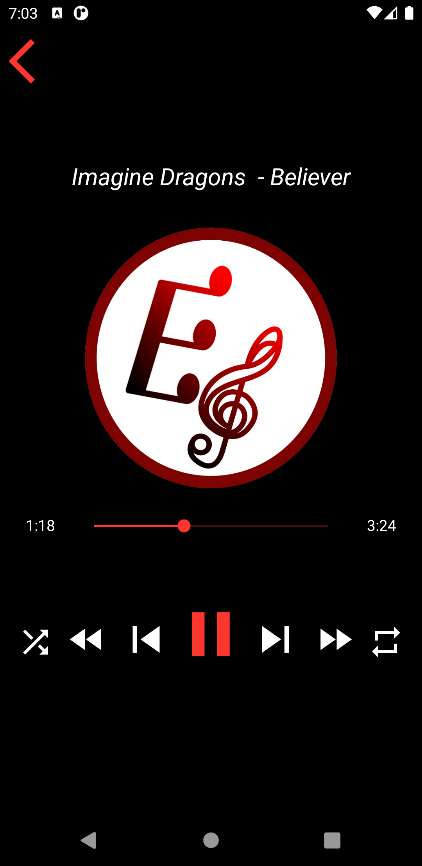


Рисунок 16 – результаты теста 23

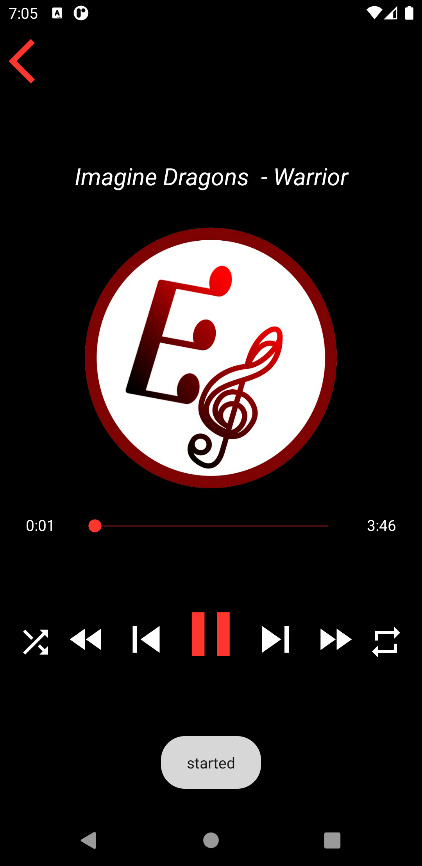
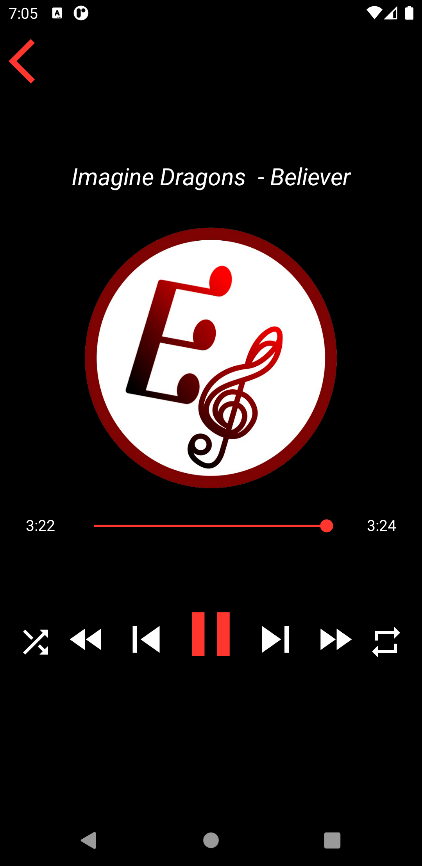


Рисунок 17 – результаты теста 24

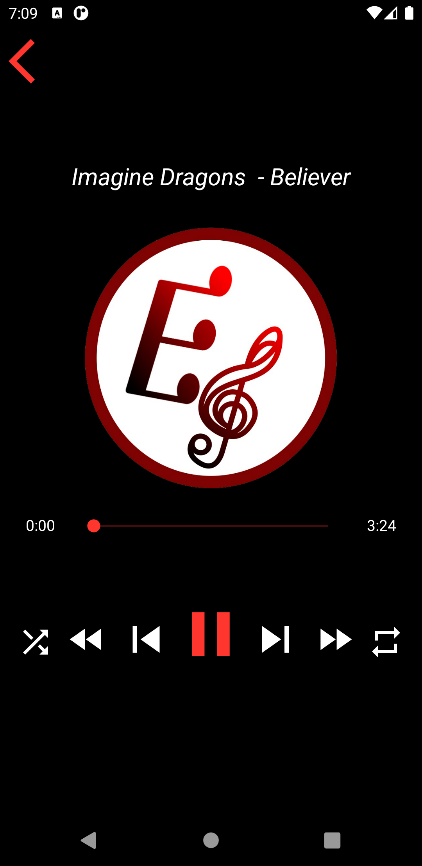
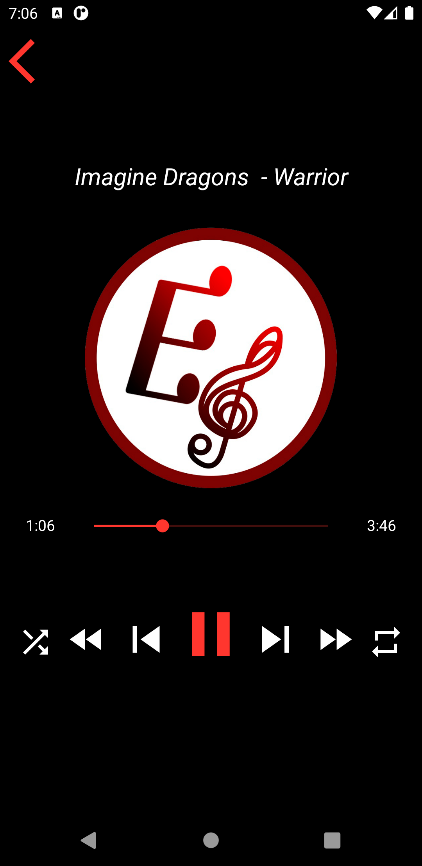


Рисунок 18 – результаты теста 25

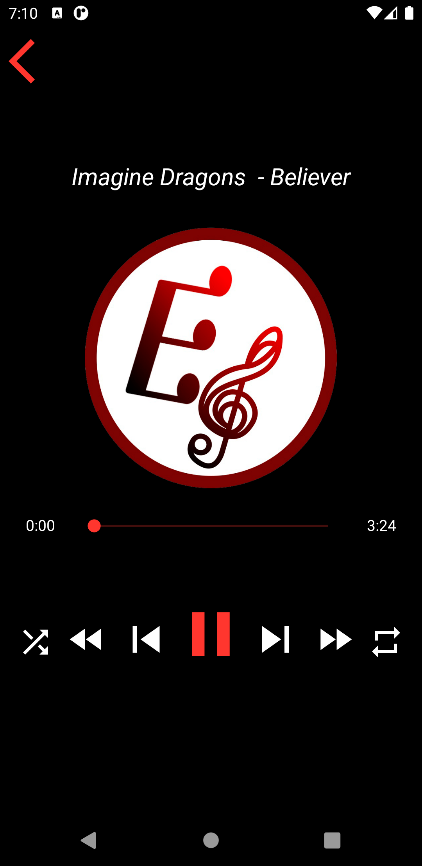
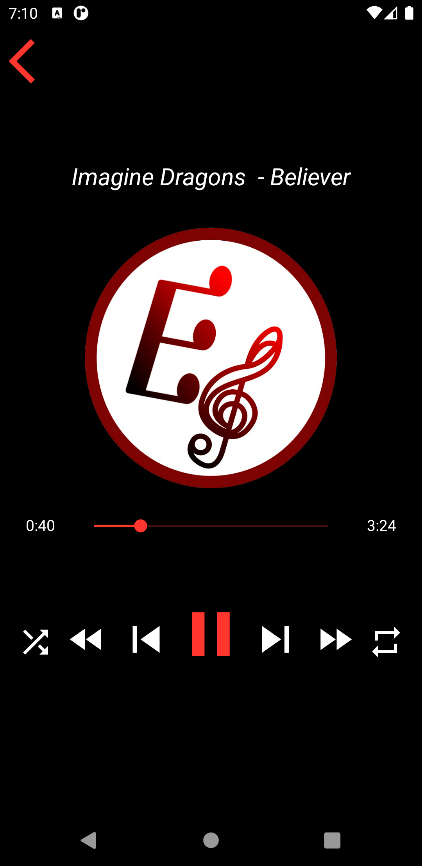


Рисунок 19 – результаты теста 26

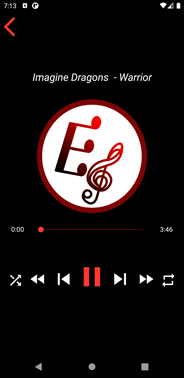
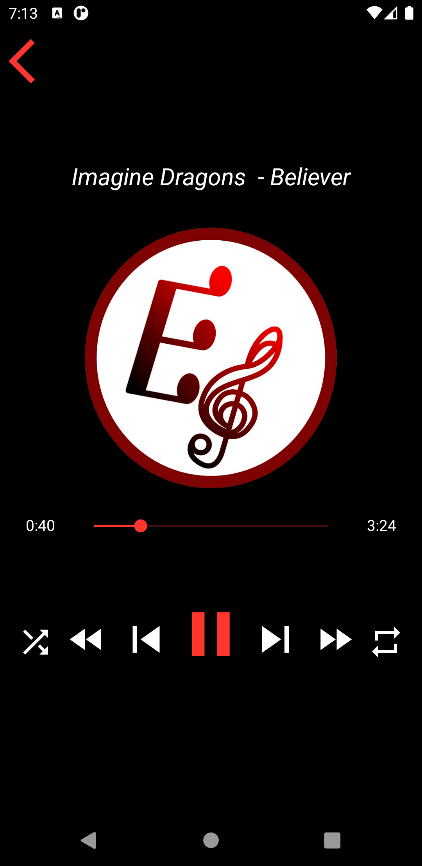


Рисунок 20 – результаты теста 27

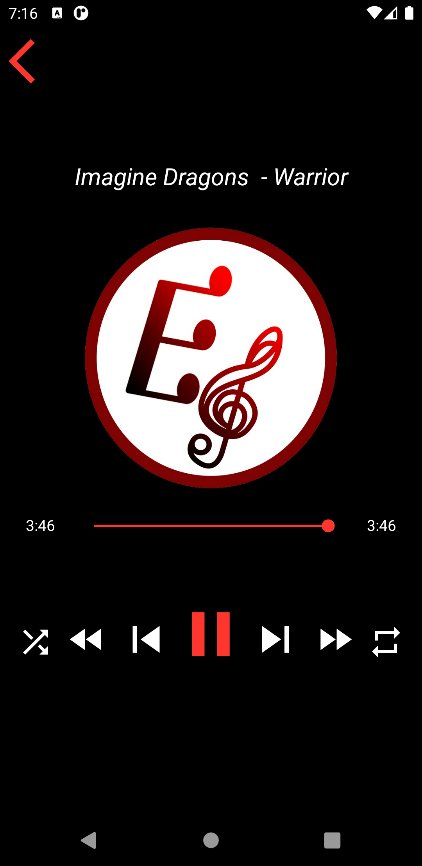
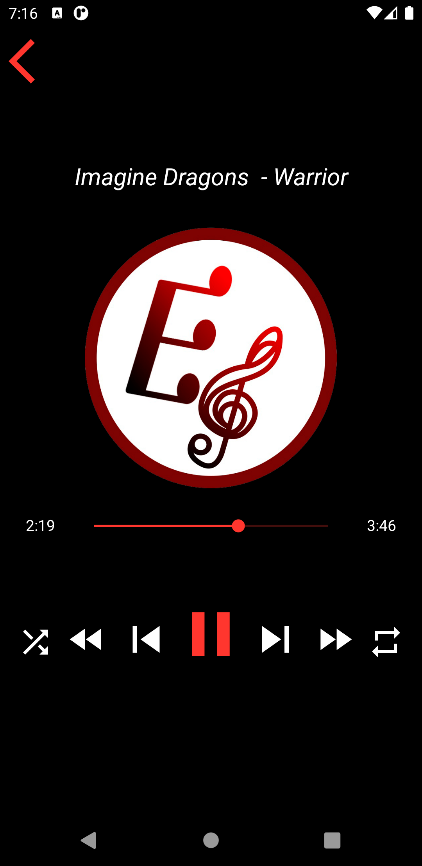


Рисунок 21 – результаты теста 28

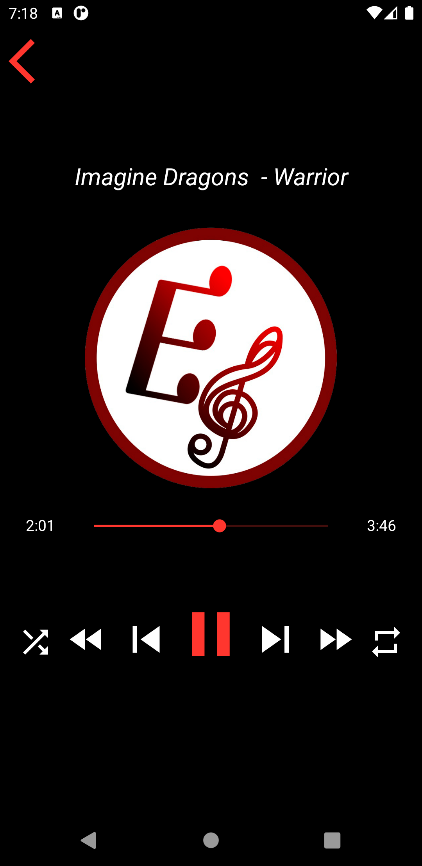
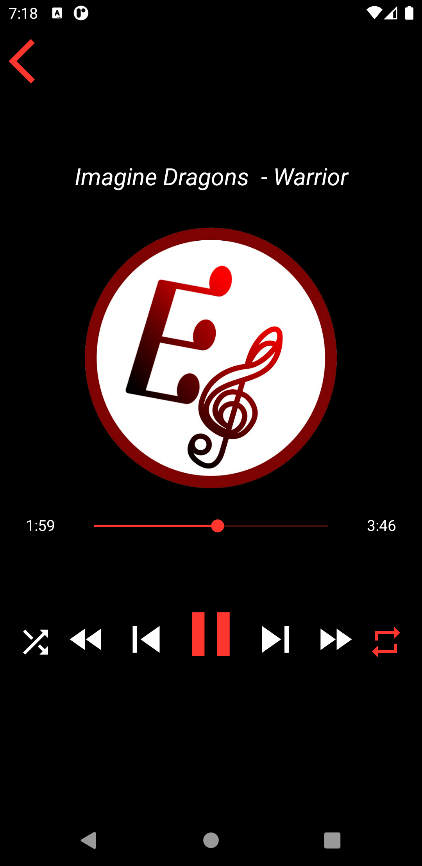


Рисунок 22 – результаты теста 29

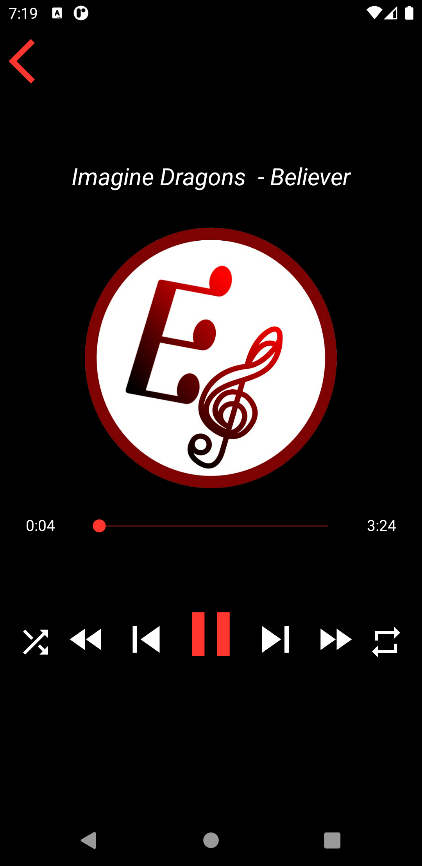
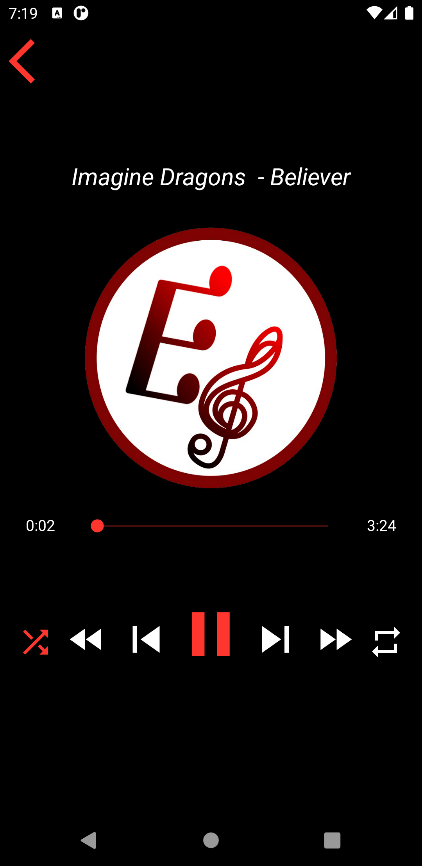


Рисунок 23 – результаты теста 30

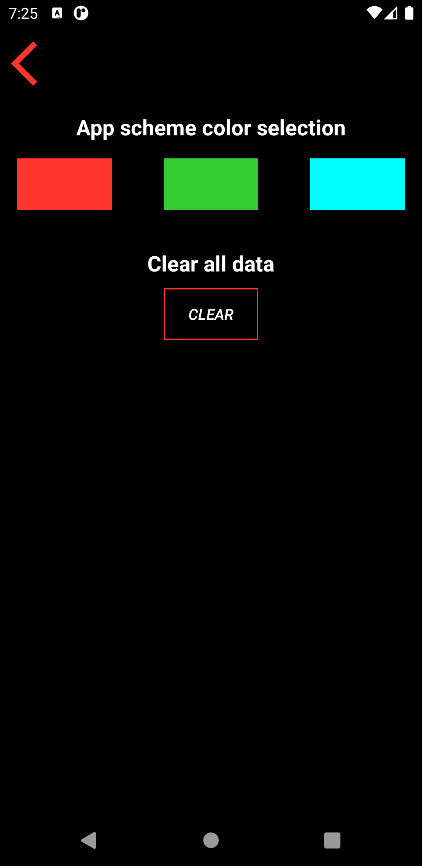
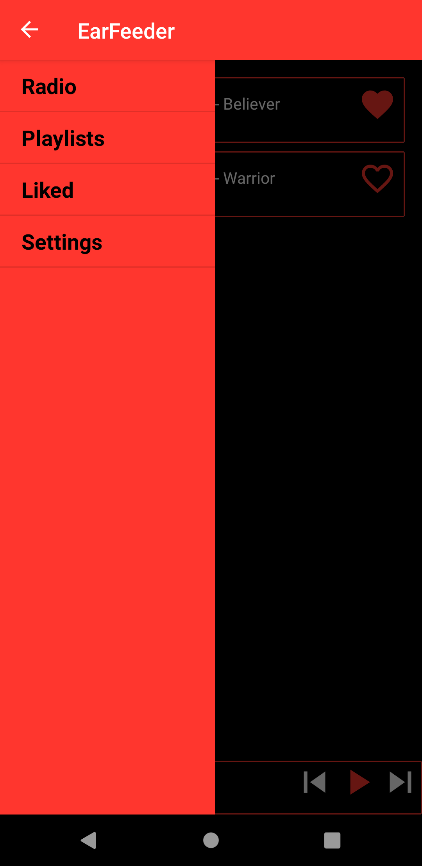


Рисунок 24 – результаты теста 31

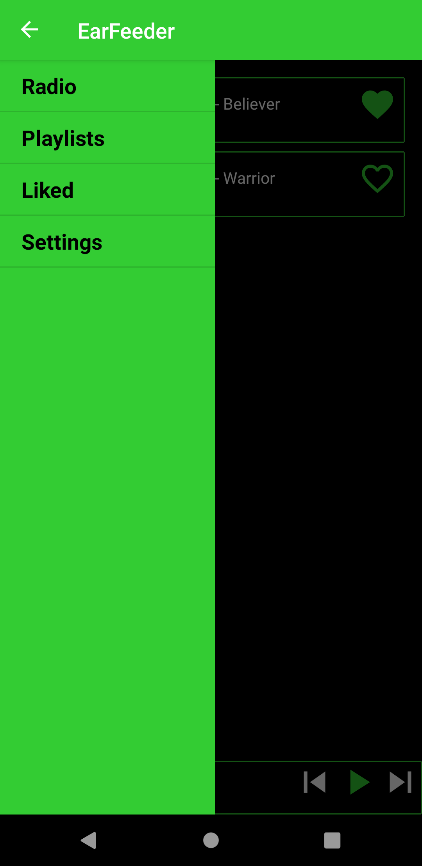
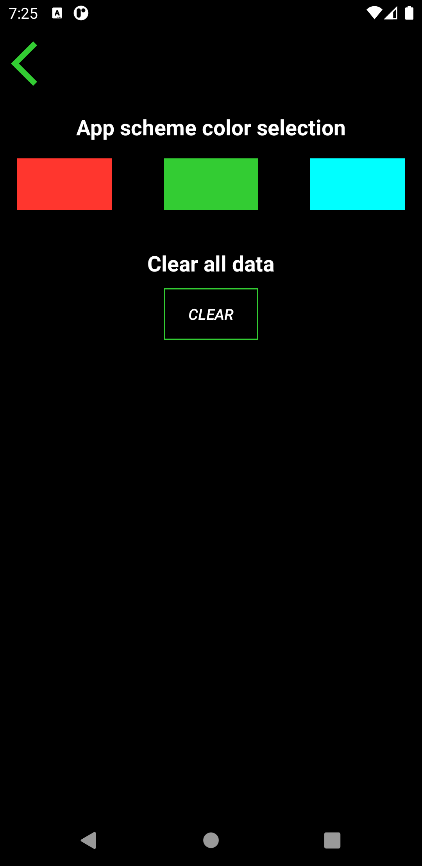


Рисунок 25 – результаты теста 31

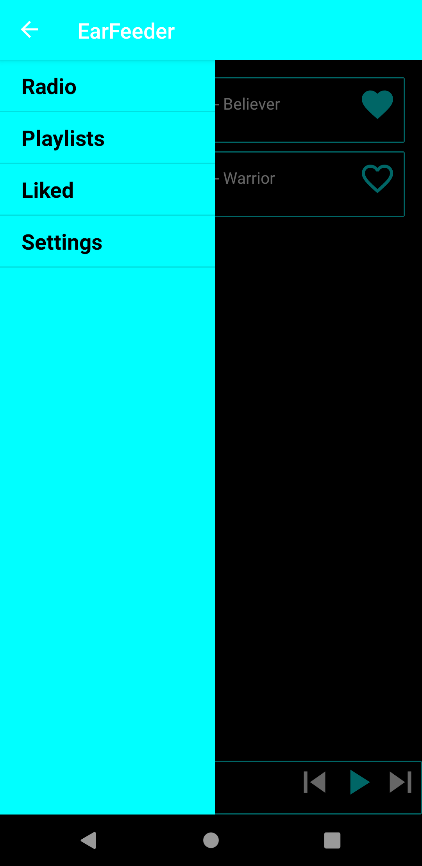
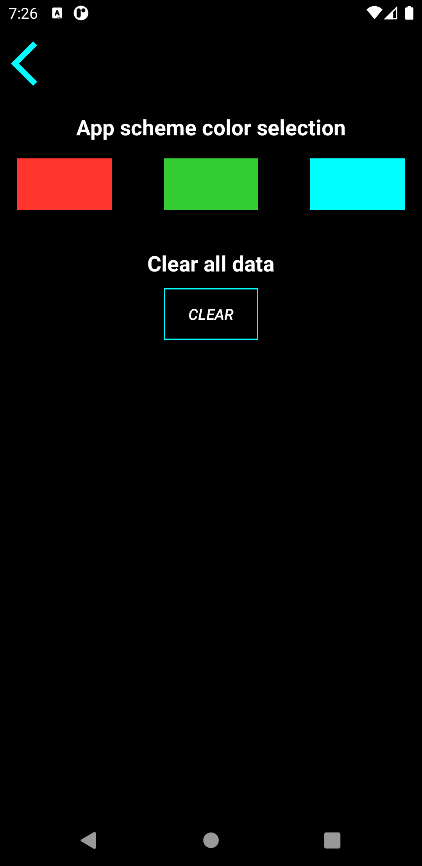


Рисунок 26 – результаты теста 31

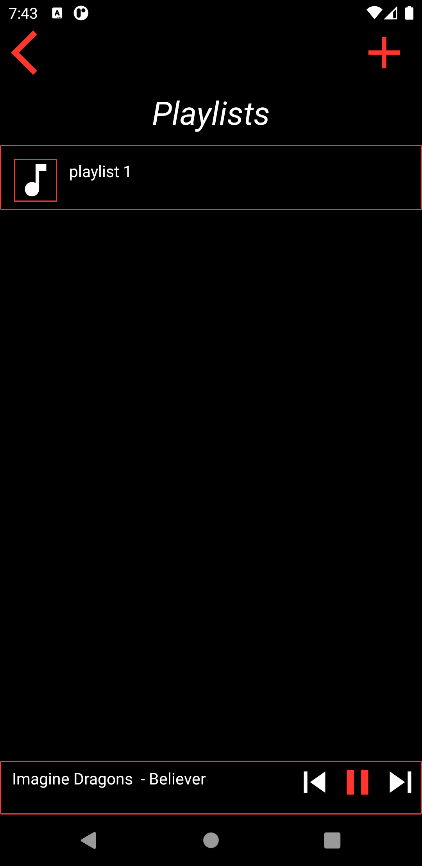
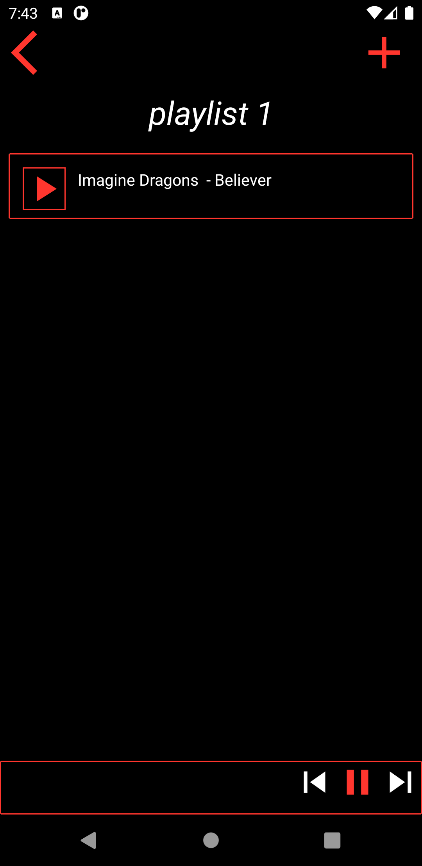


Рисунок 27 – результаты теста 36

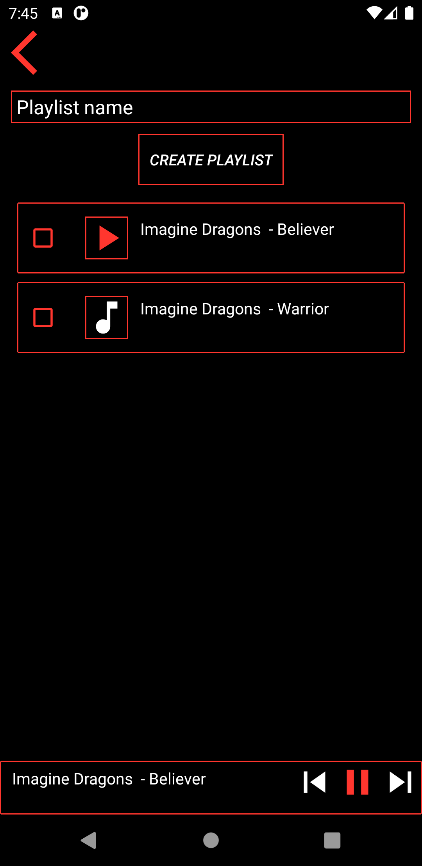


Рисунок 28 – результат теста 37

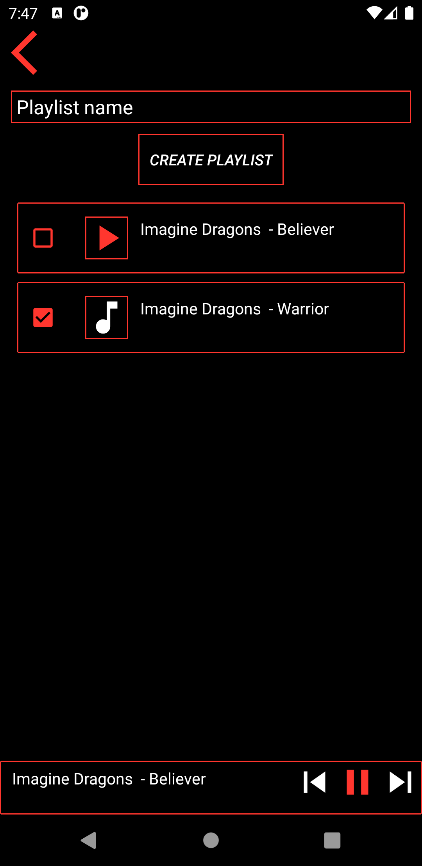


Рисунок 29 – результат теста 40

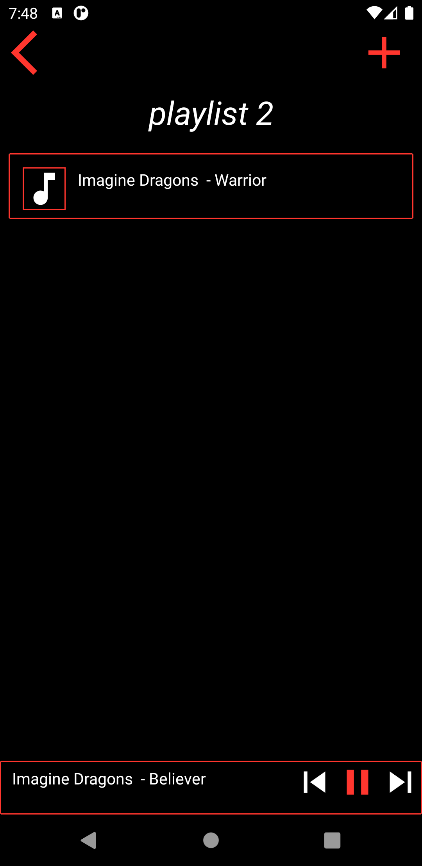
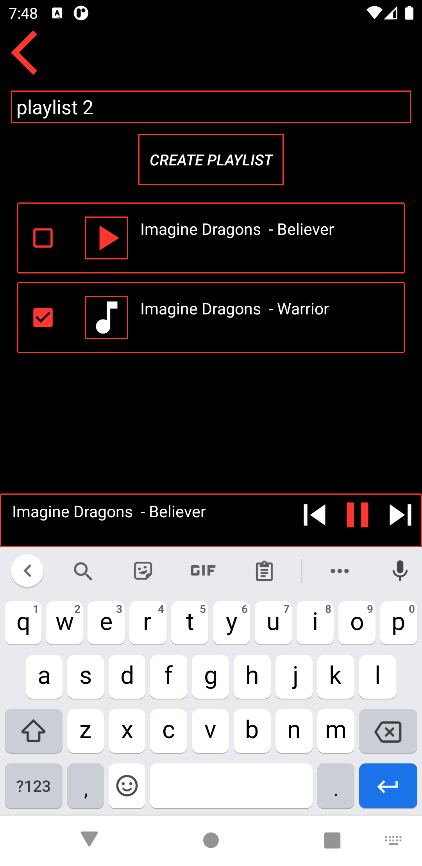


Рисунок 30 – результат теста 41

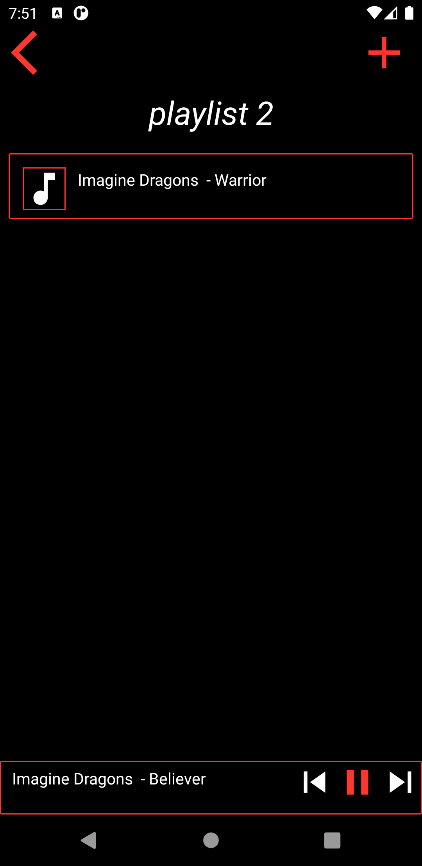
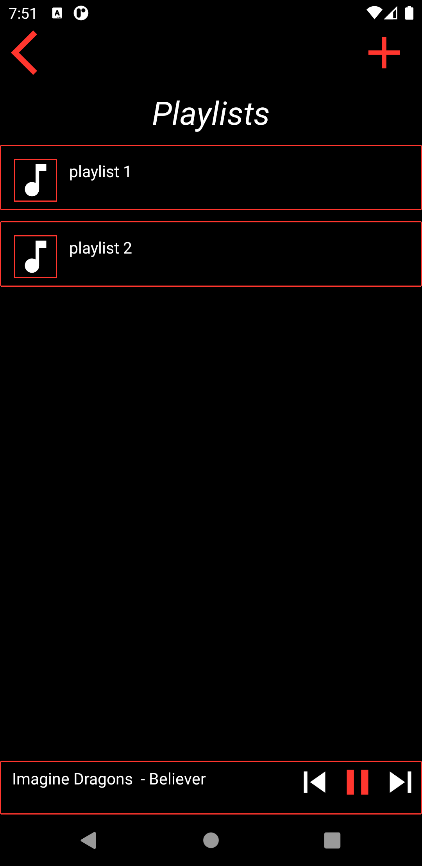


Рисунок 31 – результат теста 44

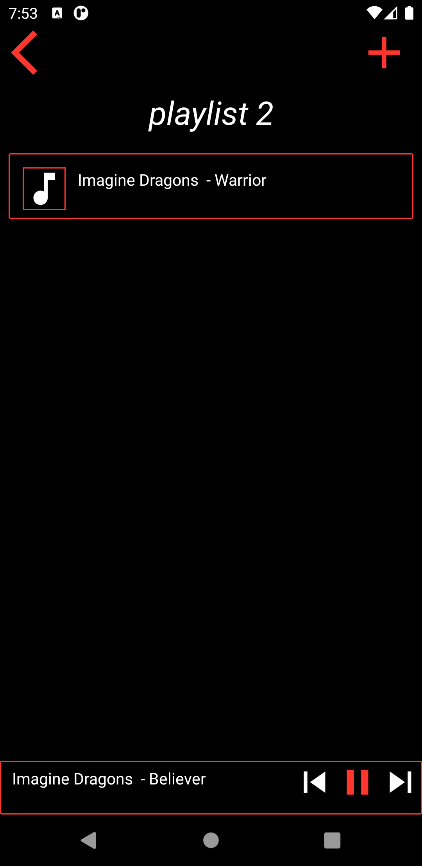
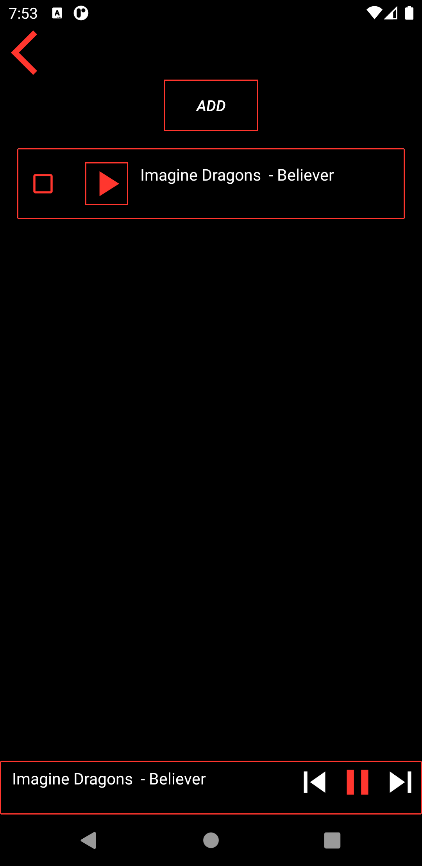


Рисунок 32– результат тестов 45 и 46

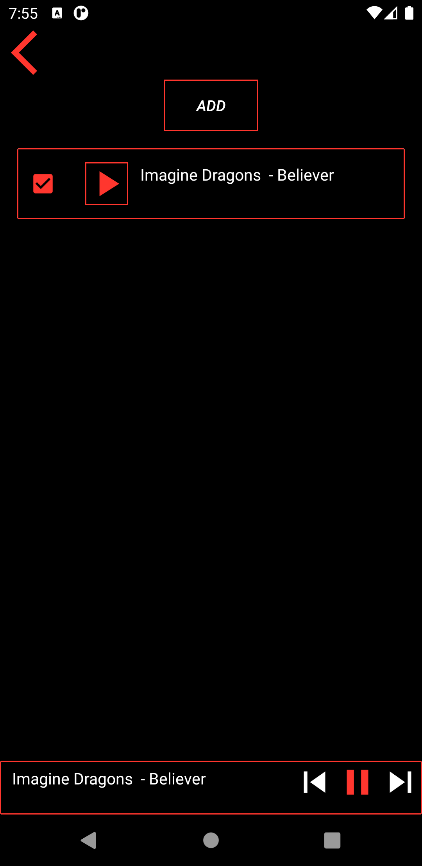
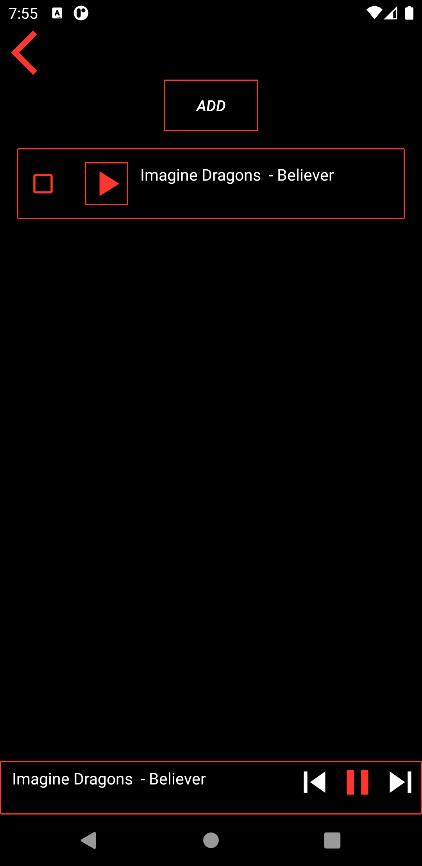


Рисунок 33 – результаты теста 48

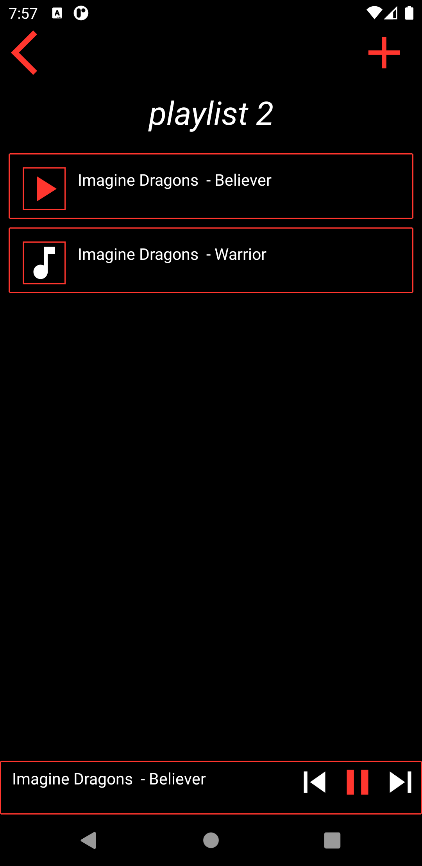
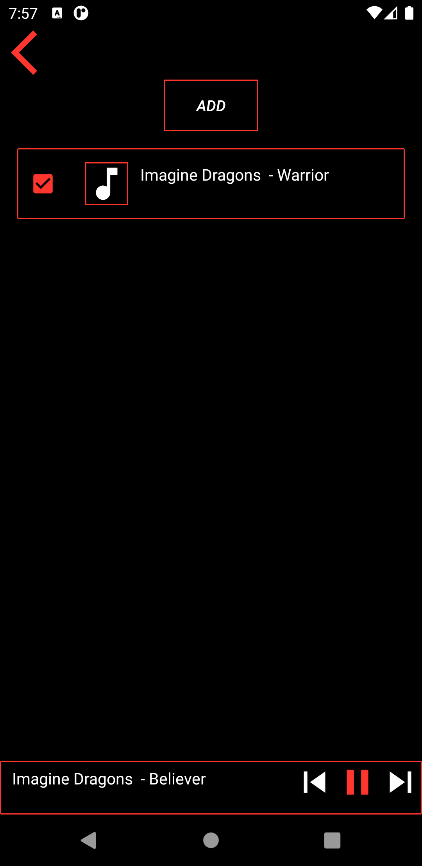


Рисунок 34 – результат теста 49

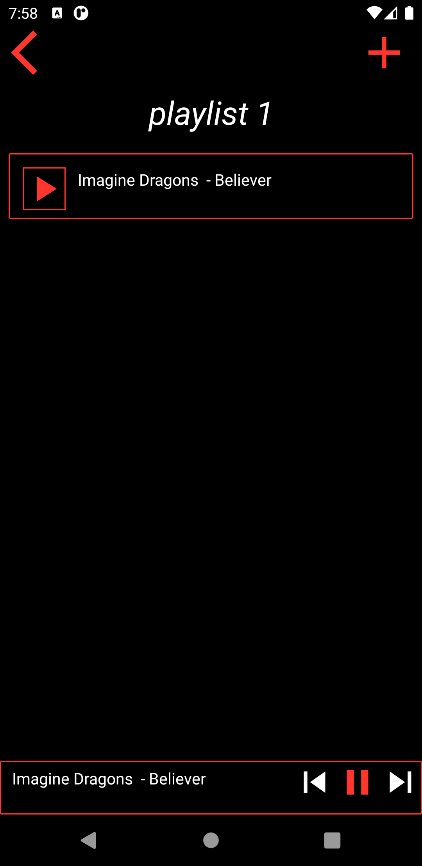
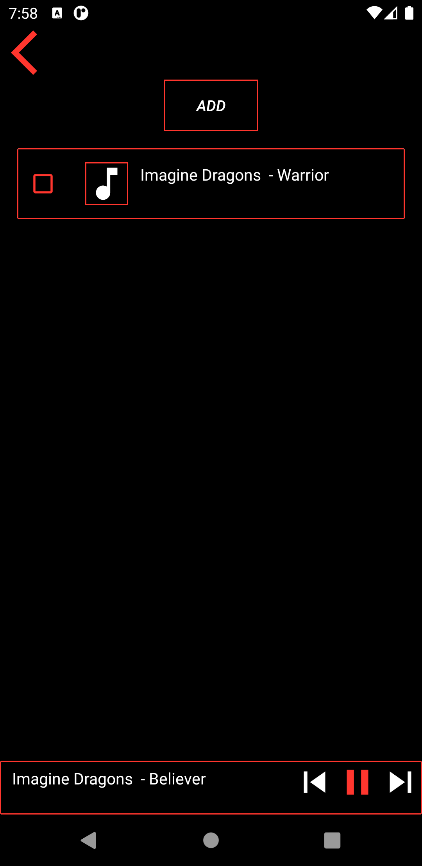


Рисунок 35 – результат теста 50

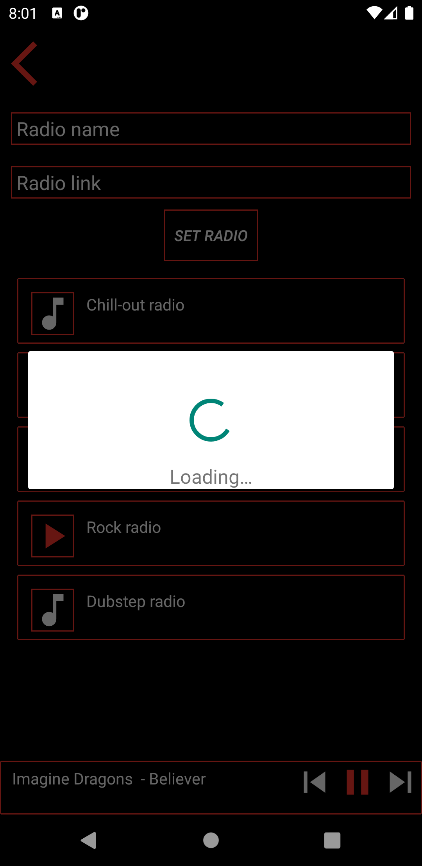
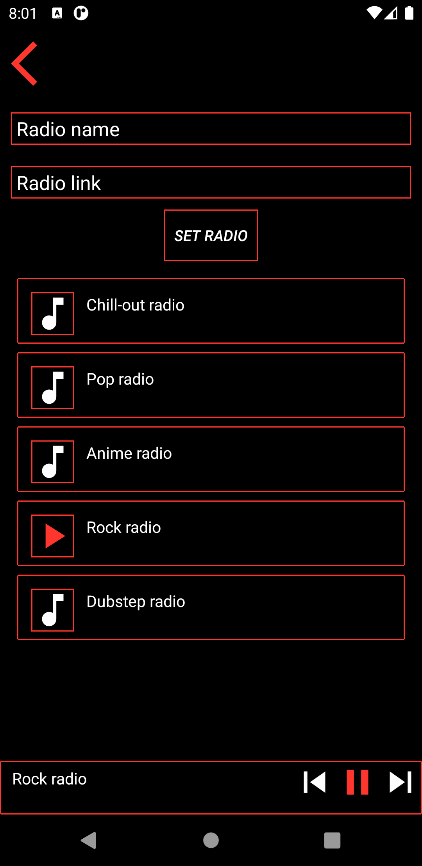


Рисунок 36 – результат теста 51

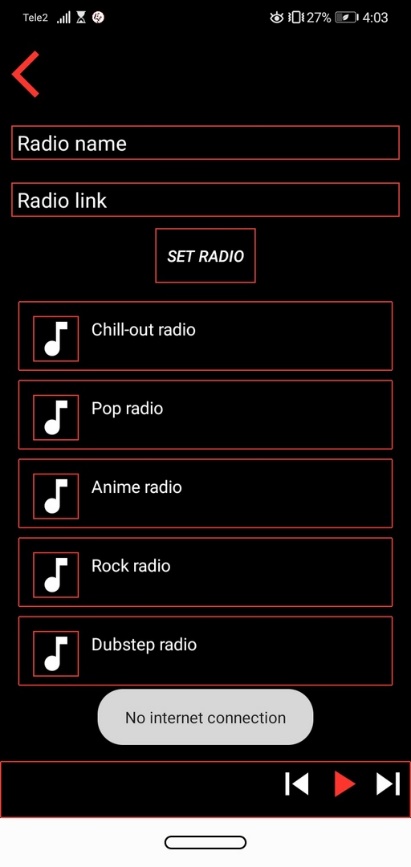


Рисунок 37 – результат теста 52

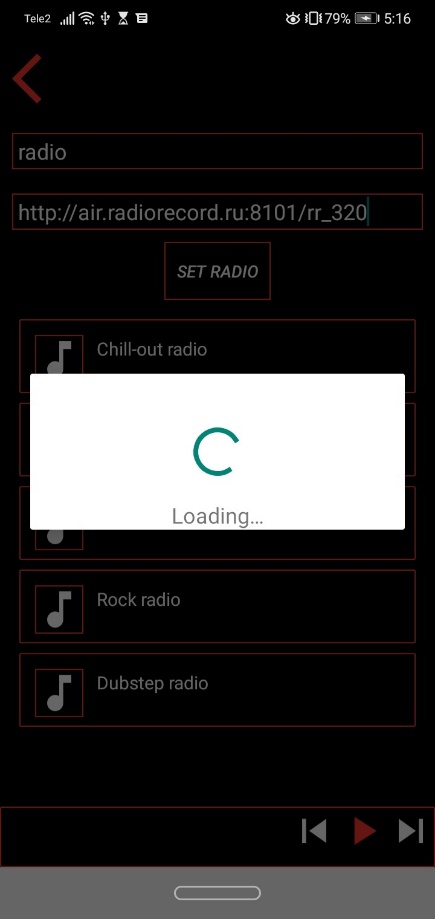
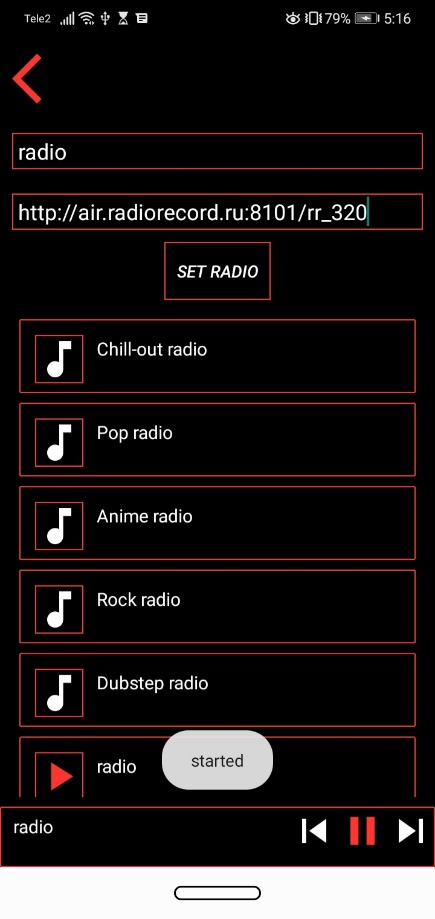


Рисунок 38– результат теста 54

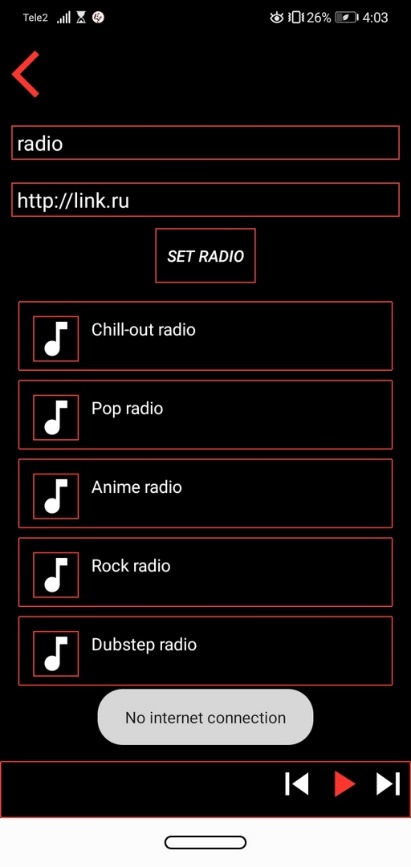


Рисунок 39 – результат теста 55

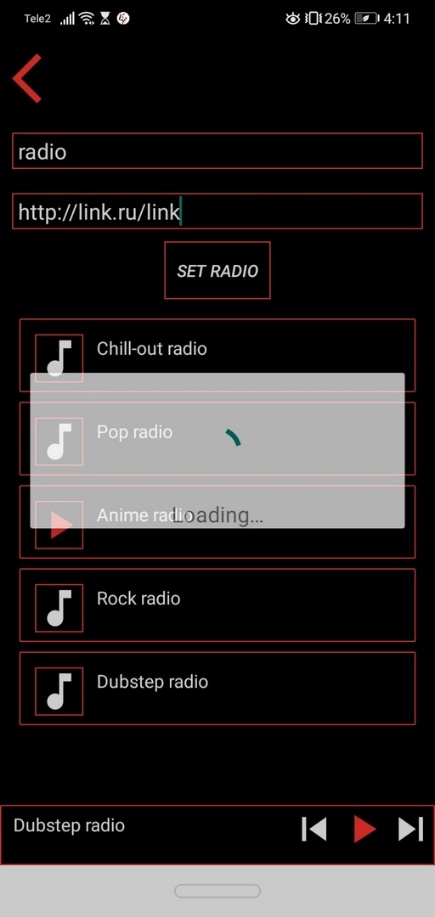
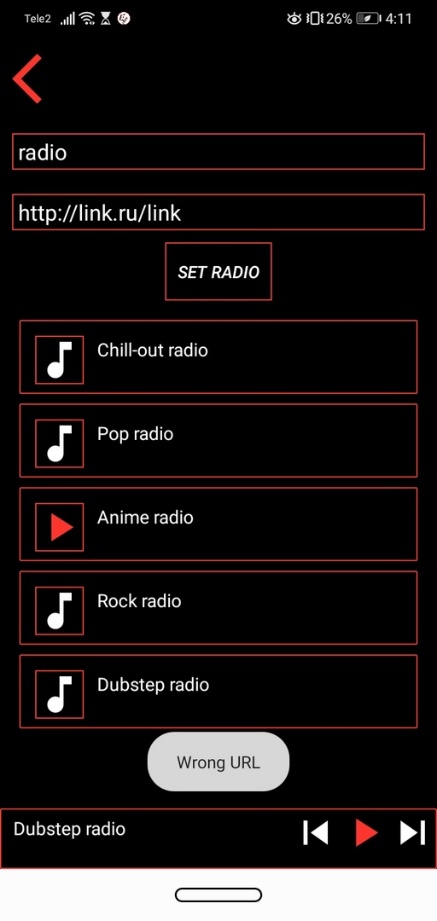


Рисунок 40 – результат теста 56

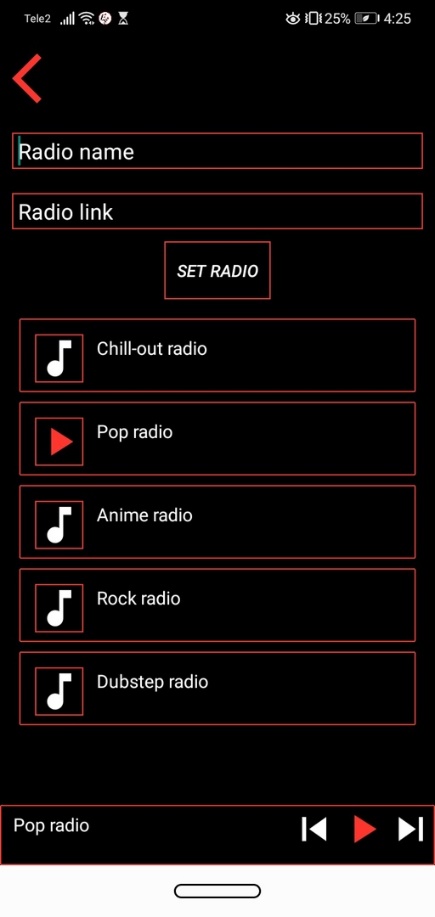
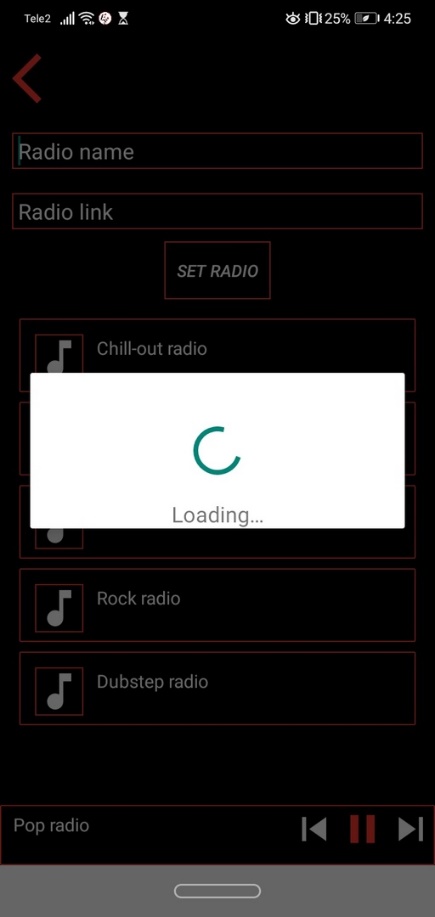
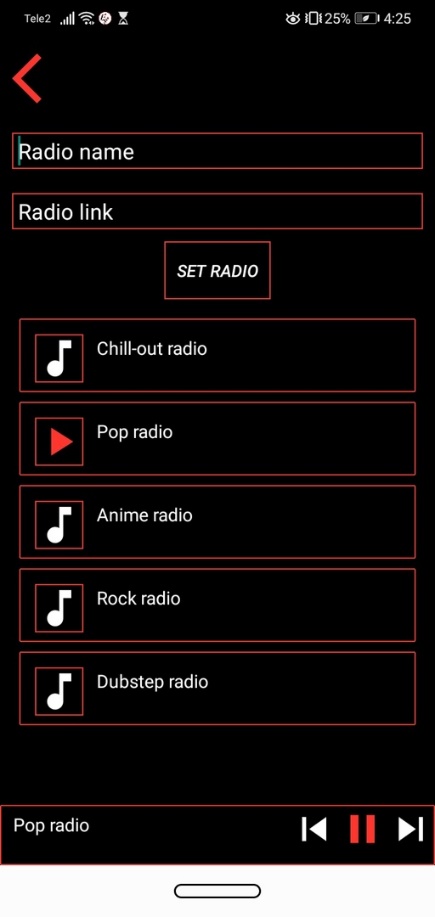
  

Рисунок 41– результат теста 57

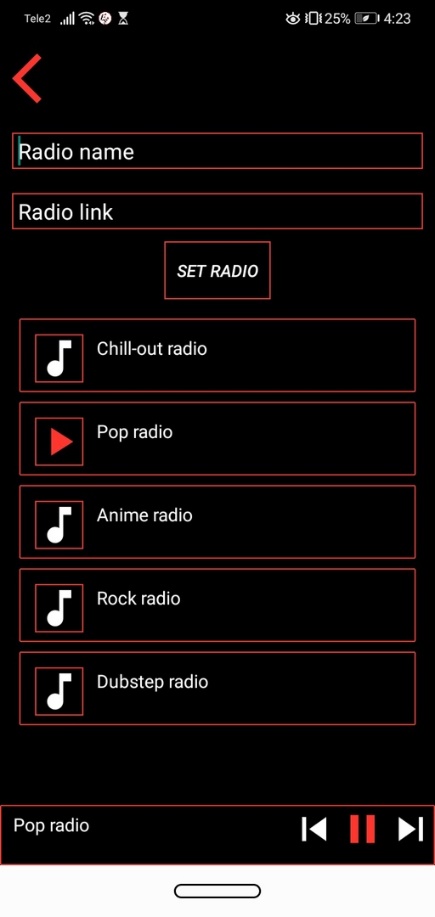
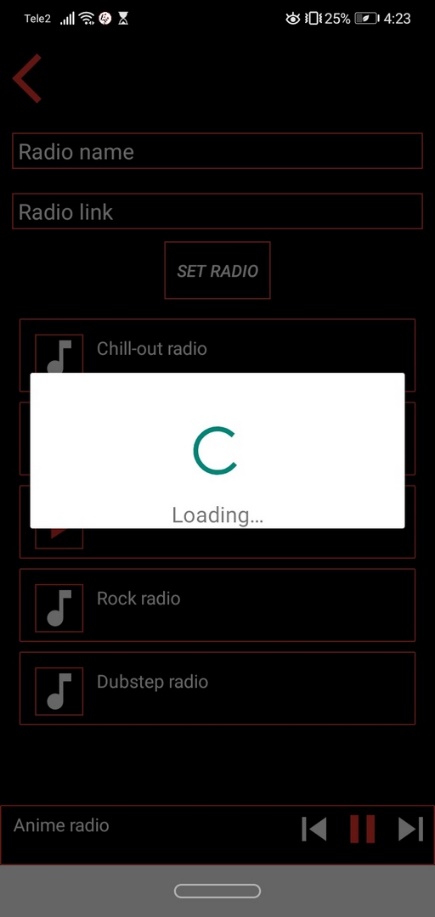
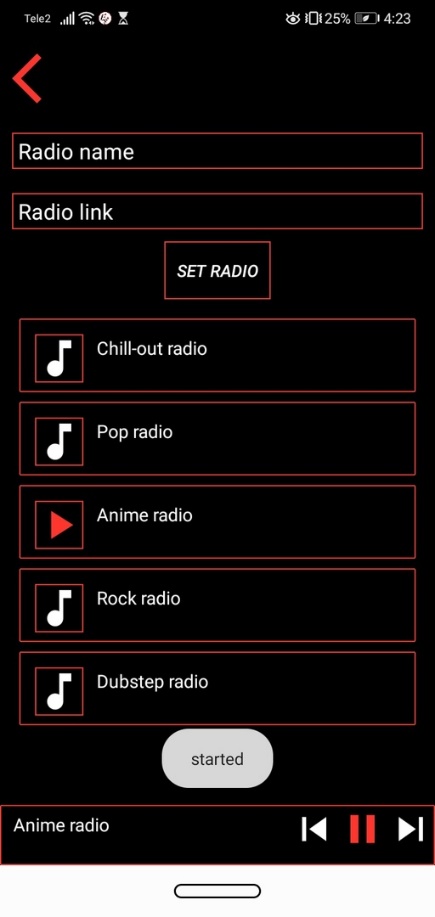
  

Рисунок 42– результат теста 58

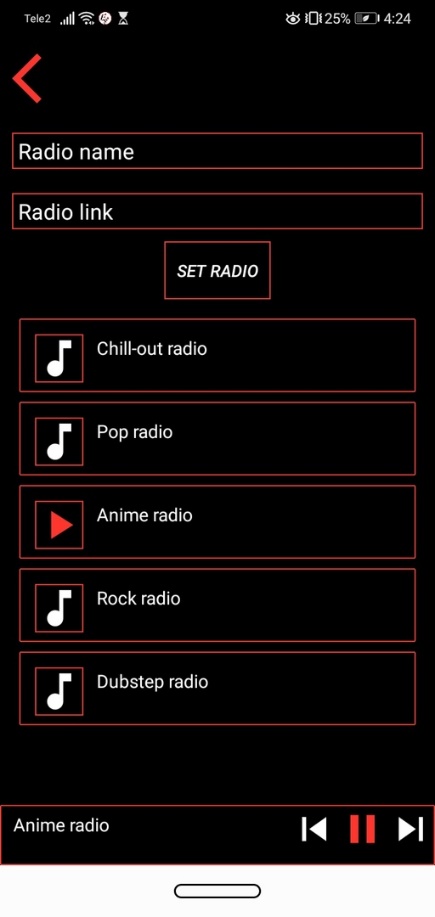
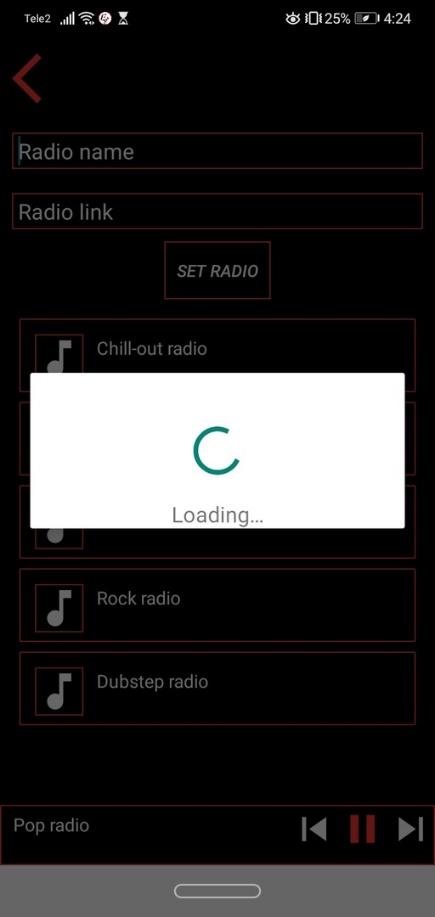
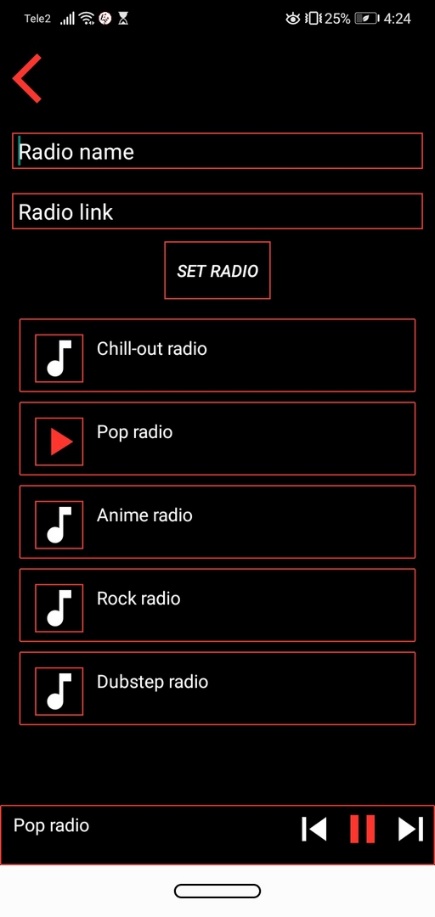
  

Рисунок 43 – результат теста 60

# Заключение

В процессе разработки данного приложения были получены и закреплены следующие навыки:

1. Проектирования и применения объектного подхода в реализации системы
2. Проектирования пользовательского интерфейса и компоновке элементов взаимодействия с пользователем
3. Создания и проектирования локальных баз данных с помощью СУБД SQLite
4. Программирования под среду Android (в том числе: работа с внутренними файлами телефона, потоки, сервисы, фрагменты, уведомления и http-запросы)
5. Применения необходимых видов паттернов программирования (singleton, DAO, observer)

В ходе дипломного проектирования были решены следующие задачи:

1. Выработаны функциональные требования приложения
2. Спроектированы интерфейс, классы и локальная база данных
3. Разработано приложение согласно реализованным требованиям и проектированию
4. Приложение было протестировано с помощью приемочного тестирования
5. Составлена документация по всей проделанной работе

Исходный код приложения опубликован в публичном репозитории на сервисе GitHub [10].

Цель курсового проекта достигнута.

Список использованных источников

1. Head first. Паттерны программирования. Эрик Фримен, Элизабет Робсон [2020] (дата обращения: 15.04.2021)
2. Сборник ссылок для радио [электронный ресурс] // URL: <https://mc-terra.ru/index.php?topic=7771.0> (дата обращения: 12.04.2021)
3. Start Android [электронный ресурс] // учебник по андроид для начинающих и продвинутых, URL: <https://startandroid.ru/ru/> (дата обращения: 04.04.2021)
4. Java. Экспресс-курс [электронный ресурс] // Сайт Александра Климова [сайт], URL: http :// developer.alexanderklimov . ru /android / java / java. php (дата обращения: 04.04.2021)
5. API Specification for the Java 7 SE. [официальный сайт] URL: http :// docs. oracle. com / javase /7/ docs /api / (дата обращения: 05.04.2021)
6. Отношения классов — от UML к коду [Электронный ресурс] // Хабрахабр: [сайт]. [2006]. URL: http://habrahabr.ru/post/150041/ (дата обращения: 20.05.2021)
7. Что такое ORM [Электронный ресурс]. [2017]. URL: <https://myrusakov.ru/what-is-it-orm-php.html> (дата обращения: 27.05.2021)
8. Data Access Object (DAO) [Электронный ресурс] // Хабрахабр: [сайт]. [2015]. URL: <https://habr.com/ru/post/262243/> (дата обращения: 27.05.2021)
9. Одиночка (Singleton) | Паттерны [Электронный ресурс] // Метанит: [сайт]. [2018]. URL: <https://metanit.com/sharp/patterns/2.3.php>
10. Данный проект [Электронный ресурс] // GitHub: [сайт]. URL: <https://github.com/MAL1S/MusicPlayer.git>