**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 2](#_Toc168965776)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc168965777)

[2 ОБЗОР АНАЛОГОВ ПРИЛОЖЕНИЯ 5](#_Toc168965778)

[3 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 6](#_Toc168965779)

[4 ОСНОВНАЯ ЧАСТЬ 8](#_Toc168965780)

[4.1 АРХИТЕКТУРА ПРОГРАММНОЙ СРЕДЫ 8](#_Toc168965781)

[4.2 РЕАЛИЗАЦИЯ КОДА ПРОГРАММЫ НА ЯЗЫКЕ JAVA 10](#_Toc168965782)

[4.2.1 СТРУКТУРА ОБЪЕКТОВ 10](#_Toc168965783)

[4.2.2 СПОЛЬЗОВАННЫЕ ТЕХНОЛОГИИ 11](#_Toc168965784)

[4.2.3 РЕАЛИЗОВАННЫЙ ФУНКЦИОНАЛ 11](#_Toc168965785)

[4.2.4 УЛУЧШЕНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ОПЫТА 11](#_Toc168965786)

[4.3 ХРАНЕНИЕ ДАНЫХ 13](#_Toc168965787)

[4.4 ОБЕСПЕЧЕНЕНИЕ БЕЗОПАСНОСТИ ИНФОРМАЦИИ 16](#_Toc168965788)

[4.5 СИСТЕМА КОНТРОЛЯ ВЕРСИЙ 17](#_Toc168965789)

[5 WIREFRAME ПРИЛОЖЕНИЯ 18](#_Toc168965790)

[6 ПРИМЕР РАБОТЫ ПРИЛОЖЕНИЯ 20](#_Toc168965791)

[7 ИНСТРУКЦИЯПОЛЬЗОВАТЕЛЮ 23](#_Toc168965792)

[ЗАКЛЮЧЕНИЕ 24](#_Toc168965793)

[СПИСОК ЛИТЕРАТУРЫ 25](#_Toc168965794)

[ПРИЛОЖЕНИЕ 26](#_Toc168965795)

# **ВВЕДЕНИЕ**

В современном мире поток информации постоянно растет, и доступ к ней становится все более важным и актуальным. Одним из важных источников информации являются данные о авиарейсах, которые необходимы для планирования работы аэропорта и отслеживания текущих рейсов. В этом контексте разработка информационно-справочной системы "Авиарейсы аэропорта" становится весьма значимой.

Цель курсовой работы: разработать мобильное приложение – информационно-справочную систему "Авиарейсы аэропорта" средствами языка Java на платформе Android, предоставляющую пользователям удобный доступ к информации о рейсах, их расписании, статусе и другой сопутствующей информации.

Разработка мобильного приложения "Авиарейсы аэропорта" на языке Java позволит пользователям получать необходимую информацию о авиарейсах непосредственно на своих мобильных устройствах.

1) проанализировать предметную область мобильного приложения;

2) сделать обзор существующих аналогов разрабатываемого приложения;

3) сформировать техническое задание на разработку программы в соответствии с ГОСТ 19.201-78;

4) описать архитектуру программной системы, привести структурную и функциональную диаграммы, схему базы данных;

5) спроектировать интерфейс мобильного приложения;

6) реализовать код программы на языке высокого уровня Java, протестировать его и отладить;

7) реализовать контрольный пример работы программы, начиная с открытия, показать все этапы работы вашего приложения.

Структура пояснительной записки: введение, 7 пунктов, заключение, список из 9 используемых источников, 50 приложений.

# **1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

В современной авиации с развитием технологий возникают и новые вызовы. Безопасность и надежность информации становятся все более критичными аспектами в свете угроз кибербезопасности и требований к оперативности данных. Для обеспечения безопасности пассажиров и персонала аэропортов необходимо учитывать аспекты защиты данных, а также методы предотвращения и быстрого реагирования на возможные сбои в системе.

Важно также учитывать управление информацией о рейсах. Обеспечение точности данных, своевременное обновление расписания и статусов рейсов, а также интеграция с другими информационными системами аэропорта — все это ключевые аспекты эффективного управления информацией о авиарейсах.

Мобильное приложение "Авиарейсы аэропорта" предоставляет пользователям доступ к информации о различных авиарейсах, их расписании, статусе вылетов и прилетов. Пользователи могут просматривать детальную информацию о каждом рейсе, получать уведомления о изменениях статуса и времени вылета/прилета, а также искать рейсы по различным критериям.

Для информационно-справочной системы создается база данных "Авиарейсы", которая содержит следующие данные:

* Сведения о рейсах (номер рейса, пункт вылета и назначения, время вылета и прилета).
* Статусы рейсов.
* Логотипы авиакомпаний.

С базой данных могут работать администраторы, для которых доступны следующие задачи:

* Добавление информации о новых рейсах в систему.
* Обновление и корректировка информации о текущих рейсах.
* Удаление устаревшей информации о рейсах.

В курсовой работе информационная модель данных предметной области представлена с помощью ER-диаграммы.

В данной предметной области, бизнес-процессы, в которых будет задействована программа, включают управление информацией о рейсах, добавление новой информации, обновление текущих данных, а также анализ и использование существующей информации. Эти процессы взаимосвязаны и вместе формируют цикл управления информацией в системе. Детальное представление этих процессов можно увидеть на представленной диаграмме IDEF0 в приложении к курсовой работе №1.

Вывод: Информационно-справочные системы, такие как "Авиарейсы аэропорта", представляют собой необходимый инструмент в современном мире, обеспечивая доступ к широкому спектру данных и информации. Они упрощают процесс получения информации, повышают эффективность работы и способствуют принятию обоснованных решений, что особенно важно в условиях высокой динамики авиационного транспорта.

# **2 ОБЗОР АНАЛОГОВ ПРИЛОЖЕНИЯ**

На рынке Android-приложений отсутствуют прямые аналоги приложения, специализированного для информации о авиарейсах аэропорта. Существующие аналоги, такие как FlightAware, Flightradar24 и Google Flights, не являются полностью специализированными для предоставления детальной информации о всех аспектах авиарейсов в конкретном аэропорту и имеют ряд недостатков.

Пользователи отмечают, что в подобных приложениях иногда сложно ориентироваться из-за избытка информации и большого количества дополнительных функций, которые могут быть не всегда необходимы для конкретного пользователя. Также возникают проблемы с навигацией, фильтрацией и сравнением рейсов. Ограниченные возможности поиска и отсутствие персонализированных рекомендаций также являются значительными недостатками существующих платформ.

В целом, отзывы пользователей указывают на необходимость специализированного приложения, которое бы предлагало удобный интерфейс, интуитивно понятную навигацию и надежное взаимодействие с приложением даже без доступа в интернет. Такое приложение могло бы значительно улучшить опыт получения информации о авиарейсах, обеспечивая пользователей точными и своевременными данными.

Вывод

Существующие аналоги не удовлетворяют полностью потребности пользователей из-за недостатков в навигации, поиске и персонализации. Разработка специализированного приложения "Авиарейсы аэропорта" имеет смысл для улучшения опыта пользователей и обеспечения им более удобного доступа к необходимой информации о рейсах. Кроме того, данное приложение может иметь потенциал для сотрудников аэропорта.

# **3 ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

1. Требования к программе или программному изделию
2. Требования к функциональным характеристикам:
   * Регистрация и авторизация пользователя.
   * Отображение описания рейсов.
   * Предоставление списка рейсов.
   * Возможность добавление информации о рейсах администратором.
   * Отображение динамического статуса рейсов в реальном времени.
3. Требования к надежности:
   * + Приложение должно включать механизмы защиты и ролевую модель с использованием аутентификации и авторизации.
     + Приложение должно обеспечивать бесперебойный доступ ко всем функциям.
     + Приложение должно иметь механизмы контроля целостности данных для предотвращения их изменения или подделки.
4. Требования к составу и параметрам технических средств:

Приложение должно быть доступно для использования на мобильных устройствах с операционной системой Android.

1. Требования к маркировке и упаковке:

Приложение должно иметь адаптивную иконку.

1. Технико-экономические показатели:

Ориентировочная экономическая эффективность разработки приложения оценивается как положительная, учитывая следующие факторы:

1.Приложение предназначено для активного использования широким кругом пользователей, что обеспечит стабильный и постоянный спрос.

2. Приложение предлагает современные и удобные функции, что делает его более привлекательным для пользователей по сравнению с существующими аналогами. Разработка приложения будет более эффективной и экономичной благодаря использованию современных технологий и методов разработки.

# **4 ОСНОВНАЯ ЧАСТЬ**

## **4.1 АРХИТЕКТУРА ПРОГРАММНОЙ СРЕДЫ**

Архитектура программной системы включает несколько ключевых аспектов:

Модульная структура: Система состоит из различных модулей, каждый из которых выполняет конкретные функции или задачи приложения. Такой подход облегчает разработку, тестирование и обслуживание системы.

Клиент-сервер: в рамках мобильного приложения клиентская часть взаимодействует с серверной для получения данных, обновлений и других ресурсов. Это позволяет добиться масштабируемости, удобства обновления и централизованного управления данными.

Пользовательский интерфейс: Архитектура включает компоненты, отвечающие за отображение пользовательского интерфейса, обработку действий пользователя и взаимодействие с серверной частью приложения.

Серверная часть: Серверная часть отвечает за выполнение бизнес-логики, работу с базой данных, аутентификацию и авторизацию пользователей, а также обработку запросов от клиентской части.

База данных: Система использует базу данных Firebase для хранения и управления данными. Это решение обусловлено требованиями и особенностями приложения, обеспечивая гибкость и масштабируемость.

Коммуникация: для взаимодействия между компонентами системы используются API, сетевые протоколы и сообщения, что обеспечивает эффективную передачу данных и синхронизацию.

Безопасность: Архитектура включает механизмы защиты данных, аутентификацию и авторизацию пользователей, обработку ошибок и предотвращение уязвимостей, что обеспечивает высокий уровень безопасности.

Масштабируемость и производительность: Система спроектирована с учетом возможности масштабирования и высокой производительности при увеличении объема данных и пользовательского трафика. Также предусмотрено расширение функционала для удовлетворения будущих потребностей.

Проектирование программного обеспечения включает определение структурных компонентов системы и их взаимодействий. Результаты проектирования могут быть представлены в виде структурной схемы, дающей полное представление о проектируемой системе.

## **4.2 РЕАЛИЗАЦИЯ КОДА ПРОГРАММЫ НА ЯЗЫКЕ JAVA**

Java – это объектно-ориентированный язык программирования, поддерживающий также структурное, обобщённое и функциональное программирование, что делает его мультипарадигменным. Технология Java включает стандартизированный язык программирования и программную платформу.

Платформы Java представлены в нескольких вариантах:

* Standard Edition (SE) – предназначена для разработки локальных и небольших сетевых приложений.
* Enterprise Edition (EE) – используется для создания масштабируемых сетевых приложений корпоративного уровня.

### **4.2.1 СТРУКТУРА ОБЪЕКТОВ**

Архитектура приложения включает следующие объекты:

1. MainActivity: это основная активность приложения, которая включает в себя три фрагмента: CreatorFragment, InfoFragment, и InstructionFragment. Так же он содержит список из ListAdapter и ADD

2. CreatorFragment, InfoFragment, и InstructionFragment: это фрагменты, дающие пользователю информацию о приложении.

3. ADD: это активити для добавления новой информации.

4. ListAdapter: адаптер для отображения информации о рейсе. Содержит в себе переходы на Change, активити для изменения информации, и DetailedActivity, активити для подробного просмотра.

4. authentication и login: это отдельные активности, которые обрабатывают регистрацию пользователя, вход в систему.

Эта архитектура представляет собой типичную модель для Android-приложений, где есть одна основная активность, содержащая несколько фрагментов для различных частей пользовательского интерфейса, а также отдельные активности для обработки специфических задач. Это обеспечивает хорошую модульность и повторное использование кода.

### **4.2.2 СПОЛЬЗОВАННЫЕ ТЕХНОЛОГИИ**

Gradle, интегрированный в Android Studio, представляет собой мощный инструмент сборки, который упрощает процесс компиляции, сборки и развертывания приложений для Android. Он обеспечивает автоматизацию различных задач сборки и предлагает гибкую систему управления зависимостями и настройками проекта.

В приложении использованы современные методы решения задач, что позволяет повысить эффективность разработки и обеспечить высокое качество конечного продукта.

### **4.2.3 РЕАЛИЗОВАННЫЙ ФУНКЦИОНАЛ**

Реализована функциональность ленты, загружающей и обновляющей данные с базы данных. Также было реализовано взаимодействие с галереей для выбора изображений. Эти функции прошли тщательное тестирование и отладку, чтобы гарантировать их корректную работу.

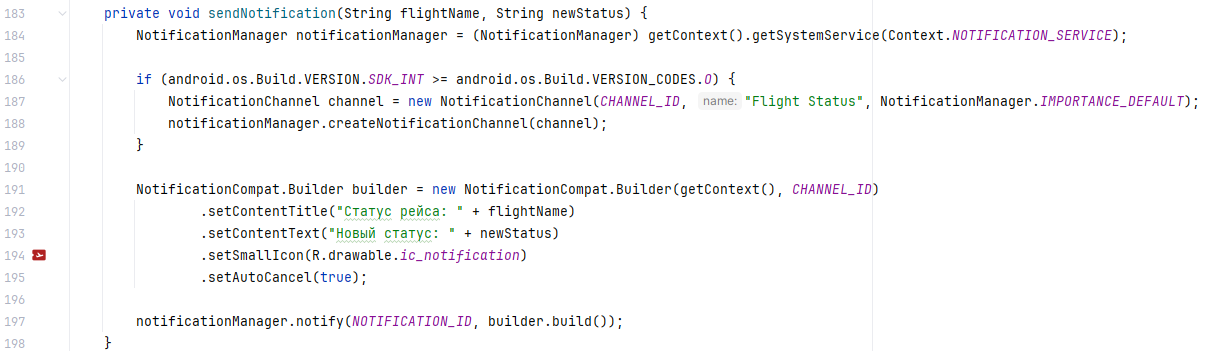
### **4.2.4 УЛУЧШЕНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ОПЫТА**

**УВЕДОМЛЕНИЯ**

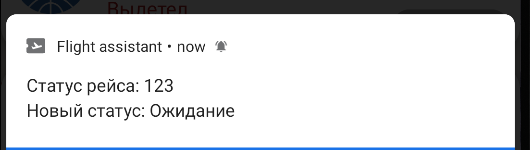
В приложении реализованы уведомления пользователя о смене статуса рейса для удобства отслеживания информации.

Для создания уведомлений используется следующая конструкция – листинг 1:

*Листинг 1. ListAdapter.java*



Получение уведомления– рисунок 2.



*Рисунок 2. Уведомление.*

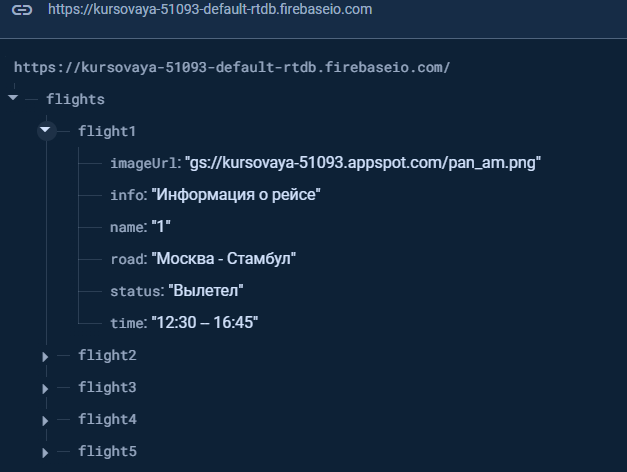
Вывод

Уведомления от приложения улучшают пользовательский опыт, делая приложение более полезным для пользователя.

## **4.3 ХРАНЕНИЕ ДАНЫХ**

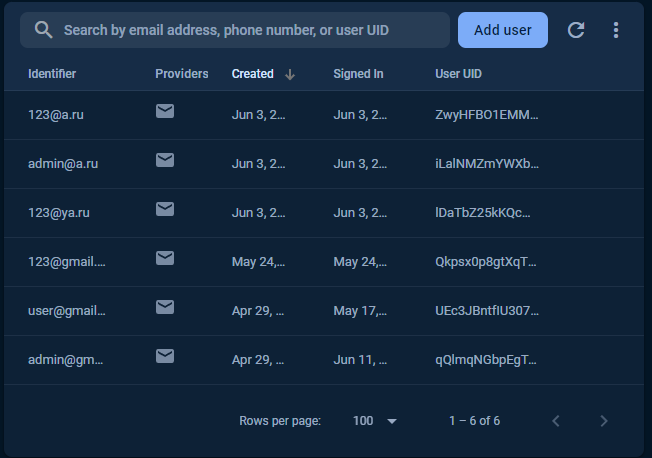
В программном комплексе мы используем Firebase для хранения данных, управления пользователями и хранения файлов. Firebase предоставляет различные сервисы для этих целей, такие как Realtime Database, Authentication и Storage.

Realtime Database — это инструмент для хранения и синхронизации данных в реальном времени. Он позволяет хранить информацию о рейсах и обеспечивает мгновенную синхронизацию с клиентскими приложениями. Данные о рейсах хранятся в соответствии с уникальными идентификаторами, что обеспечивает эффективное управление информацией. Вся информации хранится в соответствии с идентификаторами – рисунок 3.



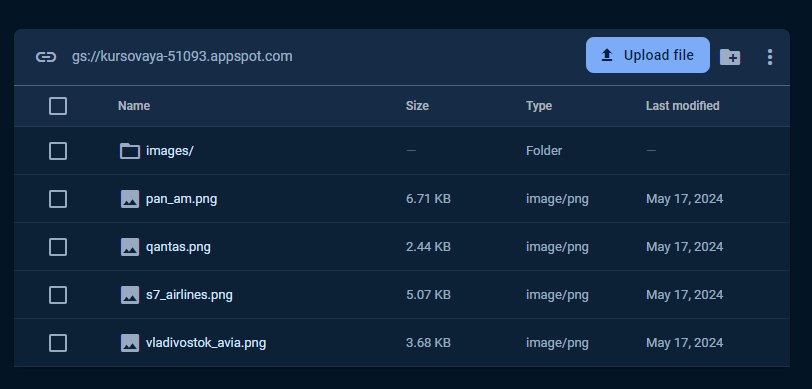
*Рисунок 3. Firebase Realtime Database.*

Firebase Authentication обеспечивает управление аутентификацией пользователей, поддерживая различные методы входа, включая электронную почту и пароль. Для интеграции Firebase в программный комплекс необходимо добавить соответствующие библиотеки и настроить проект. Это включает добавление зависимостей Firebase в файл build.gradle, в том числе библиотек для Realtime Database, Authentication и Storage.который добавлен в проект для инициализации Firebase – рисунок 4.



*Рисунок 4. Firebase* Authentication*.*

Firebase Storage предназначен для хранения и получения различных файлов, включая изображения. В этом программном комплексе Firebase Storage используется для хранения изображений обложек рейсов. Пример хранения изображения рейса в Firebase Storage – рис. 5.



*Рисунок 5. Firebase* Storage*.*

Вывод

Использование Firebase в программном комплексе обеспечивает надежное и эффективное управление данными пользователей и рейсов. Realtime Database позволяет сохранять и синхронизировать данные в реальном времени, Authentication обеспечивает безопасный доступ к системе, а Storage предоставляет возможности для хранения и управления мультимедийными файлами. Это создает основу для разработки современных и масштабируемых приложений с богатым функционалом и удобным интерфейсом, повышая их привлекательность и конкурентоспособность.

## **4.4 ОБЕСПЕЧЕНЕНИЕ БЕЗОПАСНОСТИ ИНФОРМАЦИИ**

В приложении реализована ролевая модель для обеспечения безопасности данных. Она включает две роли: Пользователь и Администратор. Каждая роль имеет свои уровни доступа и привилегии. Например, Администратор может добавлять, изменять и удалять рейсы. Это обеспечивает эффективное управление доступом к данным и защиту конфиденциальности. Пример представлен на рисунке 6.

|  |  |
| --- | --- |
| *а) администратор* | *б) пользователь* |

*Рисунок 6. Ролевая модель приложения.*

Вывод: Применение такого подхода к безопасности данных в приложении обеспечивает защиту конфиденциальной информации и предотвращает возможные атаки на систему. Это осуществляется путем определения четких ролей и уровней доступа. Такая стратегия помогает сохранить целостность и конфиденциальность данных, предотвращая несанкционированный доступ и использование информации.

## **4.5 СИСТЕМА КОНТРОЛЯ ВЕРСИЙ**

Использование Git для управления кодом и ресурсами проекта позволило нам эффективно отслеживать изменения, создавать различные версии приложения и управлять историей изменений. Ветвление в Git позволило работать над новыми функциями и исправлениями ошибок в изолированных средах, что повысило стабильность и надежность процесса разработки. Регулярные коммиты обеспечили возможность быстрого возврата к предыдущим версиям и сохранность кодовой базы в случае проблем.

Проект доступен по ссылке: https://github.com/paraumir/Flight\_assistent

# **5 WIREFRAME ПРИЛОЖЕНИЯ**

Реализация пользовательского сценария:

1. Просмотр информации о рейсе.

- Пользователь находится в списке рейсов.

- На экране пользователь видит актуальную информацию о рейсах. Если пользователь администратор, то он способен видеть элементы, для удаления, изменения и добавления рейсов.

- Пользователь нажимает на рейс.

- После вызывается новая активность.

- Пользователь нажимает на кнопку оставить отзыв.

Вайрфрейм (англ. wireframe) представляет собой набор простых, схематичных чертежей или макетов, которые используются в проектировании веб-сайтов, мобильных приложений и других интерфейсов.

Вайрфрейм приложения представлен на рис 7.

|  |  |  |
| --- | --- | --- |
| *Вход* | *Регистрация* | *Главная* |
| Детально | *Главная админ* | *Добавить\ Изменить* |
|  |  |  |
|  |  |  |

*Рисунок 7. Wireframe приложения.*

# **6 ПРИМЕР РАБОТЫ ПРИЛОЖЕНИЯ**

Варианты использования от разных ролей.

1. Вход/Регистрация пользователя, после чего переход на главную страницу – рисунок 8.

|  |  |  |
| --- | --- | --- |
| *Вход* | *Регистрация* | *Главная* |

*Рисунок 8. Вход/Регистрация.*

1. Взаимодействие с списком пользователем представлено на рисунке 9.

|  |  |  |
| --- | --- | --- |
| *Поиск* | *Детально* | *Инструкция* |

*Рисунок 9. Взаимодействие с списком - пользователь.*

1. Взаимодействие с списком в роле администратора представлено на рисунке 10.

|  |  |  |
| --- | --- | --- |
| *Список* | *Добавление* | *Изменение* |
|  |  |  |
|  |  |  |

*Рисунок 10. Взаимодействие с списком - администратор.*

# **7 ИНСТРУКЦИЯПОЛЬЗОВАТЕЛЮ**

Главная

Раздел главная состоит из списка актуальных рейсов. При нажатии на рейс происходит переход к детальному отображению.

Инфо

На верхней панели присутствуют информационные кнопки. При нажатии на них выводится информация о приложении.

Редактирование

Для администратора. На верхней панели расположена кнопка добавления рейса с переходом в соответствующий интерфейс. На блоках расположены кнопки удаления и изменения.

Эта инструкция поможет быстро освоить основные функции приложения и эффективно использовать его для поиска и управления информацией о рейсах.

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы удалось успешно решить все поставленные задачи, что подтверждает достижение цели разработки мобильного приложения — информационно-справочной системы "Авиарейсы аэропорта".

Изначально был проведен глубокий анализ предметной области, что позволило лучше понять её специфику и особенности. Затем был выполнен обзор существующих аналогов, что помогло выделить уникальные черты и преимущества нашего приложения.

Формулирование технического задания стало основой для точного определения требований и ожиданий от продукта. На следующем этапе была разработана архитектура программной системы, обеспечившая эффективное проектирование и последующую реализацию приложения.

Проектирование интерфейса мобильного приложения было ориентировано на удобство и простоту использования для конечных пользователей. Далее был написан и отлажен код программы на языке высокого уровня Java, а также проведено тестирование, что гарантировало стабильную и надежную работу приложения.

На завершающем этапе было проведено контрольное испытание программы, в ходе которого были продемонстрированы все этапы работы приложения, что подтвердило его функциональность и соответствие заявленным требованиям.

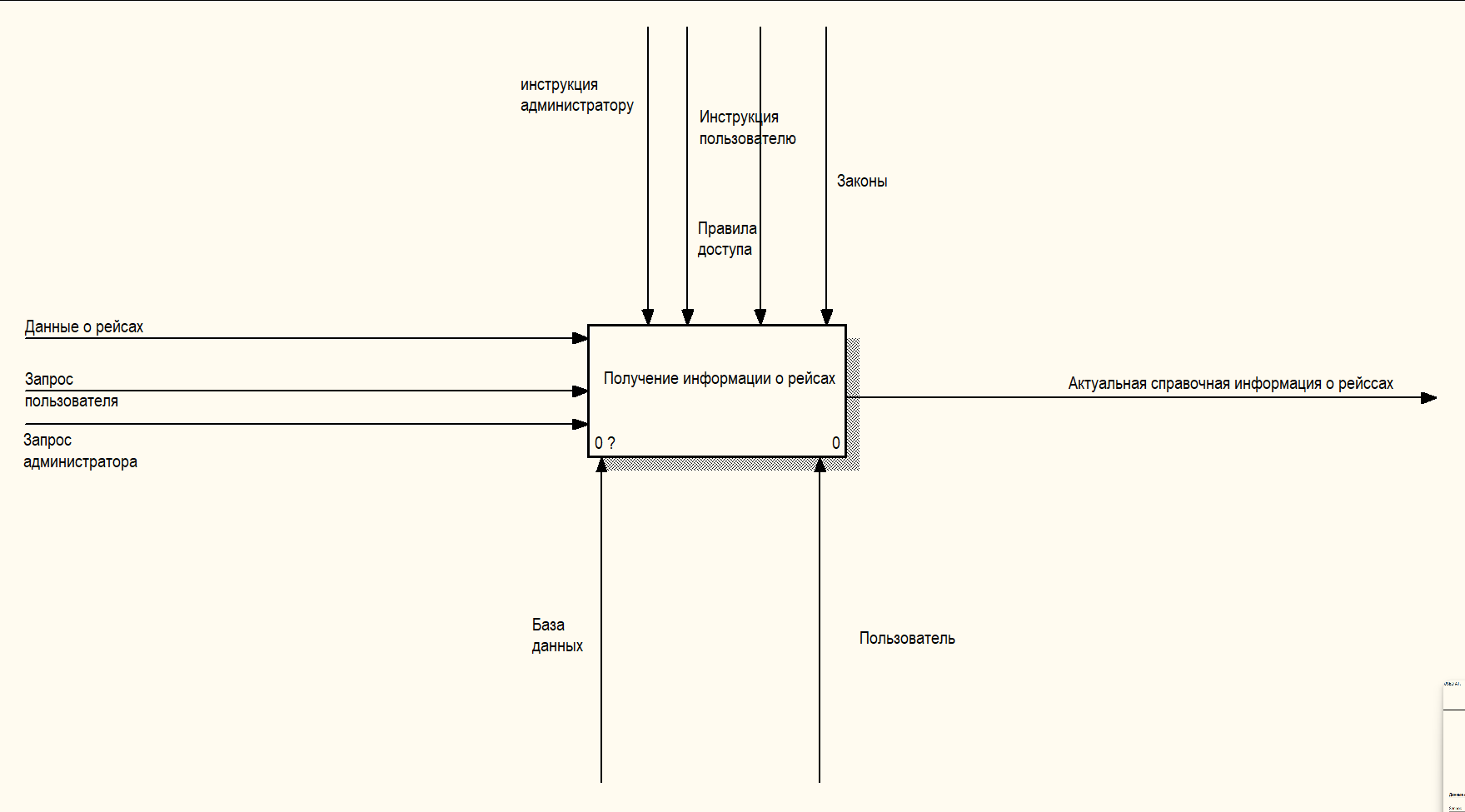
Таким образом, выполнение всех поставленных задач позволило достичь цели курсовой работы и создать мобильное приложение, предоставляющее пользователям удобный доступ к информации о рейсах аэропорта, их характеристиках, работе и обслуживании.

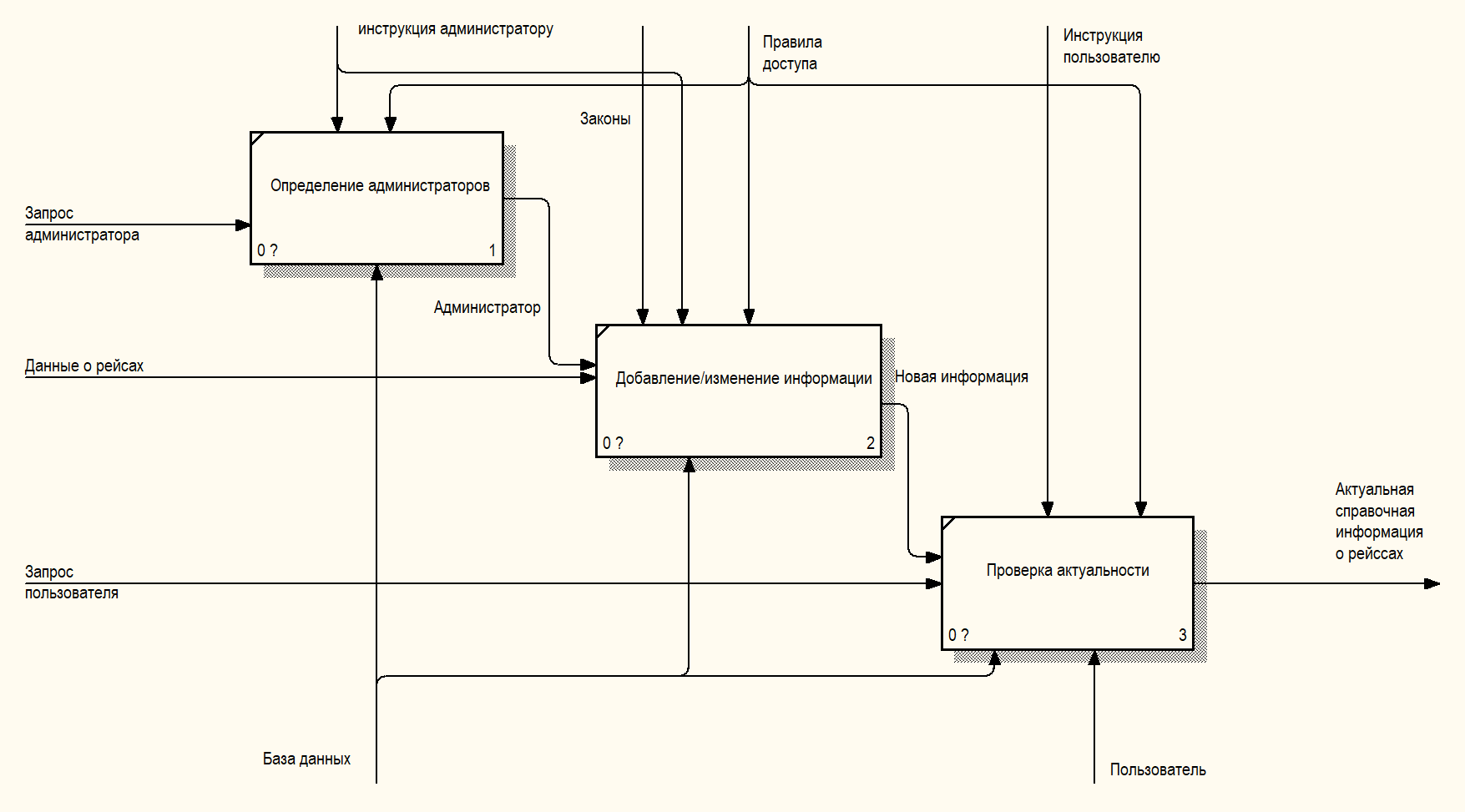
# **СПИСОК ЛИТЕРАТУРЫ**

1. ГОСТ 19.201-78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению.
2. Developer Android / [Электронный ресурс] //: [сайт]. — URL: https://developer.android.com/ (дата обращения: 20.05.2024).
3. Колисниченко Д. Н. Программирование для Android. Самоучитель. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2020 — 288 с.
4. Клифтон Я. Проектирование пользовательского интерфейса в Android. 2-е изд. – М.: ДМК-Пресс, 2017. – 452 с
5. Рудаков А.В. Технология разработки программных продуктов. Практикум: учеб. пособие для студ. - 4 -е изд. – М.: Издательский центр «Академия», 2014. – 192 с.
6. Android Tools / [Электронный ресурс] //: [сайт]. — URL: https://android-tools.ru/ (дата обращения: 20.05.2024).
7. The Destination for Java Developers [Электронный ресурс]. URL: https://dev.java/ (дата обращения 16.05.2024).
8. Github: Where the world builts software [Электронный ресурс]. URL: https://github.com/ (дата обращения 18.05.2024).
9. Firebase / [Электронный ресурс] //: [сайт]. — URL: https://firebase.google.com/docs?hl=en (дата обращения: 20.05.2024)

# **ПРИЛОЖЕНИЕ**

*Приложение 1. IDEF0*





*Приложение 2. MainActivity.java*

package com.example.kurs;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentManager;  
import androidx.fragment.app.FragmentTransaction;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.text.Editable;  
import android.text.TextWatcher;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.Toast;  
  
import com.example.kurs.databinding.ActivityMainBinding;  
import com.google.firebase.auth.FirebaseAuth;  
import com.google.firebase.auth.FirebaseUser;  
import com.google.firebase.database.DataSnapshot;  
import com.google.firebase.database.DatabaseError;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.database.ValueEventListener;  
  
import java.util.ArrayList;  
  
public class MainActivity extends AppCompatActivity {  
  
 private ActivityMainBinding binding;  
 private ListAdapter listAdapter;  
 private DatabaseReference databaseReference;  
 private final String LOG\_TAG = "myLogs";  
 private String admin = "qQlmqNGbpEgTrpH05YDtPCcRHSJ3";  
 private Boolean isAdmin;  
 private ArrayList<ListData> dataArrayList = new ArrayList<>();  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityMainBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
  
 databaseReference = FirebaseDatabase.*getInstance*().getReference().child("flights");  
  
 binding.addButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 startActivity(new Intent(MainActivity.this, ADD.class));  
 }  
 });  
  
 FirebaseAuth mAuth = FirebaseAuth.*getInstance*();  
 FirebaseUser currentUser = mAuth.getCurrentUser();  
 if (currentUser != null) {  
 isAdmin = admin.equals(currentUser.getUid());  
 }  
  
 if (isAdmin) {  
 binding.addButton.setVisibility(View.*VISIBLE*);  
 } else {  
 binding.addButton.setVisibility(View.*GONE*);  
 }  
  
 binding.infoButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 showFragment(new InfoFragment());  
 }  
 });  
  
 binding.creatorButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 showFragment(new CreatorFragment());  
 }  
 });  
  
 binding.instructionButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 showFragment(new instructionFragment());  
 }  
 });  
  
 databaseReference.addValueEventListener(new ValueEventListener() {  
 @Override  
 public void onDataChange(DataSnapshot dataSnapshot) {  
 dataArrayList.clear();  
 for (DataSnapshot snapshot : dataSnapshot.getChildren()) {  
 ListData listData = snapshot.getValue(ListData.class);  
 if (listData != null) {  
 listData.setFlight(snapshot.getKey());  
 dataArrayList.add(listData);  
 }  
 }  
 listAdapter = new ListAdapter(MainActivity.this, dataArrayList);  
 binding.listview.setAdapter(listAdapter);  
 }  
  
 @Override  
 public void onCancelled(DatabaseError databaseError) {  
 Toast.*makeText*(MainActivity.this, "Ошибка загрузки данных", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
  
 binding.listview.setClickable(true);  
  
 binding.listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 @Override  
 public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {  
 ListData selectedFlight = (ListData) adapterView.getItemAtPosition(i);  
 Intent intent = new Intent(MainActivity.this, DetailedActivity.class);  
 intent.putExtra("name", selectedFlight.getName());  
 intent.putExtra("time", selectedFlight.getTime());  
 intent.putExtra("road", selectedFlight.getRoad());  
 intent.putExtra("inf", selectedFlight.getInfo());  
 intent.putExtra("image", selectedFlight.getImageUrl());  
 intent.putExtra("status", selectedFlight.getstatus());  
 startActivity(intent);  
 }  
 });  
  
binding.searchEditText.addTextChangedListener(new TextWatcher() {  
 @Override  
 public void beforeTextChanged(CharSequence s, int start, int count, int after) {  
}  
  
 @Override  
 public void onTextChanged(CharSequence s, int start, int before, int count) {  
 if (listAdapter != null) {  
 listAdapter.filter(s.toString());  
 }  
 }  
  
 @Override  
 public void afterTextChanged(Editable s) {  
}  
 });  
 }  
  
 private void showFragment(Fragment fragment) {  
 FragmentManager fragmentManager = getSupportFragmentManager();  
 FragmentTransaction transaction = fragmentManager.beginTransaction();  
 transaction.replace(binding.fragmentContainer.getId(), fragment);  
 transaction.addToBackStack(null);  
 transaction.commit();  
 }  
}

*Приложение 3. activity\_main.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity"  
 android:background="@drawable/backgrnd2">  
 <LinearLayout  
 android:id="@+id/button\_panel"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 android:layout\_marginTop="0dp"  
 android:background="@color/brick">  
  
 <Button  
 android:id="@+id/add\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:backgroundTint="@color/brick"  
 android:text="+"  
 android:textSize="35sp"  
 app:cornerRadius="20dp" />  
  
 <Button  
 android:id="@+id/info\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:backgroundTint="@color/brick"  
 android:text=""  
 android:textSize="35sp"  
 app:cornerRadius="20dp"  
 android:drawableRight="@drawable/baseline\_discount\_24"/>  
  
 <Button  
 android:id="@+id/creator\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:backgroundTint="@color/brick"  
 android:text=""  
 android:textSize="35sp"  
 app:cornerRadius="20dp"  
 android:drawableRight="@drawable/baseline\_accessibility\_new\_24" />  
  
 <Button  
 android:id="@+id/instruction\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:backgroundTint="@color/brick"  
 android:text=""  
 android:textSize="35sp"  
 app:cornerRadius="20dp"  
 android:drawableRight="@drawable/baseline\_book\_24"/>  
 </LinearLayout>  
  
 <EditText  
 android:id="@+id/searchEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@id/button\_panel"  
 android:background="@color/silver"  
 android:hint="Поиск по номеру рейса"  
 android:padding="8dp"  
 app:layout\_constraintTop\_toBottomOf="@+id/button\_panel" />  
  
 <ListView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:id="@+id/listview"  
 android:scrollbars="vertical"  
 android:layout\_marginTop="110dp"  
 tools:listitem="@layout/list\_item"  
 android:divider="@android:color/transparent"  
 android:dividerHeight="10.0sp"  
 android:layout\_below="@id/searchEditText"/>  
  
 <FrameLayout  
 android:id="@+id/fragment\_container"  
 android:layout\_width="360dp"  
 android:layout\_height="360dp"  
 android:layout\_marginTop="70dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="@+id/button\_panel" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

*Приложение 4. Login.java*

package com.example.kurs;  
  
import androidx.appcompat.app.AppCompatActivity;  
import android.os.Bundle;  
import androidx.annotation.NonNull;  
import android.content.Intent;  
import android.util.Patterns;  
import android.view.View;  
import android.widget.Toast;  
import com.example.kurs.databinding.ActivityLoginBinding;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
  
public class login extends AppCompatActivity {  
  
 private ActivityLoginBinding binding; private FirebaseAuth auth;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityLoginBinding.*inflate*(getLayoutInflater());setContentView(binding.getRoot());auth = FirebaseAuth.*getInstance*();  
  
 binding.loginButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String email = binding.loginEmail.getText().toString();  
 String pass = binding.loginPassword.getText().toString();  
  
 if (!email.isEmpty() && Patterns.*EMAIL\_ADDRESS*.matcher(email).matches()) {  
 if (!pass.isEmpty()) {  
 auth.signInWithEmailAndPassword(email, pass)  
 .addOnSuccessListener(new OnSuccessListener<AuthResult>() {  
 @Override  
 public void onSuccess(AuthResult authResult) {  
 Toast.*makeText*(login.this, "Вы вошли", Toast.*LENGTH\_SHORT*).show();  
 startActivity(new Intent(login.this, MainActivity.class));  
 finish();  
 }  
 }).addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 Toast.*makeText*(login.this, "ошибка входа", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 } else {  
 binding.loginPassword.setError("Поле не может быть пустым");  
 }  
 } else if (email.isEmpty()) {  
 binding.loginEmail.setError("Поле не может быть пустым");  
 } else {  
 binding.loginEmail.setError("Неверная почта");  
 }  
 }  
 });  
  
 binding.signUpRedirectText.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 startActivity(new Intent(login.this, authentification.class));  
 }  
 });  
 }  
}

*Приложение 5. Activity\_login.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:background="@drawable/backgrnd2"  
 tools:context=".login">  
  
 <androidx.cardview.widget.CardView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="30dp"  
 app:cardCornerRadius="30dp"  
 app:cardElevation="20dp">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_gravity="center\_horizontal"  
 android:padding="24dp"  
 android:background="@drawable/brick\_border">  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Войти"  
 android:id="@+id/loginText"  
 android:textSize="36sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:textColor="@color/brick"/>  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/login\_email"  
 android:background="@drawable/custom\_edittext"  
 android:layout\_marginTop="40dp"  
 android:padding="8dp"  
 android:hint="Почта"  
 android:drawableLeft="@drawable/baseline\_account\_circle\_24"  
 android:textColor="@color/black"  
 android:drawablePadding="8dp"/>  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/login\_password"  
 android:background="@drawable/custom\_edittext"  
 android:layout\_marginTop="20dp"  
 android:inputType="textPassword"  
 android:padding="8dp"  
 android:hint="Пароль"  
 android:drawableLeft="@drawable/baseline\_bug\_report\_24"  
 android:textColor="@color/black"  
 android:drawablePadding="8dp"/>  
  
 <Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:text="Войти"  
 android:id="@+id/login\_button"  
 android:textSize="18sp"  
 android:layout\_marginTop="30dp"  
 android:backgroundTint="@color/brick"  
 app:cornerRadius = "20dp"/>  
  
  
 <TextView  
 android:id="@+id/signUpRedirectText"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="20dp"  
 android:padding="8dp"  
 android:layout\_gravity="center"  
 android:text="Зарегистрировать"  
 android:textAlignment="center"  
 android:textColor="@color/brick"  
 android:textSize="18sp" />  
  
 </LinearLayout>  
  
 </androidx.cardview.widget.CardView>  
</LinearLayout>

*Приложение 6. ListData.java*

package com.example.kurs;  
  
public class ListData {  
 private String name;  
 private String time;  
 private String road;  
 private String info;  
 private String imageUrl;  
  
 private String status;  
  
 private String Flight;  
  
 public ListData() {  
 }  
  
 public ListData(String name, String time, String road, String info, String imageUrl, String status) {  
 this.name = name;  
 this.time = time;  
 this.road = road;  
 this.info = info;  
 this.imageUrl = imageUrl;  
 this.status = status;  
 }  
 public String getstatus() {return status;}  
 public void setstatus(String status) {this.status = status;}  
 public String getFlight() {return Flight;}  
 public void setFlight(String Flight) {this.Flight = Flight;}  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getTime() {  
 return time;  
 }  
  
 public void setTime(String time) {  
 this.time = time;  
 }  
  
 public String getRoad() {  
 return road;  
 }  
  
 public void setRoad(String road) {  
 this.road = road;  
 }  
  
 public String getInfo() {  
 return info;  
 }  
  
 public void setInfo(String info) {  
 this.info = info;  
 }  
  
 public String getImageUrl() {  
 return imageUrl;  
 }  
  
 public void setImageUrl(String imageUrl) {  
 this.imageUrl = imageUrl;  
 }  
}

*Приложение 7. ListAdapter.java*

package com.example.kurs;  
  
import android.app.NotificationChannel;  
import android.app.NotificationManager;  
import android.content.Context;  
import android.content.Intent;  
import android.graphics.Color;  
import android.os.Handler;  
import android.util.Log;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ArrayAdapter;  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.core.app.NotificationCompat;  
import com.bumptech.glide.Glide;  
import com.example.kurs.databinding.ListItemBinding;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.auth.FirebaseAuth;  
import com.google.firebase.auth.FirebaseUser;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.Locale;  
  
public class ListAdapter extends ArrayAdapter<ListData> {  
  
 private Boolean isAdmin;  
 final String LOG\_TAG = "myLogs";  
 private static final String *CHANNEL\_ID* = "flight\_status\_channel";  
 private static final int *NOTIFICATION\_ID* = 1;  
  
 private String Admin = "qQlmqNGbpEgTrpH05YDtPCcRHSJ3";  
 private Handler handler;  
 private Runnable statusUpdater;  
 public ArrayList<ListData> originalList;  
 public ArrayList<ListData> filteredList;  
  
 public ListAdapter(@NonNull Context context, ArrayList<ListData> dataArrayList) {  
 super(context, R.layout.*list\_item*, dataArrayList);  
 this.originalList = new ArrayList<>(dataArrayList);  
 this.filteredList = dataArrayList;  
  
 FirebaseAuth mAuth = FirebaseAuth.*getInstance*();  
 FirebaseUser currentUser = mAuth.getCurrentUser();  
 if (currentUser != null) {  
 isAdmin = Admin.equals(currentUser.getUid());  
 }  
  
 handler = new Handler();  
 statusUpdater = new Runnable() {  
 @Override  
 public void run() {  
 notifyDataSetChanged();  
 handler.postDelayed(this, 5000); *// 5 секунд* }  
 };  
 handler.post(statusUpdater);  
 }  
  
 @NonNull  
 @Override  
 public View getView(int position, @Nullable View view, @NonNull ViewGroup parent) {  
 ListData listData = getItem(position);  
 ListItemBinding binding;  
  
 if (view == null) {  
 binding = ListItemBinding.*inflate*(LayoutInflater.*from*(getContext()), parent, false);  
 view = binding.getRoot();  
 view.setTag(binding);  
 } else {  
 binding = (ListItemBinding) view.getTag();  
 }  
  
 if (listData != null) {  
 binding.listName.setText("Рейс №" + listData.getName());  
 binding.listTime.setText(listData.getTime());  
 updateStatus(binding, listData);  
  
 if (listData.getImageUrl() != null && !listData.getImageUrl().isEmpty()) {  
 if (listData.getImageUrl().startsWith("gs://")) {  
 StorageReference storageReference = FirebaseStorage.*getInstance*().getReferenceFromUrl(listData.getImageUrl());  
 storageReference.getDownloadUrl().addOnSuccessListener(uri -> {  
 Glide.*with*(getContext())  
 .load(uri)  
 .into(binding.listImage);  
 }).addOnFailureListener(exception -> {  
 binding.listImage.setImageResource(R.drawable.*plane*);  
 });  
 } else {  
 Glide.*with*(getContext())  
 .load(listData.getImageUrl())  
 .into(binding.listImage);  
 }  
 } else {  
 binding.listImage.setImageResource(R.drawable.*plane*);  
 }  
  
 if (isAdmin) {  
 binding.changeButton.setVisibility(View.*VISIBLE*);  
 binding.deleteButton.setVisibility(View.*VISIBLE*);  
 } else {  
 binding.changeButton.setVisibility(View.*GONE*);  
 binding.deleteButton.setVisibility(View.*GONE*);  
 }  
  
 binding.changeButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String itemId = getItem(position).getFlight();  
 Intent intent = new Intent(getContext(), Change.class);  
 intent.putExtra("itemId", itemId);  
 getContext().startActivity(intent);  
 }  
 });  
  
 binding.deleteButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 DatabaseReference databaseReference = FirebaseDatabase.*getInstance*().getReference().child("flights").child(getItem(position).getFlight());  
 databaseReference.removeValue();  
 String imageUrl = getItem(position).getImageUrl();  
 if (imageUrl != null && !imageUrl.isEmpty()) {  
 StorageReference photoRef = FirebaseStorage.*getInstance*().getReferenceFromUrl(imageUrl);  
 photoRef.delete();  
 }  
 remove(getItem(position));  
 notifyDataSetChanged();  
 }  
 });  
 }  
  
 return view;  
 }  
  
 private void updateStatus(ListItemBinding binding, ListData listData) {  
 String currentTime = getCurrentTime();  
 String flightTime = listData.getTime();  
int currentHours = Integer.*parseInt*(currentTime.substring(0, 2));  
 int currentMinutes = Integer.*parseInt*(currentTime.substring(3, 5));  
 int flightHours = Integer.*parseInt*(flightTime.substring(0, 2));  
 int flightMinutes = Integer.*parseInt*(flightTime.substring(3, 5));  
  
 int currentTotalMinutes = currentHours \* 60 + currentMinutes;  
 int flightTotalMinutes = flightHours \* 60 + flightMinutes;  
  
 String newStatus;  
 int diffMinutes = flightTotalMinutes - currentTotalMinutes;  
  
 if (diffMinutes > 15) {  
 newStatus = "Ожидание";  
 binding.listStatus.setTextColor(Color.*rgb*(41, 102, 13));  
 } else if (diffMinutes > 0) {  
 newStatus = "Подготовка к вылету";  
 binding.listStatus.setTextColor(Color.*rgb*(191, 179, 6));  
 } else {  
 newStatus = "Вылетел";  
 binding.listStatus.setTextColor(Color.*rgb*(178, 34, 34));  
 }  
  
 if (!newStatus.equals(listData.getstatus())) {  
 listData.setstatus(newStatus);  
 DatabaseReference databaseReference = FirebaseDatabase.*getInstance*().getReference("flights");  
 String itemId = listData.getFlight();  
 databaseReference.child(itemId).child("status").setValue(newStatus).addOnSuccessListener(new OnSuccessListener<Void>() {  
 @Override  
 public void onSuccess(Void aVoid) {  
 Log.*d*(LOG\_TAG, "Статус успешно обновлён");  
 sendNotification(listData.getName(), newStatus);  
 }  
 });  
 }  
  
 binding.listStatus.setText(newStatus);  
 }  
 private void sendNotification(String flightName, String newStatus) {  
 NotificationManager notificationManager = (NotificationManager) getContext().getSystemService(Context.*NOTIFICATION\_SERVICE*);  
  
 if (android.os.Build.VERSION.*SDK\_INT* >= android.os.Build.VERSION\_CODES.*O*) {  
 NotificationChannel channel = new NotificationChannel(*CHANNEL\_ID*, "Flight Status", NotificationManager.*IMPORTANCE\_DEFAULT*);  
 notificationManager.createNotificationChannel(channel);  
 }  
  
 NotificationCompat.Builder builder = new NotificationCompat.Builder(getContext(), *CHANNEL\_ID*)  
 .setContentTitle("Статус рейса: " + flightName)  
 .setContentText("Новый статус: " + newStatus)  
 .setSmallIcon(R.drawable.*ic\_notification*)  
 .setAutoCancel(true);  
  
 notificationManager.notify(*NOTIFICATION\_ID*, builder.build());  
 }  
  
 private String getCurrentTime() {  
 Date date = new Date();  
 SimpleDateFormat formatter = new SimpleDateFormat("HH:mm", Locale.*getDefault*());  
 return formatter.format(date);  
 }  
  
 public void filter(String text) {  
 filteredList.clear();  
 if (text.isEmpty()) {  
 filteredList.addAll(originalList);  
 } else {  
 for (ListData item : originalList) {  
 if (item.getName().toLowerCase().contains(text.toLowerCase())) {  
 filteredList.add(item);  
 }  
 }  
 }  
 notifyDataSetChanged();  
 }  
  
 @Override  
 public int getCount() {  
 return filteredList.size();  
 }  
  
 @Nullable  
 @Override  
 public ListData getItem(int position) {  
 return filteredList.get(position);  
 }  
}

*Приложение 8. List\_item.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<androidx.cardview.widget.CardView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_marginStart="12dp"  
 android:layout\_marginEnd="12dp"  
 android:layout\_marginVertical="10dp"  
 android:layout\_marginHorizontal="10dp"  
 app:cardCornerRadius="20dp"  
 app:cardElevation="8dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@color/white">  
  
 <RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="100dp">  
  
 <com.google.android.material.imageview.ShapeableImageView  
 android:id="@+id/listImage"  
 android:layout\_width="60dp"  
 android:layout\_height="60dp"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginBottom="10dp"  
 android:scaleType="centerCrop" />  
  
 <TextView  
 android:id="@+id/listName"  
 android:layout\_width="120dp"  
 android:layout\_height="30dp"  
 android:layout\_marginStart="154dp"  
 android:layout\_marginTop="25dp"  
 android:layout\_toEndOf="@id/listImage"  
 android:text="Рейс №"  
 android:textColor="@color/brick"  
 android:textSize="20sp" />  
  
 <TextView  
 android:id="@+id/listTime"  
 android:layout\_width="140dp"  
 android:layout\_height="30dp"  
 android:layout\_marginStart="-260dp"  
 android:layout\_marginTop="25dp"  
 android:layout\_toEndOf="@id/listName"  
 android:text="Время:"  
 android:textColor="@color/brick"  
 android:textSize="16sp" />  
  
 <TextView  
 android:id="@+id/listStatus"  
 android:layout\_width="140dp"  
 android:layout\_height="50dp"  
 android:layout\_marginStart="-260dp"  
 android:layout\_marginTop="55dp"  
 android:layout\_toEndOf="@id/listName"  
 android:text="Статус:"  
 android:textColor="@color/brick"  
 android:textSize="16sp" />  
  
 <Button  
 android:id="@+id/delete\_button"  
 android:layout\_width="55dp"  
 android:layout\_height="33dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_alignParentEnd="true"  
 android:layout\_marginTop="-1dp"  
 android:layout\_marginEnd="10dp"  
 android:backgroundTint="@color/brick"  
 android:clickable="true"  
 android:focusable="false"  
 android:text="X"  
 android:textSize="9sp"  
 android:visibility="visible"  
 app:cornerRadius="20dp" />  
  
 <Button  
 android:id="@+id/change\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="34dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_alignParentEnd="true"  
 android:layout\_marginTop="63dp"  
 android:layout\_marginEnd="9dp"  
 android:backgroundTint="@color/silver"  
 android:clickable="true"  
 android:focusable="false"  
 android:text="Изменить"  
 android:textSize="9sp"  
 android:visibility="visible"  
 app:cornerRadius="20dp" />  
  
 </RelativeLayout>  
  
</androidx.cardview.widget.CardView>

*Приложение 9.instructionfragment.java*

package com.example.kurs;  
  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
  
import com.example.kurs.databinding.FragmentInstructionBinding;  
  
public class instructionFragment extends Fragment {  
  
 private FragmentInstructionBinding binding;  
  
 public instructionFragment() {  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 binding = FragmentInstructionBinding.*inflate*(inflater, container, false);  
 View view = binding.getRoot();  
  
 binding.closeButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 getParentFragmentManager().popBackStack();  
 }  
 });  
  
 return view;  
 }  
  
 @Override  
 public void onDestroyView() {  
 super.onDestroyView();  
 binding = null;  
 }  
}

*Приложение 10. Fragmen\_instructiont.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:background="@drawable/custom\_fragment">  
  
 <TextView  
 android:id="@+id/info\_text"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Инструкция для User:\n\n У вас есть список. Вы можете его листать. При нажатии на элемент, элемент откроется подробно.\n\nИнструкция для Admin:\n\nВесь функционал роли User.\n\nЕсть возможности редактирования элементов списка и самого списка.\n\nПри редактировании необходимо заполнять все поля."  
 android:padding="16dp"/>  
  
 <Button  
 android:id="@+id/close\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Закрыть"  
 android:backgroundTint="@color/brick"/>  
  
</LinearLayout>

*Приложение 11.Infofragment.java*

package com.example.kurs;  
  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
  
import com.example.kurs.databinding.FragmentInfoBinding;  
  
public class InfoFragment extends Fragment {  
  
 private FragmentInfoBinding binding;  
  
 public InfoFragment() {}  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 binding = FragmentInfoBinding.*inflate*(inflater, container, false);  
 View view = binding.getRoot();  
  
 binding.closeButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 getParentFragmentManager().popBackStack();  
 }  
 });  
  
 return view;  
 }  
  
 @Override  
 public void onDestroyView() {  
 super.onDestroyView();  
 binding = null;}  
}

*Приложение 12. Fragmen\_info.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:background="@drawable/custom\_fragment">  
  
 <TextView  
 android:id="@+id/info\_text"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Информация о приложении:\n\nМобильная информационно-справочная система Авиарейсы аэропорта.\n\nПриложение имеет две роли User и Admin.\n\nВся информация в приложении вымышлена.\n\nПриложении находится в бета-версии."  
 android:padding="16dp"/>  
  
 <Button  
 android:id="@+id/close\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Закрыть"  
 android:backgroundTint="@color/brick"/>  
  
</LinearLayout>

*Приложение 13.Creatorfragment.java*

package com.example.kurs;  
  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
  
import com.example.kurs.databinding.FragmentCreatorBinding;  
  
public class CreatorFragment extends Fragment {  
  
 private FragmentCreatorBinding binding;  
  
 public CreatorFragment() {  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 binding = FragmentCreatorBinding.*inflate*(inflater, container, false);  
 View view = binding.getRoot();  
  
 binding.closeButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 getParentFragmentManager().popBackStack();  
 }  
 });  
  
 return view;  
 }  
  
 @Override  
 public void onDestroyView() {  
 super.onDestroyView();  
 binding = null;  
 }  
}

*Приложение 14. Fragmen\_creator.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:background="@drawable/custom\_fragment">  
  
 <TextView  
 android:id="@+id/info\_text"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Создатель:\n\nРассадин Глеб Андреевич\n\nИКБО-33-22\n\n2024\n\nglebras@yandex.ru"  
 android:padding="16dp"/>  
  
 <Button  
 android:id="@+id/close\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Закрыть"  
 android:backgroundTint="@color/brick"/>  
  
</LinearLayout>

*Приложение 15.DetailedActivity.java*

package com.example.kurs;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
  
import com.example.kurs.databinding.ActivityDetailedBinding;  
import com.bumptech.glide.Glide;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
  
public class DetailedActivity extends AppCompatActivity {  
  
 ActivityDetailedBinding binding;  
 FirebaseStorage storage;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityDetailedBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
  
 storage = FirebaseStorage.*getInstance*();  
  
 Intent intent = getIntent();  
 if (intent != null){  
 String name = intent.getStringExtra("name");  
 String time = intent.getStringExtra("time");  
 String road = intent.getStringExtra("road");  
 String inf = intent.getStringExtra("inf");  
 String imageUrl = intent.getStringExtra("image");  
 String status = intent.getStringExtra("status");  
  
 binding.detailName.setText("Рейс №" + name);  
 binding.detailTime.setText(time);  
 binding.detailIngredients.setText(road);  
 binding.detailDesc.setText(inf);  
 binding.detailStatus.setText(status);  
  
 if (imageUrl != null) {  
 if (imageUrl.startsWith("gs://")) {  
 *// Handle gs:// URL using Firebase Storage* StorageReference storageReference = storage.getReferenceFromUrl(imageUrl);  
 storageReference.getDownloadUrl().addOnSuccessListener(uri -> {  
 Glide.*with*(this)  
 .load(uri)  
 .into(binding.detailImage);  
 }).addOnFailureListener(e -> {  
 });  
 } else {  
 Glide.*with*(this)  
 .load(imageUrl)  
 .into(binding.detailImage);  
 }  
 }  
 }  
 }  
}

*Приложение 16. Activity\_Detailed.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<ScrollView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@color/white">  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".DetailedActivity"  
 android:background="@color/white">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/detailName"  
 android:layout\_marginTop="14dp"  
 android:text="Рейс №"  
 android:textSize="24sp"  
 android:textColor="@color/brick"  
 android:textStyle="bold"  
 android:layout\_gravity="center"/>  
  
 <androidx.cardview.widget.CardView  
 android:layout\_width="380dp"  
 android:layout\_height="380dp"  
 android:id="@+id/cardImage"  
 android:layout\_marginTop="10dp"  
 android:layout\_gravity="center"  
 app:cardCornerRadius="20dp"  
 app:cardElevation="8dp">  
  
 <ImageView  
 android:id="@+id/detailImage"  
 android:layout\_width="380dp"  
 android:layout\_height="380dp"  
 android:layout\_gravity="center"  
 android:layout\_marginTop="10dp"  
 android:scaleType="centerCrop" />  
  
 </androidx.cardview.widget.CardView>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Время вылета/прилёта:"  
 android:textColor="@color/brick"  
 android:textStyle="bold"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginTop="30dp"  
 android:textSize="18sp"/>  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="время"  
 android:layout\_marginStart="10dp"  
 android:textSize="16sp"  
 android:id="@+id/detailTime"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Откуда - куда:"  
 android:textColor="@color/brick"  
 android:textStyle="bold"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginTop="20dp"  
 android:textSize="18sp"/>  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="текст"  
 android:layout\_marginStart="10dp"  
 android:textSize="16sp"  
 android:id="@+id/detailIngredients"/>  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Статус:"  
 android:textColor="@color/brick"  
 android:textStyle="bold"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginTop="30dp"  
 android:textSize="18sp"/>  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="статус"  
 android:layout\_marginStart="10dp"  
 android:textSize="16sp"  
 android:id="@+id/detailStatus"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Информация о рейсе:"  
 android:textColor="@color/brick"  
 android:textStyle="bold"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginTop="20dp"  
 android:textSize="18sp"/>  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="текст"  
 android:layout\_marginStart="10dp"  
 android:textSize="16sp"  
 android:id="@+id/detailDesc"/>  
  
</LinearLayout>  
</ScrollView>

*Приложение 17.Change.java*

package com.example.kurs;  
  
import android.content.Intent;  
import android.net.Uri;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.view.View;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.example.kurs.databinding.ChangeLayoutBinding;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.DatabaseError;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.database.ValueEventListener;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import com.google.firebase.storage.UploadTask;  
  
import java.util.UUID;  
import java.util.regex.Pattern;  
  
public class Change extends AppCompatActivity {  
  
 ChangeLayoutBinding binding;  
 DatabaseReference databaseReference;  
 long count;  
  
 private static final int *PICK\_IMAGE\_REQUEST* = 1;  
 private Uri imageUri;  
  
 private static final Pattern *TIME\_PATTERN* = Pattern.*compile*("^\\d{2}:\\d{2} -- \\d{2}:\\d{2}$");  
  
 String ITEMID = "";  
 String currentImageUrl = "";  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ChangeLayoutBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
  
 databaseReference = FirebaseDatabase.*getInstance*().getReference().child("flights");  
 final String itemId = getIntent().getStringExtra("itemId");  
 ITEMID = itemId;  
 databaseReference.child(itemId).addListenerForSingleValueEvent(new ValueEventListener() {  
 @Override  
 public void onDataChange(@NonNull com.google.firebase.database.DataSnapshot dataSnapshot) {  
 if (dataSnapshot.exists()) {  
 ListData data = dataSnapshot.getValue(ListData.class);  
 if (data != null) {  
 binding.nameEditText.setText(data.getName());  
 binding.timeEditText.setText(data.getTime());  
 binding.roadEditText.setText(data.getRoad());  
 binding.infoEditText.setText(data.getInfo());  
 binding.statusEditText.setText(data.getstatus());  
 currentImageUrl = data.getImageUrl();  
 }  
 }  
 }  
  
 @Override  
 public void onCancelled(@NonNull DatabaseError databaseError) {  
 }  
 });  
  
 binding.imageUrlButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 openGallery();  
 }  
 });  
  
 binding.changeButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String name = binding.nameEditText.getText().toString().trim();  
 String time = binding.timeEditText.getText().toString().trim();  
 String road = binding.roadEditText.getText().toString().trim();  
 String info = binding.infoEditText.getText().toString().trim();  
 String status = binding.statusEditText.getText().toString().trim();  
  
 if (!name.isEmpty() && !time.isEmpty() && !road.isEmpty() && !info.isEmpty() && !status.isEmpty()) {  
 if (*TIME\_PATTERN*.matcher(time).matches()) {  
 uploadImageToFirebase(name, time, road, info, status);  
 } else {  
 Toast.*makeText*(Change.this, "Введите время в формате XX:XX -- XX:XX", Toast.*LENGTH\_SHORT*).show();  
 }  
 } else {  
 Toast.*makeText*(Change.this, "Заполните все поля", Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 });  
 }  
  
 private void openGallery() {  
 Intent galleryIntent = new Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*);  
 startActivityForResult(galleryIntent, *PICK\_IMAGE\_REQUEST*);  
 }  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
  
 if (requestCode == *PICK\_IMAGE\_REQUEST* && resultCode == *RESULT\_OK* && data != null && data.getData() != null) {  
 imageUri = data.getData();  
 }  
 }  
  
 private void uploadImageToFirebase(final String name, final String time, final String road, final String info, final String status) {  
 if (imageUri != null) {  
 StorageReference imageRef = FirebaseStorage.*getInstance*().getReference().child("images/" + UUID.*randomUUID*().toString());  
  
 imageRef.putFile(imageUri)  
 .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {  
 @Override  
 public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 imageRef.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {  
 @Override  
 public void onSuccess(Uri uri) {  
 String imageUrl = uri.toString();  
 updateFlightInDatabase(name, time, road, info, imageUrl, status);  
 }  
 });  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 Toast.*makeText*(Change.this, "Ошибка при загрузке изображения: " + e.getMessage(), Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 } else {  
 *// No image selected, retain the current image URL* updateFlightInDatabase(name, time, road, info, currentImageUrl, status);  
 }  
 }  
  
 private void updateFlightInDatabase(String name, String time, String road, String info, String imageUrl, String status) {  
 ListData newData = new ListData(name, time, road, info, imageUrl, status);  
 databaseReference.child(ITEMID).setValue(newData).addOnCompleteListener(task -> {  
 if (task.isSuccessful()) {  
 Toast.*makeText*(Change.this, "Изменено", Toast.*LENGTH\_SHORT*).show();  
 finish();  
 } else {  
 Toast.*makeText*(Change.this, "Ошибка при изменении данных", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
}

*Приложение 18.Change\_layout.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<ScrollView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@color/white">  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".Change"  
 android:background="@color/white">  
  
 <EditText  
 android:id="@+id/nameEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Номер Рейса"  
 android:layout\_margin="16dp"/>  
  
 <EditText  
 android:id="@+id/timeEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Время"  
 android:layout\_below="@id/nameEditText"  
 android:layout\_margin="16dp"/>  
  
 <EditText  
 android:id="@+id/roadEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Откуда - куда"  
 android:layout\_below="@id/timeEditText"  
 android:layout\_margin="16dp"/>  
 <EditText  
 android:id="@+id/statusEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Статус"  
 android:layout\_below="@id/roadEditText"  
 android:layout\_margin="16dp"/>  
  
 <EditText  
 android:id="@+id/infoEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Информация о рейсе"  
 android:layout\_below="@id/statusEditText"  
 android:layout\_margin="16dp"/>  
  
 <Button  
 android:id="@+id/imageUrlButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Выбрать изображение"  
 android:layout\_below="@id/infoEditText"  
 android:layout\_margin="16dp"  
 android:backgroundTint="@color/brick"/>  
  
  
 <Button  
 android:id="@+id/changeButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Изменить"  
 android:layout\_below="@id/imageUrlButton"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="24dp"  
 android:backgroundTint="@color/brick"/>  
  
</RelativeLayout>  
</ScrollView>

*Приложение 19.authentification.java*

package com.example.kurs;  
  
import androidx.appcompat.app.AppCompatActivity;  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Toast;  
import com.example.kurs.databinding.ActivityAuthenticationBinding;  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
  
public class authentification extends AppCompatActivity {  
  
 private ActivityAuthenticationBinding binding;  
 private FirebaseAuth auth;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityAuthenticationBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
  
 auth = FirebaseAuth.*getInstance*();  
  
 binding.signupButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 String user = binding.signupEmail.getText().toString().trim();  
 String pass = binding.signupPassword.getText().toString().trim();  
  
 if (user.isEmpty()){  
 binding.signupEmail.setError("Поле не может быть пустым");  
 }  
 if (pass.isEmpty()){  
 binding.signupPassword.setError("Поле не может быть пустым");  
 } else{  
 auth.createUserWithEmailAndPassword(user, pass).addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(Task<AuthResult> task) {  
 if (task.isSuccessful()) {  
 Toast.*makeText*(authentification.this, "Вы зарегистрированы", Toast.*LENGTH\_SHORT*).show();  
 startActivity(new Intent(authentification.this, login.class));  
 } else {  
 Toast.*makeText*(authentification.this, "Ошибка регистрации" + task.getException().getMessage(), Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 });  
 }  
 }  
 });  
  
 binding.loginRedirectText.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 startActivity(new Intent(authentification.this, login.class));  
 }  
 });  
 }  
}

*Приложение 20.activity\_authentification.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:background="@drawable/backgrnd2"  
 tools:context=".authentification">  
  
 <androidx.cardview.widget.CardView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="30dp"  
 app:cardCornerRadius="30dp"  
 app:cardElevation="20dp">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_gravity="center\_horizontal"  
 android:padding="24dp"  
 android:background="@drawable/brick\_border">  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Зарегистрироваться"  
 android:textSize="23sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:textColor="@color/brick"/>  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/signup\_email"  
 android:background="@drawable/custom\_edittext"  
 android:layout\_marginTop="40dp"  
 android:padding="8dp"  
 android:hint="Почта"  
 android:drawableLeft="@drawable/baseline\_account\_circle\_24"  
 android:drawablePadding="8dp"  
 android:textColor="@color/black"/>  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/signup\_password"  
 android:background="@drawable/custom\_edittext"  
 android:layout\_marginTop="20dp"  
 android:padding="8dp"  
 android:hint="Пароль"  
 android:drawableLeft="@drawable/baseline\_bug\_report\_24"  
 android:drawablePadding="8dp"  
 android:textColor="@color/black"/>  
  
 <Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:text="Зарегистрироваться"  
 android:id="@+id/signup\_button"  
 android:textSize="18sp"  
 android:layout\_marginTop="30dp"  
 android:backgroundTint="@color/brick"  
 app:cornerRadius = "20dp"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/loginRedirectText"  
 android:text="Войти"  
 android:layout\_gravity="center"  
 android:padding="8dp"  
 android:layout\_marginTop="10dp"  
 android:textColor="@color/brick"  
 android:textSize="18sp"/>  
  
 </LinearLayout>  
  
 </androidx.cardview.widget.CardView>  
  
</LinearLayout>

*Приложение 21.ADD.java*

package com.example.kurs;  
  
import android.content.Intent;  
import android.net.Uri;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.view.View;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.example.kurs.databinding.ActivityAddBinding;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.DatabaseError;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.database.ValueEventListener;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import com.google.firebase.storage.UploadTask;  
  
import java.util.UUID;  
import java.util.regex.Pattern;  
  
public class ADD extends AppCompatActivity {  
  
 ActivityAddBinding binding;  
 DatabaseReference databaseReference;  
 long count;  
  
 private static final int *PICK\_IMAGE\_REQUEST* = 1;  
 private Uri imageUri;  
  
 private static final Pattern *TIME\_PATTERN* = Pattern.*compile*("^\\d{2}:\\d{2} -- \\d{2}:\\d{2}$");  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 binding = ActivityAddBinding.*inflate*(getLayoutInflater());  
 setContentView(binding.getRoot());  
  
 databaseReference = FirebaseDatabase.*getInstance*().getReference().child("flights");  
  
 databaseReference.addListenerForSingleValueEvent(new ValueEventListener() {  
 @Override  
 public void onDataChange(@NonNull com.google.firebase.database.DataSnapshot dataSnapshot) {  
 count = dataSnapshot.getChildrenCount();  
 }  
  
 @Override  
 public void onCancelled(@NonNull DatabaseError databaseError) {  
 }  
 });  
  
 binding.imageUrlButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 openGallery();  
 }  
 });  
  
 binding.confirmButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String name = binding.nameEditText.getText().toString().trim();  
 String time = binding.timeEditText.getText().toString().trim();  
 String road = binding.roadEditText.getText().toString().trim();  
 String info = binding.infoEditText.getText().toString().trim();  
 String status = binding.statusEditText.getText().toString().trim();  
  
 if (!name.isEmpty() && !time.isEmpty() && !road.isEmpty() && !info.isEmpty() && !status.isEmpty()) {  
 if (*TIME\_PATTERN*.matcher(time).matches()) {  
 uploadImageToFirebase(name, time, road, info, status);  
 } else {  
 Toast.*makeText*(ADD.this, "Введите время в формате XX:XX -- XX:XX", Toast.*LENGTH\_SHORT*).show();  
 }  
 } else {  
 Toast.*makeText*(ADD.this, "Заполните все поля", Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 });  
 }  
  
 private void openGallery() {  
 Intent galleryIntent = new Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*);  
 startActivityForResult(galleryIntent, *PICK\_IMAGE\_REQUEST*);  
 }  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
  
 if (requestCode == *PICK\_IMAGE\_REQUEST* && resultCode == *RESULT\_OK* && data != null && data.getData() != null) {  
 imageUri = data.getData();  
 }  
 }  
  
 private void uploadImageToFirebase(final String name, final String time, final String road, final String info, final String status) {  
 if (imageUri != null) {  
 StorageReference imageRef = FirebaseStorage.*getInstance*().getReference().child("images/" + UUID.*randomUUID*().toString());  
  
 imageRef.putFile(imageUri)  
 .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {  
 @Override  
 public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 imageRef.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {  
 @Override  
 public void onSuccess(Uri uri) {  
 String imageUrl = uri.toString();  
 addFlightToDatabase(name, time, road, info, imageUrl, status);  
 }  
 });  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 Toast.*makeText*(ADD.this, "Ошибка при загрузке изображения: " + e.getMessage(), Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 } else {  
 *// No image selected, proceed without image URL* addFlightToDatabase(name, time, road, info, "", status);  
 }  
 }  
  
 private void addFlightToDatabase(String name, String time, String road, String info, String imageUrl, String status) {  
 count += 1;  
 String flight = "flight" + count;  
 ListData newData = new ListData(name, time, road, info, imageUrl, status);  
 databaseReference.child(flight).setValue(newData).addOnCompleteListener(task -> {  
 if (task.isSuccessful()) {  
 Toast.*makeText*(ADD.this, "Добавлено", Toast.*LENGTH\_SHORT*).show();  
 finish();  
 } else {  
 Toast.*makeText*(ADD.this, "Ошибка при добавлении данных", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
}

*Приложение 21.activity\_add.xml*

*<?*xml version="1.0" encoding="utf-8"*?>*<ScrollView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@color/white">  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".ADD"  
 android:background="@color/white">  
  
 <EditText  
 android:id="@+id/nameEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Номер Рейса"  
 android:layout\_margin="16dp"/>  
  
 <EditText  
 android:id="@+id/timeEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Время"  
 android:layout\_below="@id/nameEditText"  
 android:layout\_margin="16dp"/>  
  
 <EditText  
 android:id="@+id/roadEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Откуда - куда"  
 android:layout\_below="@id/timeEditText"  
 android:layout\_margin="16dp"/>  
 <EditText  
 android:id="@+id/statusEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Статус"  
 android:layout\_below="@id/roadEditText"  
 android:layout\_margin="16dp"/>  
  
 <EditText  
 android:id="@+id/infoEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Информация о рейсе"  
 android:layout\_below="@id/statusEditText"  
 android:layout\_margin="16dp"/>  
  
 <Button  
 android:id="@+id/imageUrlButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Выбрать изображение"  
 android:layout\_below="@id/infoEditText"  
 android:layout\_margin="16dp"  
 android:backgroundTint="@color/brick"/>  
  
  
 <Button  
 android:id="@+id/confirm\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Добавить"  
 android:layout\_below="@id/imageUrlButton"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="24dp"  
 android:backgroundTint="@color/brick"/>  
  
</RelativeLayout>  
</ScrollView>