

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

БАКАЛАВРСКАЯ РАБОТА

на тему

**Программное обеспечение мобильной
поддержки процессов резервирования
авиабилетов**

Студент _____ Милованцев Вадим Сергеевич
(подпись, дата) (ФИО полностью)

Руководитель _____ Князев В.Н.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____ Макарычев П.П.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ Попова Н.А.
(подпись) (фамилия, инициалы)

Пенза, 2022

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
П.П. Макарычев
«__»____2022 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Милованцев Вадим Сергеевич гр. 18ВП2 факультета ВТ

<Фамилия, имя, отчество полностью>

2. Тема работы Программное обеспечение мобильной поддержки процессов резервирования авиабилетов

Тема утверждена приказом ПГУ № 327/О от “18” апреля 2022

3. Руководитель работы к.т.н., доцент кафедры «МОиПЭВМ» Князев В.Н.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

- 1) *Разработать приложение для мобильных устройств, обеспечивающее поддержку резервирования авиабилетов.*
- 2) *Среда выполнения приложения: операционная система Android 5 и выше*
Среда выполнения сервера: операционная система Windows 10
Языковые средства разработки: UML, Java
Инструментальные средства разработки: Android Studio, Microsoft Project, ArgoUML, Visual Paradigm, Firebase
Технологии разработки: RUP, ООП
- 3) *Основные функции мобильного приложения:*
 - а) *резервирование авиабилета;*
 - б) *поиск авиарейса в соответствии с критериями пользователя;*
 - в) *отмена билета на рейс;*
5. Перечень подлежащих разработке вопросов
 - 1) *Анализ предметной области и постановка задачи*
 - 2) *Проектирование*
 - 3) *Реализация*
 - 4) *Контроль качества программного обеспечения*
 - 5) *Оформление пояснительной записки*

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	<i>Анализ предметной области постановка задачи</i>	15%	13.04.2022– 21.04.2022	
2	<i>Проектирование</i>	25%	22.04.2022– 05.05.2022	
3	<i>Реализация</i>	40%	06.05.2022– 30.05.2022	
4	<i>Контроль качества программного обеспечения</i>	5%	31.05.2022– 02.06.2022	
5	<i>Оформление пояснительной записки</i>	15%	03.06.2022– 17.06.2022	

Дата выдачи задания "22" декабря 2022

Руководитель бакалаврской работы _____

(подпись, дата)

Задание к исполнению принял студент _____

(подпись, дата)

Работу к защите допустить

Декан факультета _____

(подпись, дата)

Реферат

Пояснительная записка содержит 86 листов, 47 рисунков, 17 таблиц, 31 использованных источников, приложений.

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, РЕЗЕРВИРОВАНИЯ АВИАБИЛЕТА, ANDROID STUDIO, FIREBASE, VISUAL PARADIGM, JAVA, C#, RUP, UML

Объектом разработки является мобильное приложение для поддержки резервирования резервирования авиабилета.

Цель работы – разработка мобильного приложения процессов резервирования авиабилетов, позволяющего искать рейсы подходящие под критерии пользователя, резервировать и отменять билеты по рейсу.

Разработка серверной части происходила на языке программирования Java. В качестве среды разработки использовалась Android Studio. В качестве СУБД была использована СУБД Firebase.

Разработка приложения производилась на языке программирования Java. В качестве среды разработки использовалась среда программирования Android Studio, операционная система Windows 10. Приложение предназначено для использования под управлением ОС Android версии 5.0 и выше.

					ПГУ 09.03.04 – 08BP182.13 ПЗ			
Изм.	Лист	№ докум.	Подп.	Дата	Программное обеспечение мобильной поддержки процессов резервирования авиабилетов Пояснительная записка	Лит.	Лист	Листов
Разраб.		Милованцев В.С						
Пров.		Князев В.Н					3	86
Н. контр.		Такташкин Д.В				Группа 18ВП2		
Утверд.		Макарычев П.П						

Содержание

Введение.....	6
1 Анализ предметной области и требований	8
1.1 Анализ предметной области и постановка задачи.....	8
1.2 Анализ программных продуктов аналогичного назначения	9
1.2.1«Аэрофлот».....	9
1.2.2«Utair».....	10
1.2.3«Aviasales»	11
1.2.4Сравнение аналогов	13
1.3 Анализ целевой аудитории	13
1.4 Анализ функциональных требований.....	15
1.5 Анализ нефункциональных требований.....	19
1.6 Планирование разработки и оценка бюджета.....	20
2 Проектирование мобильного приложения	33
2.1 Выбор технологий и инструментальных средств разработки.....	33
2.2 Проектирование архитектуры программного обеспечения	34
2.3 Проектирование интерфейса пользователя.....	35
2.4 Концептуальное проектирование базы данных	41
2.5 Логическое проектирование базы данных	44
2.6 Проектирование серверной части приложения	46
3 Реализация мобильного приложения	47
3.1 Реализация базы данных	47
3.2 Разработка серверной части.....	47
3.3 Реализация клиентской части приложения	47
4 Тестирование программного продукта.....	50
4.1 Тестирование серверной части приложения	50
4.2 Тестирование клиентской части приложения	50
4.2.1Функциональное тестирование.....	50
4.3 Метрики кода.....	51
4.3.1Метрики размера программы.....	53
4.3.2Метрики Холстеда	53

Заключение	56
Список использованных источников	57
Приложение А. Глоссарий	60
Приложение Г. Исходный код мобильного приложения.....	62
Приложение Д. Результаты тестирования серверной части.....	79
Приложение Е. Результаты тестирования мобильного приложения.....	81

Введение

Информационные технологии и достижения вычислительной техники стремительно вторгаются в социальные, экономические и политические процессы общества. Поэтому в настоящее время их внедрение во все сферы деятельности человека является одной из основных задач. Для Российской Федерации это направление является также приоритетным, поэтому в 2017 году была принята Государственная программа «Цифровая экономика Российской Федерации», которая помогает «...созданию необходимых условий для развития цифровой экономики Российской Федерации, в которой данные в цифровой форме являются ключевым фактором производства во всех сферах социально-экономической деятельности, что повышает конкурентоспособность страны, качество жизни граждан, обеспечивает экономический рост» [1].

Кроме принятой программы, в том же году был подписан документ «Стратегия развития информационного общества в Российской Федерации на 2017 – 2030 годы», в котором были сформулированы основные «...цели, задачи и меры по реализации внутренней и внешней политики Российской Федерации в сфере применения информационных и коммуникационных технологий, направленные на развитие информационного общества, формирование национальной цифровой экономики...» [2].

Это означает, что программное обеспечение будет становиться все более доступным и востребованным, что способствует его внедрению в различные сферы человеческой деятельности.

Воздушный транспорт является одним из наиболее динамично развивающихся видов транспорта, чья значимость растет с каждым днем все больше и больше. Многим людям довольно часто необходимо перемещаться по разным городам, например: к родственникам, по работе или просто отдохнуть, но быстро и с удобством добраться до другого города, не говоря уже до другой страны, довольно трудно. Из-за этого авиаперевозки являются неотъемлемой частью жизни многих людей, они всегда будут необходимы.

В настоящее время у большинства людей присутствуют телефоны, поэтому мобильное приложение является доступным способом для донесения необходимой информации до клиента. За какую стоимость можно приобрести билет, сколько возможных рейсов и сколько осталось свободных мест, такую информацию удобнее и практичнее узнать из телефона, чем смотреть в интернете или же ехать в аэропорт. В современном мире значительное большинство людей хотят экономить своё время, поэтому существует необходимость именно в удобном, понятном приложении.

Люди стремятся к прогрессу и мобильности, поэтому мобильное приложение становится наилучшим выбором как для опытного, так и для нового клиента, который только осваивает новые границы.

Исходя из этого следует отметить, что тема разработки программного обеспечения мобильной поддержки процессов резервирования авиабилетов является актуальной и перспективной.

1 Анализ предметной области и требований

1.1 Анализ предметной области и постановка задачи

Разрабатываемое мобильное приложение должно обеспечить людям удобный доступ к покупке или бронированию авиабилета на выбранный ими рейс. Благодаря ему пользователям будет очень легко и удобно просматривать всю информацию по рейсу: время вылета и прилета, количество мест, цены на перелет. Клиентам не нужно будет лично приезжать в аэропорт, самим искать авиакомпанию или агентство, чтобы забронировать рейс. Достаточно будет иметь мобильное устройство с установленным приложением, где есть мобильный интернет или Wi-fi. В приложении необходимо будет указать все необходимые данные, такие как города, даты, время вылета и прилета, чтобы система вывела список всех рейсов, подходящих под критерии пользователя

В приложении должна быть удобная система поиска рейсов в соответствии с желаемыми пользователем критериями. Так же пользователю должна быть предоставлена возможность ввода личных данных для бронирования или покупки авиабилета. Если же пользователь был зарегистрирован и вошел в свой аккаунт, ему не придется вводить личные данные повторно; система должна сама заполнить данные для бронирования очередного рейса. Так же необходимо предусмотреть просмотр полной информации по рейсу, либо в личном кабинете пользователя, либо по номеру последних четырех цифр его билета. Так же пользователю необходимо предоставить отмену бронирования рейса. В случае если пользователь не может сам разобраться в приложении или возникла какая-либо ошибка, необходимо предоставить возможность связи с администрацией. Таким образом, в приложении должны быть представлены следующие основные функции:

- регистрация новых пользователей;
- вход зарегистрированных пользователей в систему;
- бронирование или покупка билета;

- просмотр информации по билету;
- отмена бронирования;
- связь с администрацией.

1.2 Анализ программных продуктов аналогичного назначения

В настоящее время существует множество мобильных приложений для бронирования билетов на авиарейс. Рассмотрим наиболее известные мобильные приложения такого рода и произведем для них сравнительный анализ.

1.2.1 «Аэрофлот»

Компания «Аэрофлот – Российские авиалинии» была создана в 1992 году для выполнения международных рейсов из московского аэропорта Шереметьево и аэропорта Красноярска. В начале 2010 года было принято решение о присоединении к «Аэрофлоту» всех государственных авиакомпаний страны. С 2012 года было создано мобильное приложение «Аэрофлот» [3]. Главная страница мобильного приложения «Аэрофлот» представлена на рисунке 1.

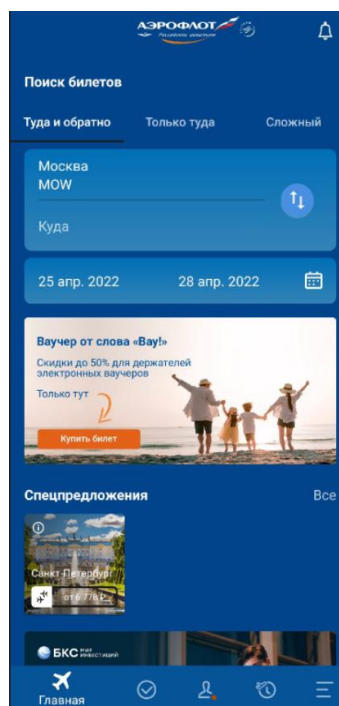


Рисунок 1 – Главная страница мобильного приложения «Аэрофлот»

На сегодняшний день пользователям приложения «Аэрофлот» доступны международные перевозки по всему миру, а отношение онлайн продаж билетов включая через мобильное приложение составляет более 47 процентов [4].

Приложение предоставляет удобный выбор маршрута перевозки, так же есть возможность заказа разных билетов (туда и обратно, только туда и маршрут с пересадками). К хорошим плюсам приложения можно отнести быструю регистрацию и просмотр статуса рейса. Так же приложение предоставляет возможность покупать и просматривать билеты на рейсы у дочерних или смежных компаний.

Недостатком приложения является его медленная обработка вводимых данных, постоянная повторная авторизация при выходе из приложения. Существенным же недостатком данного приложения является невозможность адаптации данных рейса к пользователю, а конкретнее при изменении личных данных аккаунта, билет можно будет только вернуть обратно, изменениям он не подвергнется, соответственно клиента не пустят на рейс.

1.2.2 «Utair»

Utair – является российской авиакомпанией основанной в 2002 году, занимающейся выполнением пассажирских и грузовых авиарейсов. По объему пассажиропотока входит в топ-5 в России. Авиакомпания Utair осуществляет перевозки по 150 направлениям, которые охватывают европейскую часть страны, Сибирь и Дальний Восток [5]. В 2019 году пассажиропоток этой компании составил более 7млн. человек и заняла 4 место в рейтинге российских авиакомпаний. Авиакомпания имеет крупнейший в мире по размерам и грузоподъемности действующий вертолётный флот, работающий как в России, так и в других странах и континентах [6]. Мобильное приложение появилось к 2013 году. Главный экран приложения представлен на рисунке 2.

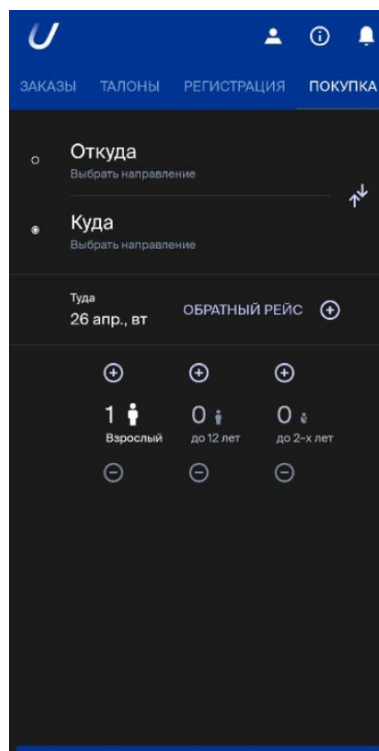


Рисунок 2 – Главный экран мобильного приложения «Utair»

Плюсами данного приложения является быстрый выбор количества необходимых билетов и быстротой и минимализмом интерфейса. Так же предусмотрена быстрая регистрация и возможность просмотреть грузовые или пассажирские перевозки. К тому же в данном приложении предусмотрено автозаполнение данных пользователя, если он был зарегистрирован или покупал ранее билеты (без учета личных данных, таких как данные паспорта и т.д.).

К минусам можно отнести невозможность выбора рейса, не относящейся к смежной компании, а так же невозможность выбора сложного маршрута. Некоторые перевозки мобильное приложение просто не отображает, хотя их можно найти на их же сайте, к тому же была замечена медленная обработка поиска необходимого рейса или его поиска. Бронирование билета в принципе невозможно.

1.2.3 «Aviasales»

Aviasales является метапоисковиком авиабилетов, существующим с 2007 года. Ежемесячная аудитория которой составляет примерно 15млн.

человек. Aviasales является поисковиком, в следствии чего не продает билета, а переносит пользователей на сайты авиакомпаний [7]. С 2012 года у компании появилось мобильное приложение, главный экран которого представлен на рисунке 3.

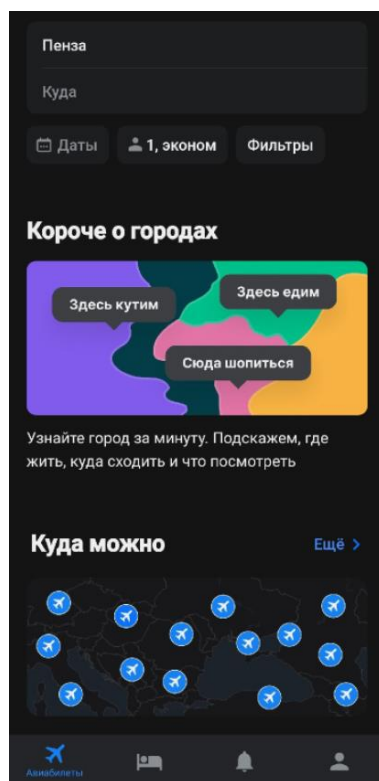


Рисунок 3 – Главная экран мобильного приложения «Aviasales»

Плюсом и минусом данного приложения является просмотр всех рейсов разных авиакомпаний. Минусом же этого является то, что Aviasales решает рейсы каких компании они будут показывать. В данном приложении нельзя купить или забронировать билет, а именно пользователю предоставляется возможность перейти на мобильную версию сайта авиакомпании, чей рейс он выбрал. Данное приложение создано лишь для просмотра рейсов, по необходимым критериям пользователя.

Плюсами приложения является быстрая обработка данных и просмотр множества авиакомпаний, что позволяет пользователю сэкономить и сделать лучший выбор.

1.2.4 Сравнение аналогов

Сравнительный анализ рассмотренных выше мобильных приложений представлен в таблице 1.

Таблица 1 – Сравнение аналогов

Мобильное приложение	Бронирование билета	Покупка билета	Просмотр всех рейсов	Выбор сложного маршрута	Автозаполнение данных
Аэрофлот	Да	Да	Присутствует	Есть	Нет
Utair	Нет	Да	Отсутствует	Нет	Да
Aviasales	Нет	Нет	Присутствует	Есть	Нет
Разрабатываемое приложение	Да	Да	Присутствует	Есть	Да

В результате анализа систем аналогичного назначения было выявлено, что в целом рассмотренные приложения соответствуют запросам пользователей, но некоторые удобные и полезные функции в них отсутствуют, таким образом было принято решение о разработке собственного приложения, которое будет полностью удовлетворять запросам пользователей.

1.3 Анализ целевой аудитории

Целевой аудиторией проекта являются люди, которым необходимо попасть в другой город по работе, навестить родных или же на отдых, которые используют мобильное приложение для бронирования или покупки билета на авиарейс. В настоящее время число пользователей авиакомпаний стремительно возрастает. В частности, подобный вывод можно сделать на основании статистики изменения числа пассажиров и пассажиропотока нашей страны на сайте Aviastat, который берет информацию из Росстата и представляет ее в виде графиков [8, 9]. График, на котором показана динамика изменения числа пассажиров и пассажирооборота в России в 1980 – 2019 годах, представлен на рисунке 4.

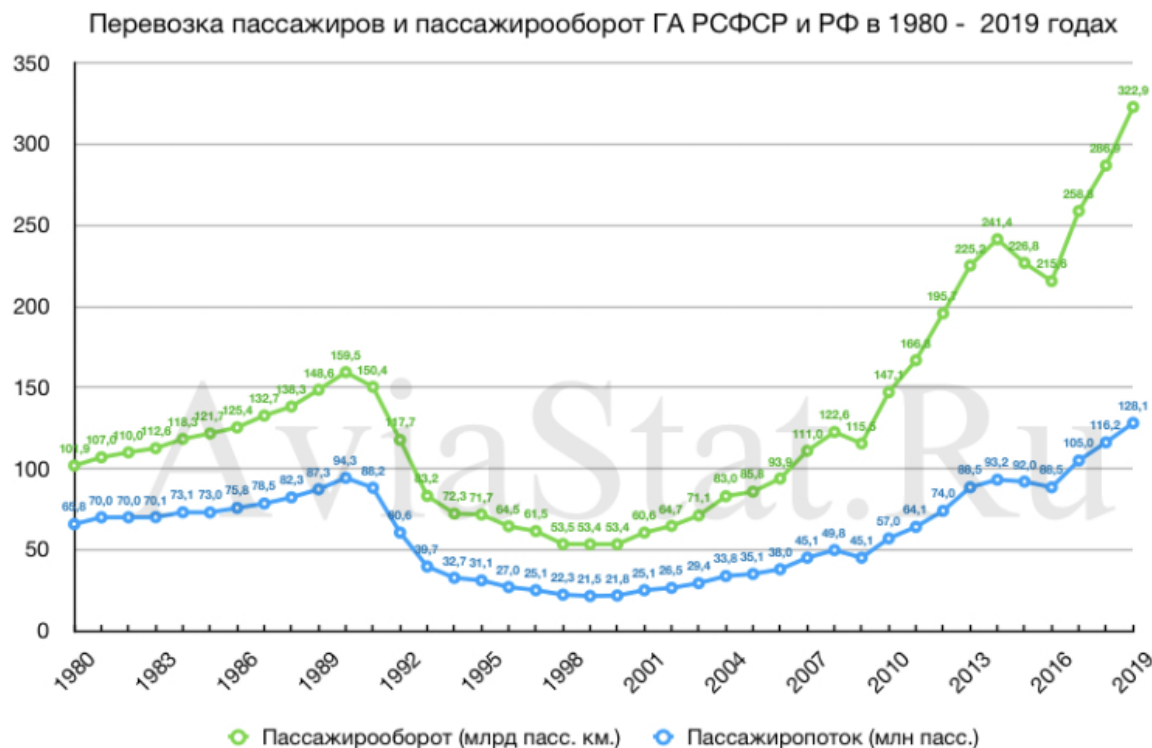


Рисунок 4 – Динамика пассажирских перевозок

Как можно увидеть, в истории авиаперевозок были периоды временного падения, однако в целом в последнее время число пассажиров и пассажирооборота стремительно растет. На рисунке 5 представлены доли производителей смартфонов на мировом рынке за последние 7 лет [10].

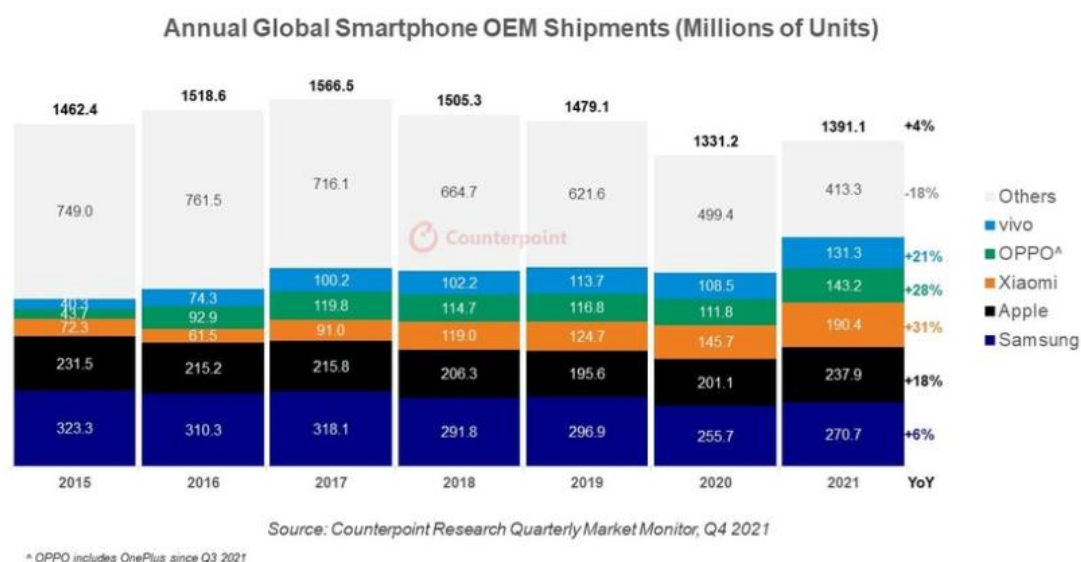


Рисунок 5 – Доли производителей смартфонов на мировом рынке

Можно заметить, что практически все основные производители смартфонов – Samsung, Xiaomi, Huawei и т.д. – используют операционную

систему Android. Единственное значимое исключение – компания Apple, использующая собственную систему iOS. Но, при всей популярности Apple, ее продукция даже в периоды пикового спроса составляет лишь в районе 20% от мирового рынка, что в несколько раз меньше, чем доля устройств с Android. Поэтому наиболее выгодным решением с точки зрения доступности продукта для пользователей будет реализация разрабатываемого приложения именно для операционной системы Android. Статистика распространения разных версий системы Android представлена на рисунке 6 [11].

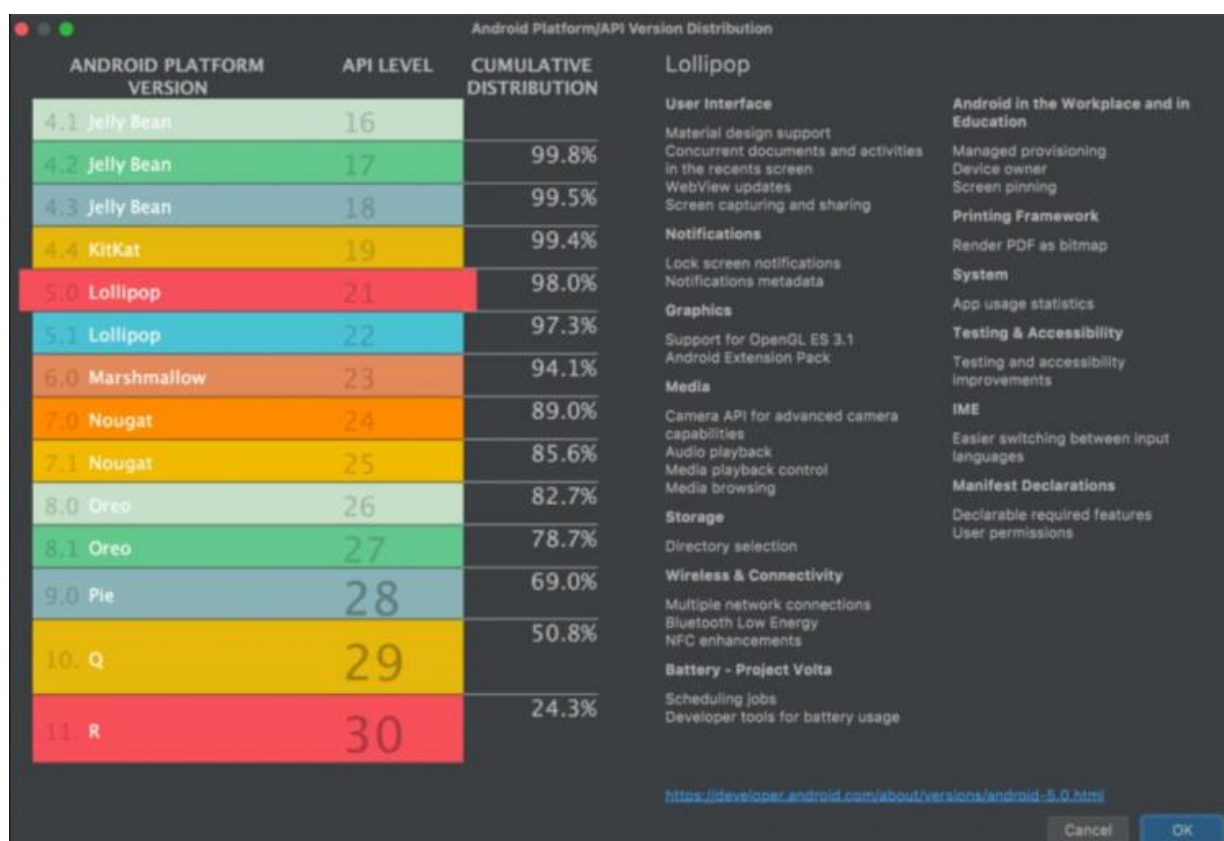


Рисунок 6 – Распространение различных версий системы Android

Можно увидеть, что 98 % пользователей системы Android использует версии не старше 5.0. Поэтому приложение будет разрабатываться таким образом, чтобы оно успешно функционировало для Android 5.0 и более новых версий системы.

1.4 Анализ функциональных требований

Поскольку основным назначением мобильного приложения резервирования авиабилетов является работа с рейсами, именно рейсы будут

являться основными объектами системы. Для каждой из них, помимо мест назначения и времени отправки, будет храниться дополнительная информация: самолет, пересадка, номер терминала отправки и т.д.

Пользователь в системе должен иметь возможность бронирования билета, а так же просмотр информации по своему билету. К тому же ему должна быть предоставлена возможность взаимодействия с личным кабинетом, такое как вход в аккаунт и регистрация. Так же необходимо предусмотреть возможность отмены купленного или забронированного билета и связь с администрацией.

Менеджер должен иметь возможность проверять данные клиента, а так же добавлять его в базу данных.

В результате анализа требований задания была создана диаграмма вариантов использования (рисунок 7).



Рисунок 7 – Диаграмма вариантов использования мобильного приложения

Составим спецификацию варианта использования «Зарегистрировать аккаунт», представленную на рисунке 8.

Название: зарегистрировать аккаунт
Назначение: регистрация нового аккаунта пользователя
Актеры: пользователь, СУБД, менеджер
Предусловие: приложение запущено
Основной сценарий: <ol style="list-style-type: none"> 1. Пользователь выбирает регистрацию 2. Пользователь вводит личные данные 3. Пользователь вводит почту и пароль 4. Система проверяет данные на корректность 5. Если все данные корректны <ol style="list-style-type: none"> 5.1. Система запрашивает у СУБД сведения о наличии аккаунта с введенной почтой. 5.2. СУБД передает сведения о наличии аккаунта с введенным логином. 5.3. Если такого аккаунта нет <ol style="list-style-type: none"> 5.3.1. СУБД добавляет в БД информацию о новом пользователе. 5.3.2. Система осуществляет вход нового пользователя. 5.4. Иначе система выводит сообщение о том, что выбранная почта занята. 6. Иначе система выводит сообщение о недопустимом логине или пароле
Постусловие: регистрация выполнена

Рисунок 8 – Спецификация варианта использования «Зарегистрировать аккаунт»

Составим спецификацию варианта использования «Войти в аккаунт», представленную на рисунке 9.

Название: войти в аккаунт
Назначение: вход в систему с существующего аккаунта пользователя
Актеры: пользователь, СУБД
Предусловие: приложение запущено
Основной сценарий: <ol style="list-style-type: none"> 1. Пользователь выбирает войти в аккаунт 2. Пользователь вводит почту и пароль 3. Система запрашивает у СУБД сведения о наличии аккаунта с введенной почтой и паролем 4. СУБД передает сведения о наличии аккаунта с такой почтой и паролем 5. Если такой аккаунт существует <ol style="list-style-type: none"> 5.1. Система осуществляет вход нового пользователя 5.2. Система уведомляет пользователя о входе 6. Иначе система сообщает пользователю о неверной почте или пароле.
Постусловие: вход в систему выполнен

Рисунок 9 – Спецификация варианта использования «Войти в аккаунт»

Составим спецификацию варианта использования «Получить информацию о рейсе», представленную на рисунке 10.

Название: получить информацию о рейсе
Назначение: получение полной информации о рейсе
Актеры: пользователь, СУБД
Предусловие: билет куплен или забронирован
Основной сценарий: 1. Пользователь выбирает получить информацию о рейсе 2. Система предлагает пользователю ввести последние четыре цифры номера билета 3. Если такой билет найден 3.1. СУБД выдает полную информацию по билету, такое как: номер самолета, номер места, цена билета, город прибытия, город отправки, время и т.д. 4. Иначе система выдает ошибку, что такой рейс не найден
Постусловие:

Рисунок 10 – Спецификация варианта использования «Получить информацию о рейсе»

Рассмотрим некоторые варианты использования поподробнее. Для варианта использования «Войти в аккаунт» составлена диаграмма деятельности, представленная на рисунке 11.

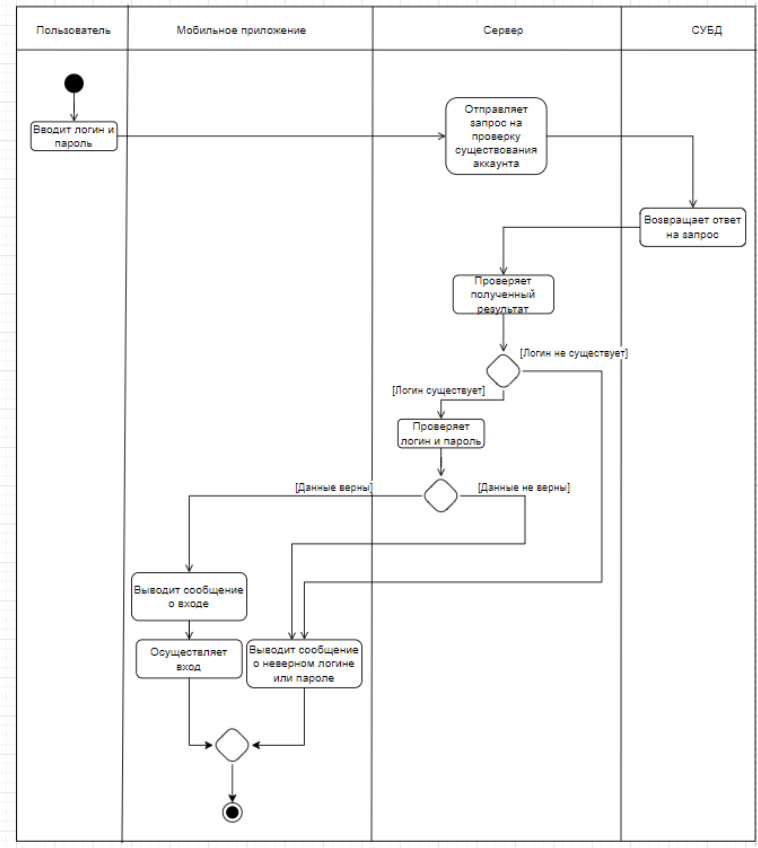


Рисунок 11 – Диаграмма деятельности варианта использования «Войти в аккаунт»

Для варианта использования «Получить информацию о рейсе» была составлена диаграмма последовательности, представленная на рисунке 12.

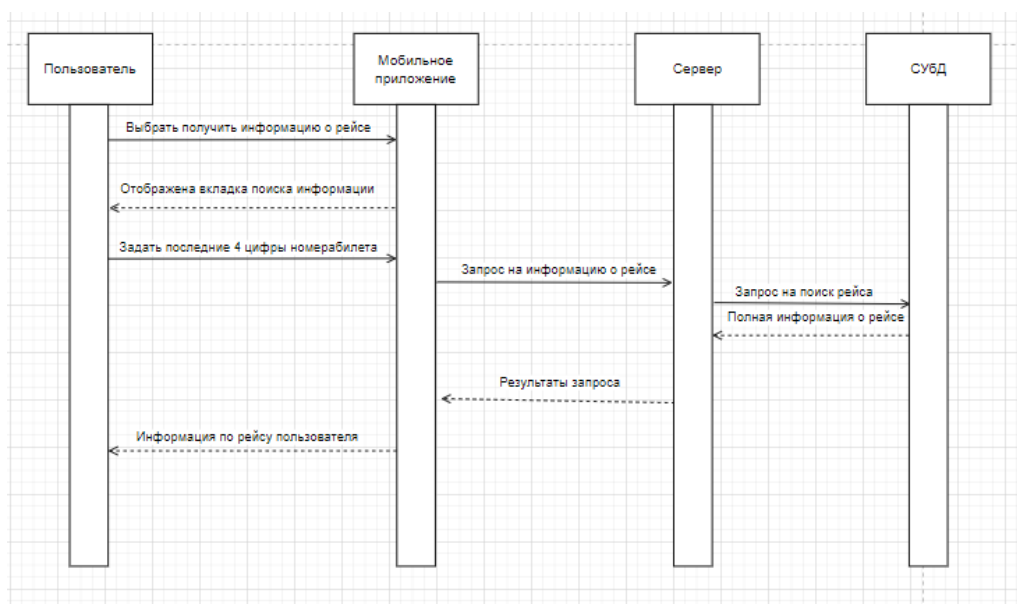


Рисунок 12 – Диаграмма последовательности варианта использования «Получить информацию о рейсе»

В результате анализа функциональных требований были успешно выявлены основные функции разрабатываемого приложения. При этом для разрабатываемого приложения были построены диаграммы вариантов использования, деятельности и последовательности, а также для некоторых выявленных вариантов использования были составлены спецификации.

1.5 Анализ нефункциональных требований

Разработка серверной части приложения должна быть выполнена на языке программирования Java [12], в среде программирования Android Studio. Данные на сервере хранятся в базе данных Firebase Realtime Database, реализованной с использованием СУБД Firebase [13, 14], запросы к базе данных осуществляются на языке Java.

Клиентская часть приложения должна быть реализована на языке программирования Java с использованием среды программирования Android Studio. Она должна представлять собой мобильное приложение с удобным и интуитивно понятным пользовательским интерфейсом. Клиентское

приложение должно быть отлажено и должно успешно работать под управлением ОС Android 5.0 и выше.

1.6 Планирование разработки и оценка бюджета

Высококачественное выполнение работ по любому проекту, в том числе и в сфере информационных технологий, невозможно при нехватке грамотного планирования, которое позволяет обеспечить выполнение работ над проектом таким образом, чтобы достичь исполнения всех поставленных целей за минимально возможное время, с минимально возможными затратами и с приемлемым качеством [15 – 19].

Задачи, представляющие собой этапы работы над проектом, представлены в таблице 2.

Таблица 2 – Перечень задач проекта

№	Название задачи	Длительность	Предшественники
1	Начало проекта	-	-
2	Анализ требований	-	-
3	Анализ аналогов	10 дней	1
4	Анализ рейсов аэропорта	8 дней	1
5	Обсуждение принятых решений с руководством	3 дня	3; 4
6	Принятие окончательного решения	1 день	5
7	Анализ требований завершен	-	6
8	Выбор средств реализации проекта	-	-
9	Анализ возможных программных средств	5 дня	1
10	Выбор программных средств	3 дня	9
11	Выбор средств завершен	-	10
12	Проектирование приложения	-	-
13	Проектирование базы данных	6 дней	11; 7
14	Проектирование архитектуры приложения	7 дней	13
15	Проектирование пользовательского интерфейса	5 дней	7; 11

Продолжение таблицы 2

№	Название задачи	Длительность	Предшественники
16	Проектирование приложения завершено	-	15; 14
17	Реализация приложения	-	-
18	Реализация базы данных	15 дней	16
19	Реализация серверной части	12 дней	18
20	Реализация клиентской части	14 дней	19
21	Реализация завершена	-	20
22	Тестирование	-	-
23	Тестирование серверной части	4 дней	21
24	Тестирование клиентской части	5 дней	21
25	Исправление ошибок	2 дня	23; 24
26	Тестирование завершено	-	25
27	Внедрение	-	-
28	Развертывание приложения на сервере	2 дня	26
29	Внесение рейсов	6 дней	28
30	Поиск ошибок	9 дней	29
31	Внедрение завершено	-	30
32	Конец проекта	-	31

Таким образом, для реализации проекта нужно выполнить 32 задачи. В таблице 3 представлено описание затрачиваемых ресурсов.

Таблица 3 – Список ресурсов проекта

Название ресурса	Тип	Стандартная ставка	Ставка сверхурочных
Программист1	Трудовой	300 Р/ч	450 Р/ч
Программист2	Трудовой	250 Р/ч	330 Р/ч
Дизайнер	Трудовой	200 Р/ч	300 Р/ч
Интернет	Затраты		
Компьютер	Материальный	1000 Р	
Мобильное устройство	Материальный	400 Р	

Распределение ресурсов по задачам представлено в таблице 4.

Таблица 4 – Назначение ресурсов

№	Название задачи	Ресурс	Единицы
1	Начало проекта		
2	Анализ требований		
3	Анализ аналогов	Программист1 Компьютер Интернет	50% 1 шт
4	Анализ рейсов аэропорта	Программист1 Программист2 Компьютер Интернет	50% 50% 2 шт
5	Обсуждение принятых решений с руководством	Программист1 Программист2 Дизайнер	100% 100% 100%
6	Принятие окончательного решения	Программист1 Программист2	100% 100%
7	Анализ требований завершен		
8	Выбор средств реализации проекта		
9	Анализ возможных программных средств	Программист1 Программист2 Компьютер Интернет	50% 50% 2 шт
10	Выбор программных средств	Программист1 Программист2 Компьютер Интернет	50% 50% 1 шт
11	Выбор средств завершен		
12	Проектирование приложения		
13	Проектирование базы данных	Программист1 Программист2 Компьютер Интернет	25% 70% 2 шт
14	Проектирование архитектуры приложения	Программист1 Программист2 Компьютер Интернет	100% 100% 2 шт
15	Проектирование пользовательского интерфейса	Программист1 Программист2 Дизайнер Компьютер Интернет	70% 50% 100% 3 шт.
16	Проектирование приложения завершено		

Продолжение таблицы 4

№	Название задачи	Ресурс	Единицы
17	Реализация проекта		
18	Реализация базы данных	Программист1 Программист2 Компьютер Интернет	100% 100% 2 шт.
19	Реализация серверной части	Программист1 Программист2 Компьютер Интернет	100% 100% 2 шт.
20	Реализация клиентской части	Программист1 Программист2 Дизайнер Компьютер Мобильное устройство Интернет	100% 100% 100% 4 шт. 2 шт.
21	Реализация завершена		
22	Тестирование		
23	Тестирование серверной части	Программист1 Программист2 Компьютер Интернет	60% 70% 2 шт.
24	Тестирование клиентской части	Программист1 Программист2 Компьютер Мобильное устройство Интернет	60% 40% 2 шт. 2 шт.
25	Исправление ошибок	Программист1 Программист2 Дизайнер Компьютер Мобильное устройство Интернет	100% 100% 100% 3 шт. 3 шт.
26	Тестирование завершено		
27	Внедрение		
28	Развертывание приложения на сервере	Программист1 Программист2 Компьютер Интернет	100% 100% 1 шт.
29	Внесение рейсов	Программист1 Программист2 Компьютер Интернет	50% 50% 2 шт.
30	Поиск ошибок	Программист1 Программист2 Компьютер Мобильное устройство Интернет	100% 100% 2 шт. 2 шт.

Окончание таблицы 4

№	Название задачи	Ресурс	Единицы
31	Внедрение завершено		
32	Конец проекта		

Таким образом, реализацией проекта будут заниматься 2 программиста и 1 дизайнер, которым для работы нужны компьютеры, мобильные устройства и доступ к интернету. Для вычисления длительности реализации проекта и его критического пути построен сетевой график (рисунок 13).

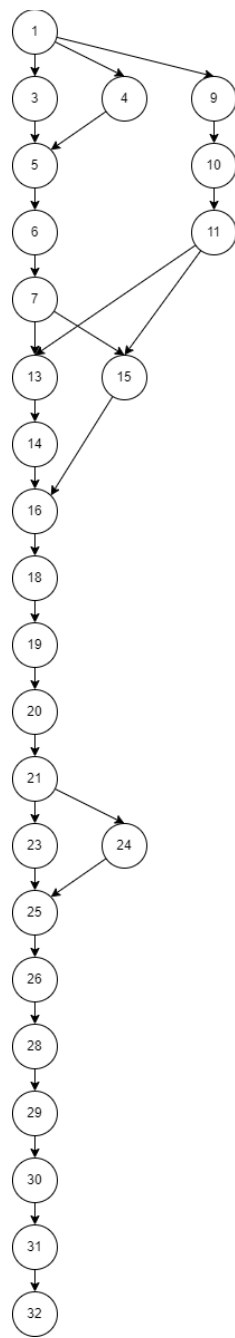


Рисунок 13 – Сетевой график проекта

Рассчитаем вручную критический путь проекта. В таблице 5 представлены расчеты раннего времени начала работ для его задач.

Таблица 5 – Расчеты раннего времени начала работ

Работа	Предшественники			Вычисление	Раннее время начала
	№	Раннее время начала	Длительность		
1	-	-	-	-	0
3	1	0	0	0+0	0
4	1	0	0	0+0	0
5	3	0	14	Max(0+14;0+8)	14
	4	0	8		
6	5	14	3	14+3	17
7	6	17	1	17+1	18
9	1	0	0	0+0	0
10	9	0	5	0+5	5
11	10	5	3	5+3	8
13	7	18	0	Max(18+0;8+0)	18
	11	8	0		
14	13	18	6	18+6	24
15	7	18	0	Max(18+0;8+0)	18
	11	8	0		
16	14	24	7	Max(24+7;18+5)	31
	15	18	5		
18	16	31	0	31+0	31
19	18	31	15	31+15	46
20	19	46	12	46+12	58
21	20	58	14	58+14	72
23	21	72	0	72+0	72
24	21	72	0	72+0	72
25	23	72	4	Max(72+4;72+5)	77
	24	72	5		
26	25	77	2	77+2	79
28	26	79	0	79+0	79
29	28	79	2	79+2	81
30	29	81	6	81+6	87
31	30	87	9	87+9	96
32	31	96	0	96+0	96

В таблице 6 рассчитано позднее время начала работ задач.

Таблица 6 – Расчеты позднего времени начала работ

Рабо- та	Последователи			Вычисление	Позднее время начала
	№	Позднее время начала	Длительность		
32	-	-	-	-	96
31	32	96	0	96-0	96
30	31	96	9	96-9	87
29	30	87	6	87-6	81
28	29	81	2	81-2	79
26	28	79	0	79-0	79
25	26	79	2	79-2	77
24	25	77	5	77-5	72
23	25	77	4	77-4	73
21	24 23	72 73	0	Min(72-0;73-0)	72
20	21	72	14	72-14	58
19	20	58	12	58-12	46
18	19	46	15	46-15	31
16	18	31	0	31-0	31
15	16	31	5	31-5	26
14	16	31	7	31-7	24
13	14	24	6	24-6	18
11	15 13	26 18	0	Min(26-0;18-0)	18
10	11	18	3	18-3	15
9	10	15	5	15-5	10
7	15 13	26 18	0	Min(26-0;18-0)	18
6	7	18	1	18-1	17
5	6	17	3	17-3	14
4	5	14	8	14-8	6
3	5	14	14	14-14	0
1	9 4 1	10 6 0	0	Min(10-0,6-0,0-0)	0

На основе найденных значений раннего и позднего времени начала работ можно найти критический путь. Соответствующие вычисления представлены в таблице 7.

Таблица 7 – Вычисление критического пути

Работа №	Раннее время начала	Позднее время начала	Резерв времени
1	0	0	0
3	0	0	0
4	0	6	6
5	14	14	0
6	17	17	0
7	18	18	0
9	0	10	10
10	5	15	10
11	8	18	10
13	18	18	0
14	24	24	0
15	18	26	8
16	31	31	0
18	31	31	0
19	46	46	0
20	58	58	0
21	72	72	0
23	72	73	1
24	72	72	0
25	77	77	0
26	79	79	0
28	79	79	0
29	81	81	0
30	87	87	0
31	96	96	0
32	96	96	0

В результате ручного вычисления критического пути проекта, представленного в таблице 7, можно сделать вывод о том, что критический путь проходит через работы с номерами 1, 3, 5, 6, 7, 13, 14, 15, 16, 18, 20, 21, 24, 25, 26, 28, 29, 30, 31, 32. Общее время выполнения проекта будет составлять 96 дней.

На рисунке 14 представлен граф проекта, на котором прямыми линиями указан критический путь проекта.

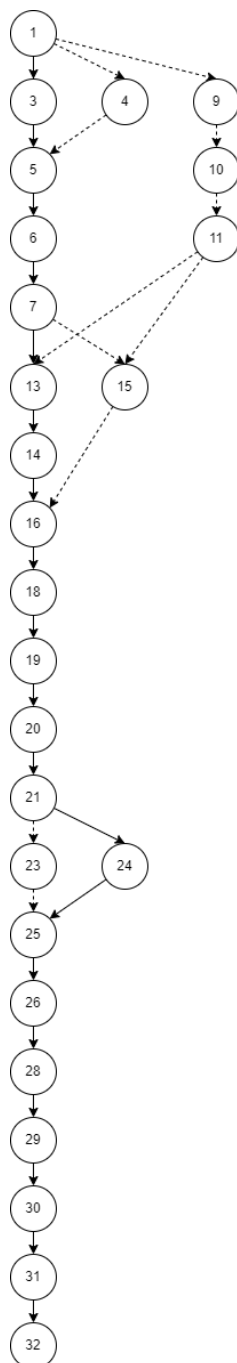


Рисунок 14 – Критический путь проекта

Для того, чтобы проверить корректность расчетов продолжительности проекта, а также вычислить величину его бюджета, воспользуемся системой управления проектами Microsoft Project, которая позволяет автоматически произвести все необходимые вычисления [20 – 23]. Список задач из таблицы 3, внесенный в систему Microsoft Project, представлен на рисунке 15. В качестве времени начала проекта была принята дата 17.01.2022.

	Название задачи	Длительность	Начало	Окончание	Предшественники
1	Начало проекта	0 дней	Пн 17.01.22	Пн 17.01.22	
2	Анализ требований	14 дней	Пн 17.01.22	Чт 03.02.22	
3	Анализ аналогов	10 дней	Пн 17.01.22	Пт 28.01.22	1
4	Анализ рейсов аэропорта	8 дней	Пн 17.01.22	Ср 26.01.22	1
5	Обсуждение принятых решений с руководством	3 дня	Пн 31.01.22	Ср 02.02.22	3,4
6	Принятие окончательного решения	1 день	Чт 03.02.22	Чт 03.02.22	5
7	Анализ требований завершен	0 дней	Чт 03.02.22	Чт 03.02.22	6
8	Выбор средств реализации проекта	8 дней	Пн 17.01.22	Ср 26.01.22	
9	Анализ возможных программных средств	5 дней	Пн 17.01.22	Пт 21.01.22	1
10	Выбор программных средств	3 дня	Пн 24.01.22	Ср 26.01.22	9
11	Выбор средств завершен	0 дней	Ср 26.01.22	Ср 26.01.22	10
12	Проектирование приложения	13 дней	Пт 04.02.22	Вт 22.02.22	
13	Проектирование базы данных	6 дней	Пт 04.02.22	Пт 11.02.22	11,7
14	Проектирование архитектуры приложения	7 дней	Пн 14.02.22	Вт 22.02.22	13
15	Проектирование пользовательского интерфейса	5 дней	Пт 04.02.22	Чт 10.02.22	7,11
16	Проектирование приложения завершенно	0 дней	Вт 22.02.22	Вт 22.02.22	15,14
17	Реализация приложения	41 дней	Ср 23.02.22	Ср 20.04.22	
18	Реализация базы данных	15 дней	Ср 23.02.22	Вт 15.03.22	16
19	Реализация серверной части	12 дней	Ср 16.03.22	Чт 31.03.22	18
20	Реализация клиентской части	14 дней	Пт 01.04.22	Ср 20.04.22	19
21	Реализация завершенна	0 дней	Ср 20.04.22	Ср 20.04.22	20
22	Тестирование	7 дней	Чт 21.04.22	Пт 29.04.22	
23	Тестирование серверной части	4 дня	Чт 21.04.22	Вт 26.04.22	21
24	Тестирование клиентской части	5 дней	Чт 21.04.22	Ср 27.04.22	21
25	Исправление ошибок	2 дня	Чт 28.04.22	Пт 29.04.22	23,24
26	Тестирование завершенно	0 дней	Пт 29.04.22	Пт 29.04.22	25
27	Внедрение	17 дней	Пн 02.05.22	Вт 24.05.22	
28	Развертывание приложения на сервере	2 дня	Пн 02.05.22	Вт 03.05.22	26
29	Внесение рейсов	6 дней	Ср 04.05.22	Ср 11.05.22	28
30	Поиск ошибок	9 дней	Чт 12.05.22	Вт 24.05.22	29
31	Внедрение завершенно	0 дней	Вт 24.05.22	Вт 24.05.22	30
32	Конец проекта	0 дней	Вт 24.05.22	Вт 24.05.22	31

Рисунок 15 – Задачи проекта в среде Microsoft Project

В среде Microsoft Project на основе введенных данных была построена диаграмма Ганта. Можно заметить, что вычисленный программой критический путь совпал с критическим путем, вычисленным вручную. Диаграмма Ганта представлена на рисунке 16.

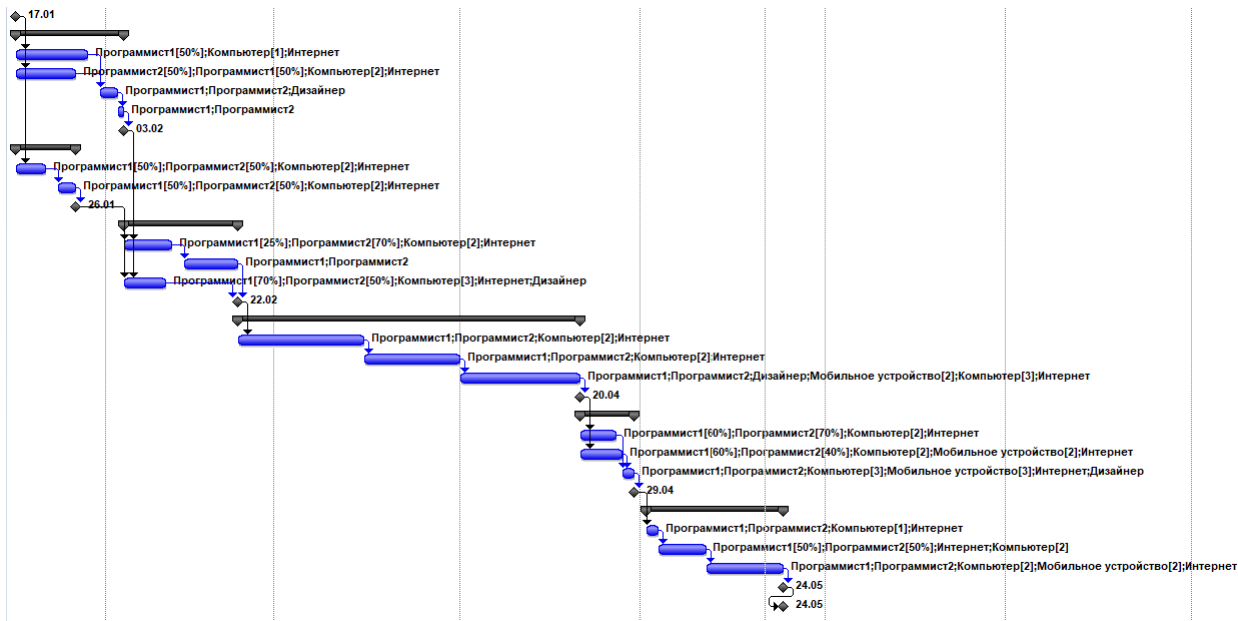


Рисунок 16 – Диаграмма Ганта

На рисунках 17 и 18 представлены графики трудовых ресурсов «Программист1» и «Программист2».

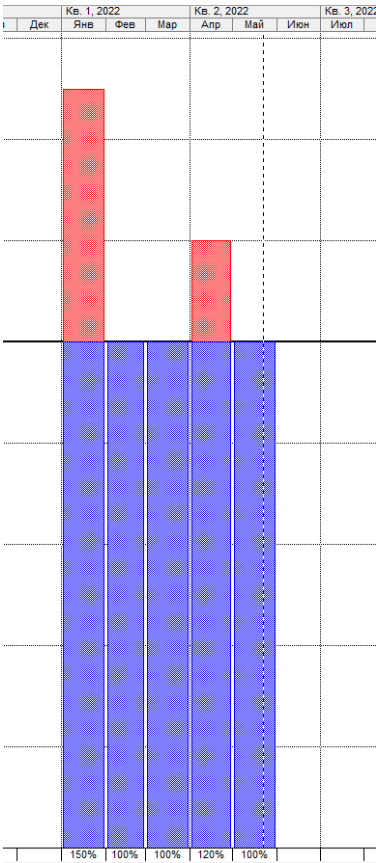


Рисунок 17 – График ресурсов для ресурса «Программист1»

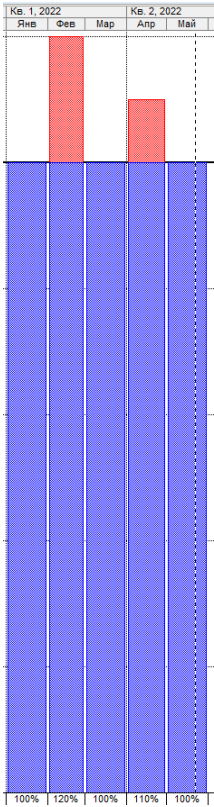


Рисунок 18 – График ресурсов для ресурса «Программист2»

На рисунке 19 представлен график трудового ресурса «Дизайнер».

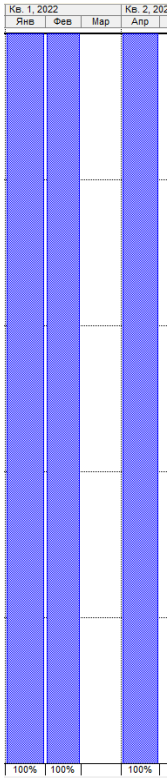


Рисунок 19 – График ресурсов для ресурса «Дизайнер»

На рисунке 20 представлен график материального ресурса «Компьютер».

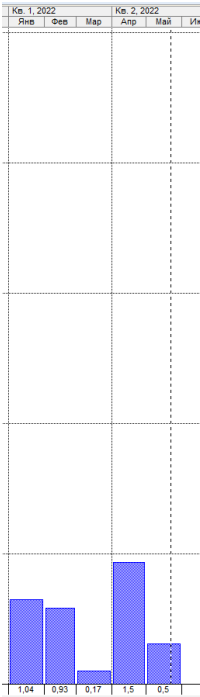


Рисунок 20 – График ресурсов для ресурса «Компьютер»

Как можно заметить, у ресурсов «Программист1», «Программист2» в январе, феврале и апреле наблюдается перегрузка.

На рисунке 21 представлен обзор проекта. Из рисунка видно, что общая длительность проекта составляет 92 дня, с 17.01.22. по 24.05.22. Предполагаемые затраты составят 462 560 руб.

Рейсы_проект			
на дату Вс 22.05.22			
Даты			
Начало:	Пн 17.01.22	Окончание:	Вт 24.05.22
Баз. начало:	НД	Баз. окончание:	НД
Факт. начало:	НД	Факт. окончание:	НД
Отклон. начала:	0 дней	Отклон. окончания:	0 дней
Длительность			
Текущий план:	92 дней	Оставшаяся:	92 дней
Базовый план:	0 дней?	Фактическая:	0 дней
Отклонение:	92 дней	Процент завершения:	0%
Трудозатраты			
Текущий план:	1 607,2 ч	Оставшиеся:	1 607,2 ч
Базовый план:	0 ч	Фактические:	0 ч
Отклонение:	1 607,2 ч	Процент завершения:	0%
Затраты			
Текущий план:	462 560,00 Р	Оставшиеся:	462 560,00 Р
Базовый план:	0,00 Р	Фактические:	0,00 Р
Отклонение:	462 560,00 Р		
Состояние задачи		Состояние ресурса	
Еще не начатые задачи	32	Трудовые ресурсы:	2
Выполняющиеся задачи	0	Трудовые ресурсы с прев. дост	2
Завершенные задачи:	0	Материальные ресурсы:	2
Итого задач:	32	Итого ресурсов:	6

Рисунок 21 – Обзорный отчет проекта

На основе произведенных оценок стоимости и сроков проекта можно сделать вывод о его экономической целесообразности.

2 Проектирование мобильного приложения

2.1 Выбор технологий и инструментальных средств разработки

В процессе разработки приложения было решено применять методику RUP. Разработка приложения в соответствии с методологией RUP носит итеративный характер. На каждой итерации перед разработчиками ставятся определенные цели, которые обязаны быть достигнуты к ее концу. В итоге в конце каждой итерации должна получиться промежуточная функциональная версия разрабатываемого ПО. В методологии RUP существуют 4 основных фазы, каждая из которых может включать в себя одну либо некоторое количество итераций [24]:

- фаза начала проекта – формируется общее видение проекта и экономическое обоснование;
- уточнение – анализ предметной области и построение архитектуры продукта;
- построение – реализация основного функционала продукта;
- внедрение – получение финальной версии продукта и его передача заказчику, фаза включает в себя бета-тестирование и устранение выявленных в его процессе ошибок, а также обучение пользователей.

Каждая фаза завершается вехой, на которой команда разработчиков оценивает осуществление поставленных целей и принимает критически важные решения о ходе последующей работы над продуктом.

Выбор в пользу данной методологии продиктован рядом ее преимуществ. С одной стороны, методология RUP обеспечивает гибкость процесса разработки и разрешает быстро вносить коррективы в проект в случае обнаружения в нем ошибок либо видоизменений требований к разрабатываемому приложению. Помимо того, благодаря предоставленной методологии получается выявить основные риски уже на ранних стадиях разработки. При этом RUP, в отличие от некоторых других знаменитых методологий, к примеру XP, не требует непрерывного взаимодействия с заказчиком, которое в рамках данного проектного плана нереально.

Для анализа требований, а также моделирования структуры проекта в соответствии с принципами RUP было решено применять диаграммы на языке моделирования UML [25]. Для построения диаграмм применялось средство UML-моделирования ArgoUML.

На этапе планирования проекта было решено применять приложение Microsoft Project. Оно позволяет обеспечить эффективное управление проектом на всех стадиях его выполнения. Microsoft Project предоставляет возможность исполнить структуризацию проекта, составить его календарный план, а еще рассчитать затраты трудовых, материальных и денежных ресурсов в процессе работы над проектом.

Для создания мобильного приложения было решено применять среду программирования Android Studio, предназначенную для разработки программ для операционной системы Android, и язык программирования Java. Серверную часть было решено разрабатывать на языке Java с применением среды программирования Android Studio. Для хранения данных используется БД, работающая под управлением СУБД Firebase. Проектирование базы данных выполнялось с применением приложения Visual Paradigm.

2.2 Проектирование архитектуры программного обеспечения

Было принято решение использовать бессерверную архитектуру[26]. Ее схема представлена на рисунке 22.

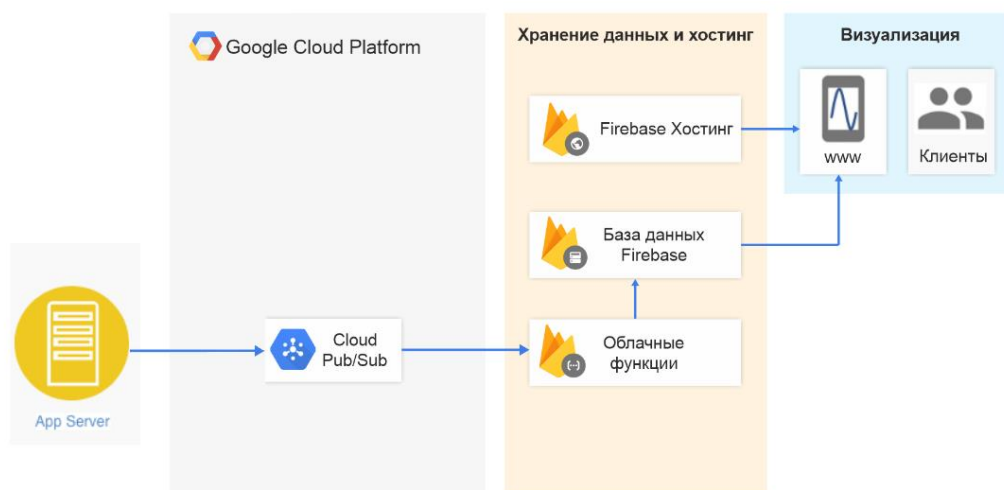


Рисунок 22 – Бессерверная архитектура

В данной архитектуре база данных располагается на облачном сервисе, СУБД, которое отвечает за взаимодействие с базой данных, встроено в клиентскую часть.

Клиентская же часть разрабатываемого ПО обеспечивает для пользователя удобный интерфейс, позволяющий пользователю просматривать необходимые ему данные из БД и взаимодействовать с ними, а также реализует возможность удаленно обращаться к серверу. В нашем случае его роль выполняет мобильное приложение.

Взаимодействие клиента и сервера осуществляется следующим образом: клиент делает запрос серверу. Серверная часть приложения, в свою очередь, на основе полученного запроса создает запрос к базе данных. СУБД обрабатывает запрос и возвращает его результаты серверу, который на основе полученных результатов формирует ответ на запрос клиента. Получив ответ, клиент представляет его пользователю в удобной для того форме.

Достоинством архитектуры является то, что есть возможность сосредоточиться на основной клиентской части, не заботясь об управлении сервером и среде его выполнения, так как все обслуживание происходит в облаке. Это подразумевает использование всех возможностей серверных вычислений без необходимости управления инфраструктурой и ее оптимизации ради масштабирования. Так же это означает, что безопасность на уровне инфраструктуры полностью ложится на облачный сервис.

2.3 Проектирование интерфейса пользователя

На основе выявленных функциональных требований был разработан интерфейс мобильного приложения. Он включает в себя страницы входа в личный кабинет, регистрации, личный кабинет пользователя, поиска рейсов, результатов поиска, выбора мест в самолете, бронирования билетов, поиска информации о рейсе, результатов поиска информации о рейсе.

Главная страница приложения представлена на рисунке 23. На ней имеются следующие элементы: 1 – кнопка для входа в личный кабинет пользователя, 2 – кнопка для поиска рейса по номеру билета, 3 – кнопка для

отмены билета, 4 – поля для ввода маршрута перелета, 5 – поля для ввода даты и время начала и конца перелета, 6 – кнопка перехода на страницу результатов подбора рейсов по выбранному маршруту.

The diagram illustrates the layout of a flight search page. At the top, there is a navigation bar with three buttons: 'Личный кабинет' (Personal Cabinet), 'Найти' (Find), and 'Отменить' (Cancel). Above the 'Личный кабинет' button is label 1, and above the 'Найти' button is label 2. A line from label 3 points to the 'Отменить' button. Below the navigation bar is a large rectangular area containing several input fields and buttons. In the middle of this area, there are three buttons labeled 'Откуда' (From), 'Куда' (To), and 'Сложный' (Complex), with a line from label 4 pointing to the 'Сложный' button. Below these are two more buttons: 'Дата и время перелета' (Flight date and time) and 'Обратный билет' (Return ticket), with a line from label 5 pointing to the 'Обратный билет' button. At the bottom of the main area is a large button labeled 'Подобрать' (Select), with a line from label 6 pointing to it.

Рисунок 23 – Страница поиска рейсов

Страница входа в личный кабинет представлена на рисунке 24. На ней имеются следующие элементы: 1 – поле ввода логина или почты пользователя, 2 – поле ввода пароля пользователя, 3 – кнопка «Войти». При нажатии кнопки «Войти» система проверяет, есть ли в базе данных данный пользователь, потом проверяется введенные данные. Если введенные данные не верны или такой пользователь не существует в системе, то выдается уведомление: «Логин или пароль не верны» или «Такого пользователя не существует. Иначе будет произведен вход в систему. Если пользователь нажмет на «Зарегистрироваться?», то он перейдет на страницу регистрации, представленной на рисунке 25.

The diagram shows a rectangular frame representing a web page. At the top center is the text "Вход". Below it are two input fields: the first is labeled "Почта/логин" and the second is labeled "Пароль". Below the password field is the text "Зарегистрироваться?". At the bottom center is a button labeled "Войти". Three horizontal lines with numbers point to the right from the frame: line 1 points to the "Почта/логин" field, line 2 points to the "Пароль" field, and line 3 points to the "Войти" button.

Рисунок 24 – Страница входа в личный кабинет

Страница регистрации представлена на рисунке 25. На ней имеются следующие элементы: 1 – поле ввода логина или почты пользователя, 2 – поле для ввода пароля пользователя, 3 – кнопка «Зарегистрироваться». При нажатии кнопки «Зарегистрироваться» система проверяет, являются ли введенные данные допустимыми. Если это условие не выполняется, будет выведено сообщение о недопустимом логине или пароле.

The diagram shows a rectangular frame representing a web page. At the top center is the text "Регистрация". Below it are two input fields: the first is labeled "Почта/логин" and the second is labeled "Пароль". Below the password field is the text "Назад". At the bottom center is a button labeled "Зарегистрироваться". Three horizontal lines with numbers point to the right from the frame: line 1 points to the "Почта/логин" field, line 2 points to the "Пароль" field, and line 3 points to the "Зарегистрироваться" button.

Рисунок 25 – Страница регистрации

Страница личного кабинета пользователя представлена на рисунке 26. На ней присутствуют следующие элементы: 1 – поле изменения почты, 2 – поле изменения пароля, 3 – поле изменения ФИО, 4 – поле изменения паспорта, 5 – поле изменения номера телефона, 6 – кнопка «Изменить». При нажатии на кнопку «Изменить» система проверяет введенные данные на корректность. Если они не корректны, то появится сообщение о недопустимых символах. Иначе изменения вступят в силу.

Личный кабинет

Почта	1
Пароль	2
ФИО	3
Паспорт	4
Номер телефона	5
Изменить	6

Рисунок 26 – Страница личного кабинета пользователя

Страница результатов поисков рейсов представлена на рисунке 27. На ней присутствуют следующие элементы: 1 – выпадающий список «Фильтры», 2 – результаты поиска рейсов по критерию пользователя. При нажатии на «Фильтры» система показывает выпадающий список где пользователь может применить различную фильтрацию к рейсам (не раньше, точное время и т.д.). Когда пользователь выбрал рейс, то он переходит на страницу выбора места в самолете, представленную на рисунке 28.

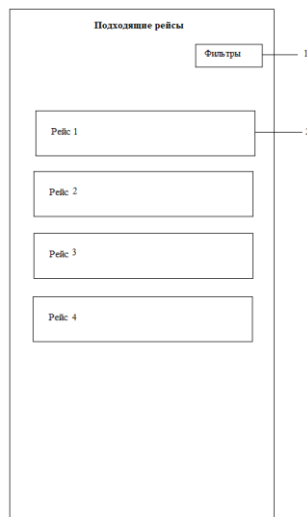


Рисунок 27 – Страница результатов поиска рейсов

Страница выбора места в самолете представлена на рисунке 28. На ней присутствуют следующие элементы: 1 – кнопки «Места», 2 – кнопка «Продолжить». При выборе места в самолете, будут показаны сведения о данном посадочном месте. Так же пользователь может выбрать несколько мест в самолете. Если пользователь не выбрал ни одно место и нажал на продолжить, то система предоставит уведомление «Пожалуйста выберите место», иначе пользователь перейдет на страницу бронирования билета, представленной на рисунке 29.

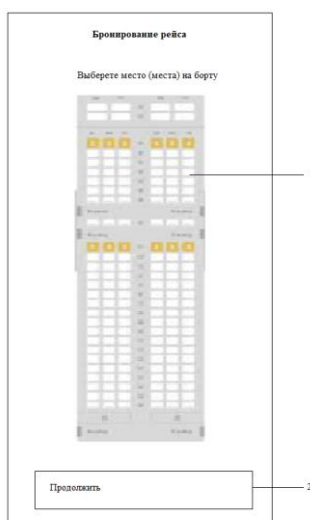


Рисунок 28 – Страница выбора места в самолете

Страница бронирования рейса представлена на рисунке 29. На ней присутствуют следующие элементы: 1 – информация по билету и формы

заполнения информации, 2 – кнопка «Забронировать». Если пользователь выбрал несколько посадочных мест, то система ему предоставит точно такое же количество билетов, к тому же, если пользователь был ранее зарегистрирован и ввел данные в личном кабинете, то данные для заполнения билета уже будут введены, так же пользователь может изменить эти личные данные по билету. Забронировать билет на перелет будет невозможно, пока все данные не будут проверены на корректность.

Бронирование рейса

Билет 1

Номер рейса

Откуда Куда

Дата и время вылета Дата и время прилета

Билет 2

Номер рейса

Откуда Куда

Дата и время вылета Дата и время прилета

Рисунок 29 – Страница бронирования билета

Страница информации о рейсе представлена на рисунке 30. На ней присутствуют следующие элементы: 1 – форма для ввода номера билета, 2 – кнопка «Найти». Пользователю предоставляется возможность ввести свой номер билета и получить по нему полную информацию. Если введенного билета не существует, то система выдаст уведомление: «Такого билета не существует». Иначе система покажет всю информацию по билету, представленную на рисунке 31.

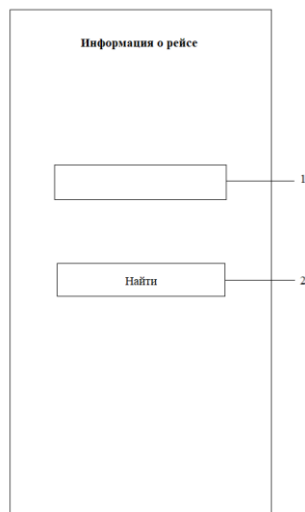


Рисунок 30 – Страница поиска информации о рейсе

Страница результатов поиска информации о рейсе представлена на рисунке 31. На ней присутствуют следующие элементы: 1 – окно для просмотра информации по перелету.

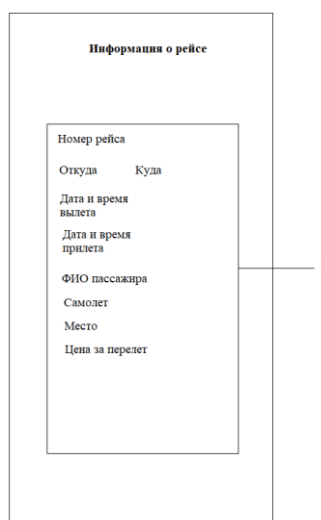


Рисунок 31 – Страница результатов поиска информации о рейсе

Все окна пользовательского интерфейса созданы с направленностью на использование в мобильном устройстве пользователя и обладают всеми характерными чертами.

2.4 Концептуальное проектирование базы данных

Концептуальная модель представляет собой описание основных сущностей (таблиц) и связей между ними без учета принятой модели БД и

синтаксиса целевой СУБД. Часто на такой модели отображаются только имена сущностей (таблиц) без указания их атрибутов. Основная задача концептуальной модели — передать фундаментальные принципы и основные функциональные возможности системы, которую она представляет. Кроме того, концептуальная модель должна быть разработана таким образом, чтобы обеспечить легко понятную системную интерпретацию для пользователей модели.

Графическое представление концептуальной модели базы данных в виде диаграммы «сущность-связь» для предметной области информационной системы представлено на рисунке 32.

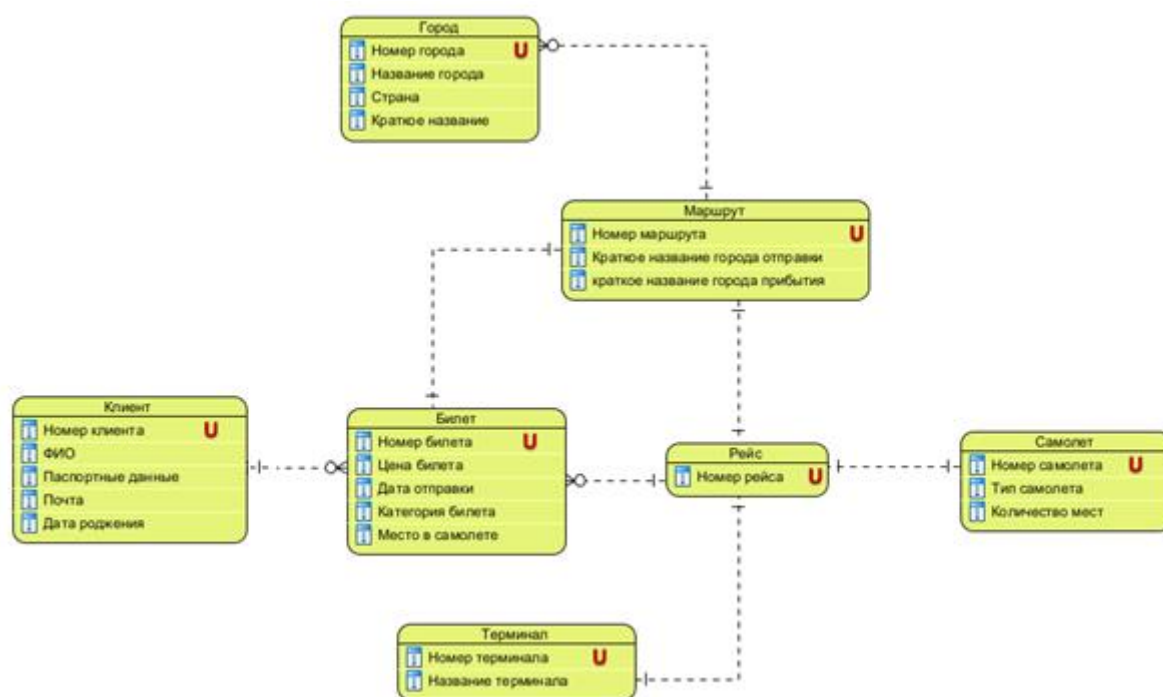


Рисунок 32 – Графическое представление концептуальной модели.

Описание сущностей концептуальной модели представлено в таблице 8.

Таблица 8 – Сущности предметной области

№	Название	Описание	Количество экземпляров	Режим обновления
1	Билет	Описание билета (номер, цена, дата и т.д.)	100000	Добавляется в систему при бронировании билета клиентом

Продолжение таблицы 8

№	Название	Описание	Количество экземпляров	Режим обновления
2	Клиент	Список пользователей системы	10000	В систему добавляются новые пользователи при регистрации
3	Маршрут	Список маршрутов	100	Обновляется при добавлении нового маршрута в систему
4	Город	Список городов	100	Обновляется при добавлении новых городов в систему
5	Терминал	Номер терминала	10	Обновляется раз в год
6	Самолет	Список самолетов	50	Обновляется при добавлении рейса в систему
7	Рейс	Список рейсов	1000	Обновляется при добавлении нового рейса в систему

Описание атрибутов сущностей представлено в таблице 9.

Таблица 9 – Описание атрибутов сущностей

№	Сущность	Атрибут	Описание
1	Билет	Номер билета	Уникальный номер билета
		Цена билета	Цена за билет
		Дата отправки	Дата и время начала перелета
		Категория билета	Категория билета (Эконом, Комфорт, Бизнес)
		Место в самолете	Номер места в самолете
2	Клиент	Номер клиента	Уникальный номер клиента
		ФИО	Фамилия, имя, отчество клиента
		Паспортные данные	Серия и номер паспорта
		Почта	Почта клиента
		Дата рождения	Дата рождения клиента
3	Маршрут	Номер маршрута	Уникальный номер маршрута
		Краткое название города отправки	Сокращенное название города отправки
		Краткое название города прибытия	Сокращенное название города прибытия

Продолжение таблицы 9

№	Сущность	Атрибут	Описание
4	Город	Номер города	Уникальный номер города
		Название города	Название города
		Страна	Страна где расположен город
		Краткое название	Сокращенное название города
5	Терминал	Номер терминала	Уникальный номер терминала
		Название терминала	Название терминала отправки
6	Самолет	Номер рейса	Уникальный номер самолета
		Тип самолета	Тип самолета
		Количество мест	Количество мест в самолете
7	Рейс	Номер рейса	Уникальный номер рейса

Отписание связей представлено в таблице 10.

Таблица 10 – Описание связей

№	Отношение	Описание
1	Клиент - Билет	Определяет билеты клиентам
2	Билет – Маршрут	Определяет маршрут билета
3	Маршрут - Город	Определяет города маршрута
4	Маршрут - Рейс	Определяет маршрут рейса
5	Самолет - Рейс	Определяет самолет рейса
6	Рейс - Билет	Определяет билеты рейса
7	Рейс - Терминал	Определяет терминал с которого будет отправлен рейс

Концептуальная модель построена в среде Visual Paradigm в нотации ERD [27].

2.5 Логическое проектирование базы данных

Логическая модель – это модель базы данных, выраженная в понятиях модели данных. Логическая модель расширяет концептуальную путем определения для сущностей их атрибутов, описаний и ограничений, уточняет состав сущностей и взаимосвязи между ними.

На основе анализа предметной области, функциональных требований и концептуальной модели была построена логическая модель базы данных представленная на рисунке 33.

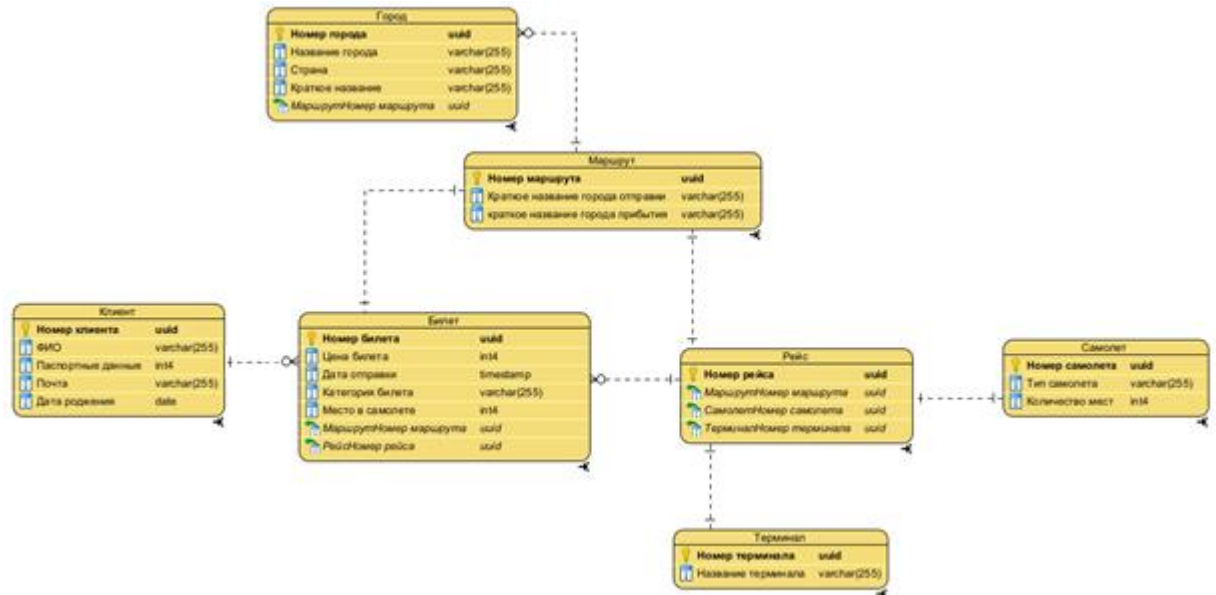


Рисунок 33 – Графическое представление логической модели базы данных

В таблице 11 представлено описание связей между сущностями предметной области.

Таблица 11 – Описание связей между сущностями предметной области

№	Тип сущности 1	Тип сущности 2	Описание	Мощность	Обязательность	Ограничения
1	Клиент	Билет	Каждый пользователь может соответствовать нескольким билетам	1:M	Необязательная (пользователь может не иметь билетов)	
2	Билет	Маршрут	Каждый билет соответствует своему маршруту	1:1	Обязательная	
3	Маршрут	Город	Каждый маршрут соответствует нескольким городам	1:M	Необязательная (город может не иметь маршрутов)	Город не обязательно привязан к какому либо маршруту
4	Маршрут	Рейс	Каждый маршрут соответствует своему рейсу	1:1	Обязательная	
5	Самолет	Рейс	Каждый самолет соответствует своему рейсу	1:1	Обязательная	
6	Рейс	Билет	Каждый рейс соответствует нескольким билетам	1:M	Обязательная	

Продолжение таблицы 11

№	Тип сущности 1	Тип сущности 2	Описание	Мощ- ность	Обязатель- ность	Ограни- чения
7	Рейс	Терминал	Каждый рейс соответствует своему номеру терминала	1:1	Обязательная	

Логическая модель построена в среде Visual Paradigm в нотации ERD.

На представленной логической модели спроектированная структура БД отображена уже более подробно. Здесь не только присутствуют типы сущностей и связи между ними, но и хранится информация о конкретных атрибутах сущностей. Однако на данный момент сущности все еще представлены в абстрактной форме, без привязки к конкретной реализации базы данных.

2.6 Проектирование серверной части приложения

Серверная часть приложения представляет собой облачное хранилище Firebase Realtime Database. Оно облегчает хранение данных на основе JSON и выполняет синхронизацию данных в реальном времени с подключенными клиентами. Приложение подключается к базе данных через WebSocket, который отвечает за синхронизацию данных в течение всего сеанса. Система аутентификации пользователей интегрируется Firebase непосредственно в базу данных. Запросы в базе данных реального времени ограничены одной операцией за раз [28].

3 Реализация мобильного приложения

3.1 Реализация базы данных

3.2 Разработка серверной части

На основе составленной ранее логической модели БД при помощи программы Visual Paradigm была построена физическая модель. Ее структура представлена на рисунке 34.

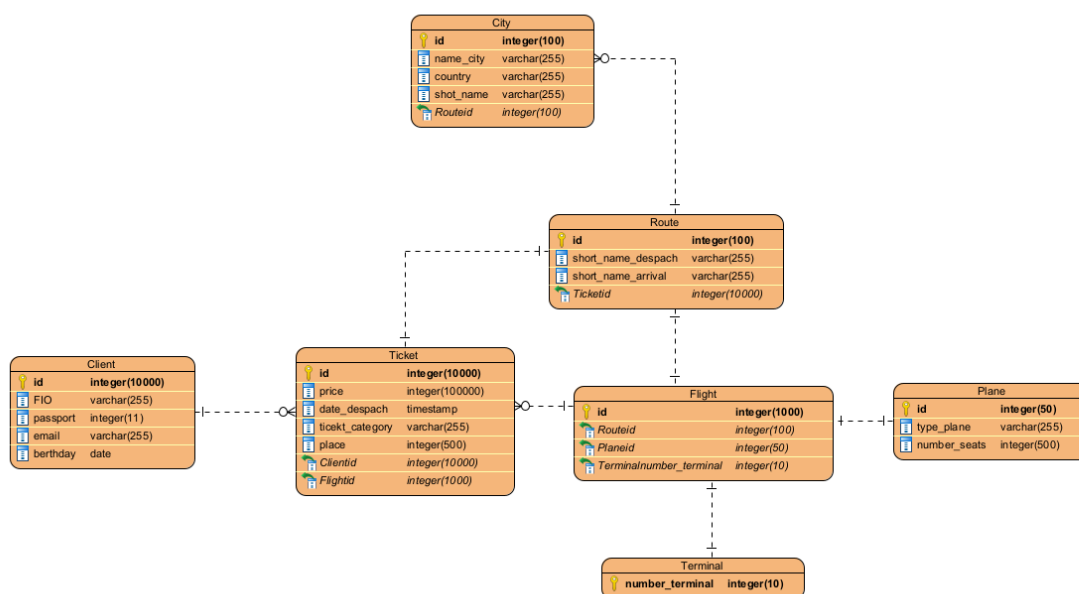


Рисунок 34 – Физическая модель базы данных

Развертывание спроектированной базы данных было выполнено хостингом Firebase.

3.3 Реализация клиентской части приложения

Разработка мобильного приложения производилась на языке Java с использованием среды программирования Android Studio. Диаграмма компонентов, отражающая структуру программы, представлена на рисунке 35.

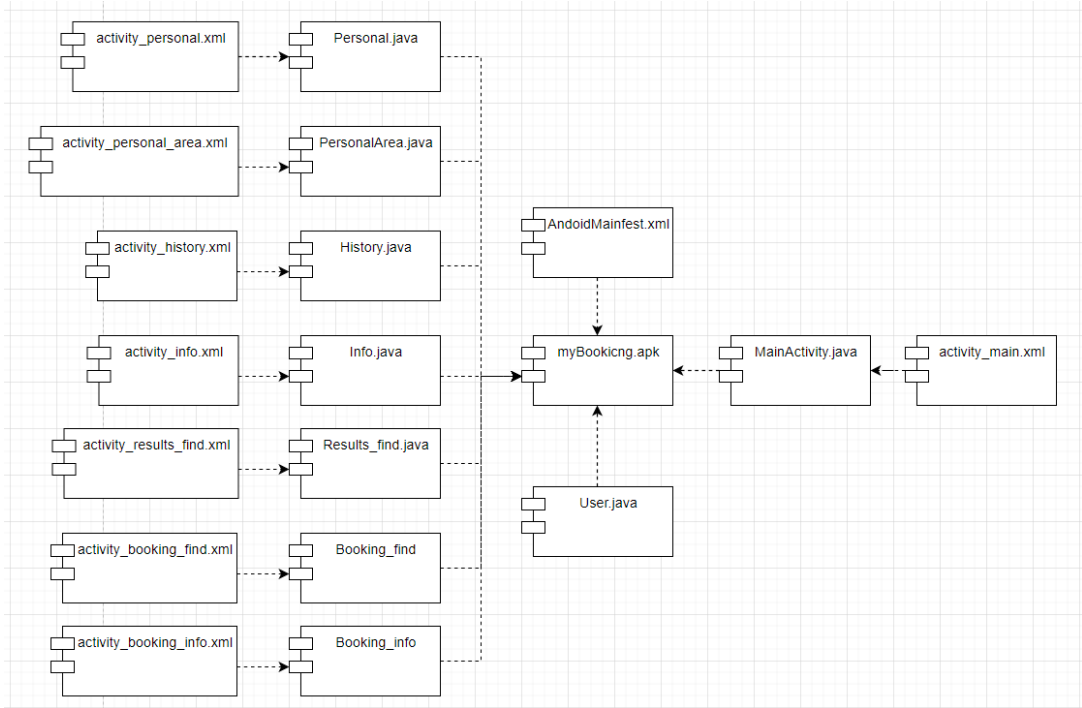


Рисунок 35 – Диаграмма компонентов мобильного приложения

Описание компонентов представлено в таблице 12.

Таблица 12 – Компоненты клиентского приложения

Компонент	Назначение
MainActivity.java	Файл класса начальной страницы приложения (страница входа в систему).
activity_main.xml	XML-файл, реализующий отображение страницы входа в систему.
Personal.java	Файл класса страницы регистрации и авторизации.
activity_personal.xml	XML-файл, реализующий отображение страницы регистрации и авторизации.
MainActivity.java	Файл класса главной страницы приложения (объединяет в себе страницу поиска, страницу списка личного кабинета).
activity_main.xml	XML-файл, реализующий отображение главной страницы приложения.
PersonalArea.java	Файл класса страницы информации о пользователе.
activity_personal_area.xml	XML-файл, реализующий отображение страницы информации о пользователе.
History.java	Файл класса страницы истории полетов.
activity_history.xml	XML-файл, реализующий отображение страницы истории полетов.

Продолжение таблицы 12

Компонент	Назначение
Info.java	Файл класса страницы информации о рейсе.
activity_info.xml	XML-файл, реализующий отображение страницы информации о рейсе.
Results_find.java	Файл класса страницы с результатами поиска рейсов.
activity_results_find.xml	XML-файл, реализующий отображение страницы с результатами поиска рейсов
Booking_find.java	Файл класса страницы с информацией по бронированию билета.
activity_booking_find.xml	XML-файл, реализующий отображение страницы с информацией по бронированию билета.
Booking_info.java	Файл класса страницы с результатами бронирования.
activity_booking_info.xml	XML-файл, реализующий отображение страницы с результатами бронирования.
User.java	Файл класса страницы пользователей.
AndroidManifest.xml	XML-файл с параметрами проекта.
myBooking.apk	Исполняемый файл приложения.

Диаграмма развертывания, на которой описывается физическое расположение компонентов системы, представлена на рисунке 36.

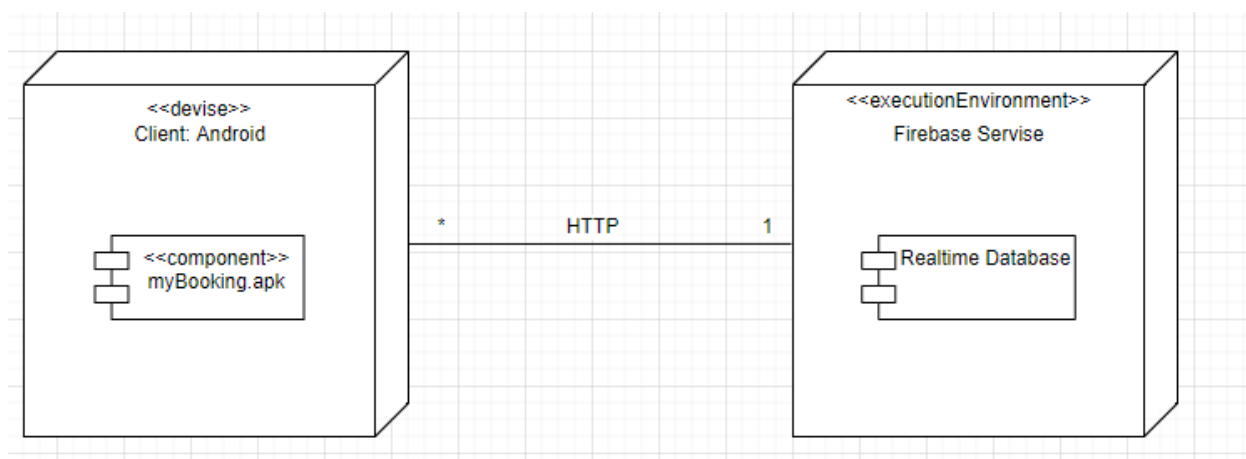


Рисунок 36 – Диаграмма развертывания мобильного приложения

Как можно увидеть на этой диаграмме, приложение работает на мобильном устройстве под управлением системы Android. API и база данных располагаются на сервере, работающем под управлением Firebase. Листинг клиентской части программы представлен в приложении Г.

4 Тестирование программного продукта

4.1 Тестирование серверной части приложения

Так как была выбрана бессерверная архитектура проекта, то тестирование серверной части можно проверить только на добавление информации в нашу облачную базу данных. Результаты тестирования серверной части приложения представлены в таблице 13. Скриншоты тестов находятся в приложении Д.

Таблица 13 – Результаты тестирования серверной части

№	Описание теста	Параметры	Результат
1	Добавление нового пользователя	email = myemail@mail.ru, password= qwer1234	Тест пройден. Пользователь успешно добавлен в систему. (рисунок Д.1).
2	Добавление нового билета	Id = 21512 FIO = Петровович Петр Петрович Email = myemail@mail.ru price = 26000 type = бизнес	Тест пройден. Билет успешно добавлен в систему. (рисунок Д.2).

Все тесты были пройдены успешно, что свидетельствует о корректности работы серверной части приложения и позволяет перейти к тестированию клиентской части.

4.2 Тестирование клиентской части приложения

4.2.1 Функциональное тестирование

Функциональное тестирование – тестирование программного обеспечения в целях проверки его на соответствие функциональным требованиям.

Для проверки корректности работы клиентской части приложения было проведено функциональное тестирование. Его результаты представлены в таблице 14. Скриншоты тестов находятся в приложении Е.

Таблица 14 – Функциональное тестирование

Вариант использования	Тест и параметры	Результат
Зарегистрироваться	Ввести валидные данные для регистрации. Почта: myemail@mail.ru. Пароль: qwer1234.	Тест пройден. Пользователь зарегистрирован в системе (рисунок Е.1).
	Ввести данные с почтой существующего в системе пользователя. Почта: myemail@mail.ru.	Тест пройден. Выведено уведомление о том, что почта уже занята (рисунок Е.2).
	Попытаться задать почту и пароль с недопустимыми символами. Почта: логин. Пароль: пароль.	Тест пройден. Выведено уведомление о вводе недопустимых данных (рисунок Е.3).
	Попытаться задать почту и пароль недопустимой длины. Почта: m. Пароль: q.	Тест пройден. Выведено уведомление о вводе недопустимых данных (рисунок Е.4).
	Произвести попытку задать пароль без букв. Почта: email@mail.ru. Пароль: qwerty	Тест пройден. Выведено уведомление о вводе недопустимых данных (рисунок Е.5).
Авторизоваться	Ввести валидные данные. Почта: myemail@mail.ru. Пароль: qwer1234.	Тест пройден. Авторизация выполнена успешно (рисунок Е.6).
	Ввести неверные данные. Почта: myemail@mail.ru. Пароль: qwert12145.	Тест пройден. Выведено уведомление о неверном логине или пароле (рисунок Е.7).
Искать рейсы	Ввести поисковые данные: Откуда: Москва Куда: Анапа Дата отправки: 26.06.2022 Нажать на кнопку подобрать.	Тест пройден. Найдены рейсы по поисковому запросу (рисунок Е.8, Е.9.).

Функциональное тестирование показало корректность работы клиентской части мобильного приложения.

4.3 Метрики кода

Метриками кода называются некоторые количественные величины, позволяющие подсчитать, насколько код программы обладает тем или иным

свойством. Существует огромное количество самых разнообразных метрик, имеющих различное назначение. Наиболее известными являются следующие группы метрик:

- метрики размера программы (SLOC-метрики);
- метрики сложности потока управления программ (метрики Холстеда и цикломатическая сложность).

Рассмотрим метод `Avtorization` класса `Personal`, отвечающий за обработку запроса на вход в систему существующего пользователя нового пользователя на сервере. Его код представлен в листинге 1.

Листинг 1 – Вход в систему

```
// Метод входа в аккаунт
public void on_click_avtorization(View view) {
    if (!TextUtils.isEmpty(input_email.getText().toString())
        && !TextUtils.isEmpty(input_password.getText().toString())) { //
        Проверка на пустые поля

        mAuth.signInWithEmailAndPassword(input_email.getText().toString(
        ),
        input_password.getText().toString()).addOnCompleteListener(this,
        new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult>
            task) {
                if (task.isSuccessful()) { // Проверка входа
                в аккаунт
                    Toast.makeText(getApplicationContext(),
                    "Успешный вход", Toast.LENGTH_SHORT).show();
                }
                else {
                    Toast.makeText(getApplicationContext(),
                    "Неуспешный вход", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Данный метод запускается при попытке произвести вход пользователя в систему. Он проверяет существование пользователя с заданными логином и паролем. В случае нахождения такого пользователя подтверждает вход пользователя в систему. При неверном логине или пароле отправляется соответствующее сообщение об ошибке.

Для данного метода будут вычислены две группы метрик: метрики размера программы и метрики Холстеда.

4.3.1 Метрики размера программы

Данные метрики представляют собой подсчет количества строк программного кода, а также числа строк с комментариями. Вычислим метрики размера программы для приведенного выше программного кода. Результаты представлены в таблице 15.

Таблица 15 – Метрики размера программы

Метрика	Количество
Количество строк кода	15
Строки-комментарии	3

Также вычислим процент комментариев. Он равен $(3/15)*100=20$. Это свидетельствует о достаточном количестве комментариев в коде для его хорошего понимания при отсутствии избыточных комментариев, переполняющих программу.

4.3.2 Метрики Холстеда

Для вычисления метрик Холстеда необходимо измерить 4 характеристики программы [29 – 31]:

- число уникальных операторов;
- число уникальных операндов;
- общее число операторов;
- общее число операндов.

С целью вычисления этих характеристик были составлен словарь операторов (таблица 16) и словарь операндов (таблица 17).

Таблица 16 – Словарь операторов

Оператор	Число операторов
if	2
&&	1
;	3
()	23

Продолжение таблицы 16

Оператор	Число операторов
!	2
{}	6
new	1
this	1

Таблица 17 – Словарь операндов

Операнд	Число операндов
input_email	2
input_password	2
task	2
mAuth	1

Таким образом, число уникальных операторов программы $n1 = 9$, число уникальных операндов программы $n2 = 4$, общее число операторов в программе $N1 = 41$, общее число операндов в программе $N2 = 7$.

На основе метрик Холстеда можно вычислить целый ряд характеристик программы. Рассмотрим некоторые из них:

- словарь программы (n):

$$n = n1 + n2;$$

- длина программы (N):

$$N = N1 + N2;$$

- объем программы (V):

$$V = N * \log_2 n;$$

- сложность программы (D):

$$D = (n1/2) * (N2/n2);$$

- оценка интеллектуальных усилий (E):

$$E = D * V.$$

Рассчитаем словарь программы: $n = 9 + 4 = 13$.

Рассчитаем длину программы $N = 41 + 7 = 48$.

Рассчитаем объем программы: $V = 48 * \log_2 13 = 177,6$.

Рассчитаем сложность программы: $D = (9/2) * (7/4) = 7,875$.

Рассчитаем оценку интеллектуальных усилий: $E = 7,875 * 177,6 = 1398,6$.

Полученные характеристики показывают, что программный код имеет достаточно высокий уровень качества и удобства для дальнейшей поддержки.

Заключение

В процессе реализации выпускной квалификационной работы было разработано мобильное приложение для поддержки процессов резервирования авиабилетов.

На этапе анализа требований и предметной области был выполнен анализ программных продуктов аналогичного назначения, в итоге которого были выявлены их основные превосходства и недочеты, определены функциональные и нефункциональные требования к создаваемому приложению и произведен анализ целевой аудитории.

Было осуществлено проектирование приложения и оценка бюджета. Итоги планирования показали, что всеобщая продолжительность проектного плана составляет 92 дней, а полагаемые затраты на реализацию – 462 560 рублей, что свидетельствует об экономической рациональности данного проектного плана.

На этапе проектирования была сформирована логическая модель базы данных и разработана архитектура мобильного приложения. Было принято решение разрабатывать бессерверную архитектуру, база данных расположена на облачном сервисе, инструментом является Firebase. Клиентская часть была разработана на языке Java с применением среды программирования Android Studio.

Было исполнено тестирование серверной и клиентской частей разрабатываемого приложения. Оно показало корректность его работы. Также рассчитаны метрики программного кода для оценки его качества.

В последующем планируется опубликовать разработанное приложение в магазине мобильных приложений для системы Android Google Play.

Таким образом, в процессе реализации выпускной квалификационной работы были успешно выполнены все поставленные задачи.

Список использованных источников

1. Распоряжение Президента Российской Федерации от 9 мая 2017 г. № 203 «Стратегия развития информационного общества в Российской Федерации на 2017 – 2030 годы». [Электронный ресурс]. URL: http://zmedu.hostedu.ru/files/ykaz_7668.pdf (дата обращения: 12.04.2022).
2. Программа «Цифровая экономика Российской Федерации». Утверждена распоряжением Правительства Российской Федерации от 28 июля 2017 г. № 1632-р. [Электронный ресурс]. URL: <http://static.government.ru/media/files/9gFM4FHj4PsB79I5v7yLVuPgu4bvR7M0.pdf> (дата обращения: 12.04.2022).
3. Аэрофлот. О компании Аэрофлот [Электронный ресурс]. URL: <https://ir.aeroflot.ru/ru/company-overview/about-aeroflot> (дата обращения: 25.04.2022).
4. Годовой отчет Аэрофлота [Электронный ресурс]. URL: https://ir.aeroflot.ru/fileadmin/user_upload/files/rus/common_info/gosa_doc_2021/13-3.pdf (дата обращения: 24.04.2022).
5. Utair [Электронный ресурс]. URL: <https://aerotur.aero/airline/ut> (дата обращения: 24.04.2022).
6. Авиакомпания Utair [Электронный ресурс]. URL: <https://avianity.ru/aviakompanii/utair> (дата обращения: 24.04.2022).
7. Aviasales [Электронный ресурс]. URL: <https://ru.wikipedia.org/wiki/Aviasales> (дата обращения: 24.04.2022).
8. Перевозки пассажиров в России [Электронный ресурс]. URL: <https://www.aviastat.ru/statistics/12-perevozki-passazhiro-v-rossii-itogi-2019-goda> (дата обращения: 24.04.2022).
9. Росстат [Электронный ресурс]. – URL: <https://rosstat.gov.ru/> (дата обращения: 26.04.2022).
10. Мировой рынок смартфонов [Электронный ресурс]. URL: <https://www.tadviser.ru/index.php> (дата обращения: 26.04.2022).

11. Мировая статистика распространения разных версий Android [Электронный ресурс]. – URL: <https://3dnews.ru/1054313/desyatka-ostayotsya-samoy-populyarnoy-versiey-os-android> (дата обращения: 25.04.2022).
12. Блох, Д. Java. Эффективное программирование / Effective Java, 2018 – 464 с.
13. Документация Firebase [Электронный ресурс]. URL: <https://firebase.google.com/docs> (дата обращения: 25.04.2022).
14. Документация Firebase Realtime Database [Электронный ресурс]. URL: <https://firebase.google.com/docs/database> (дата обращения: 25.04.2022).
15. Ехлаков, Ю.П. Управление программными проектами. Стандарты, модели: Учебное пособие / Ю.П. Ехлаков. – СПб.: Лань, 2019. – 244 с.
16. Иванов, П.В. Управление проектами: Учебное пособие / П.В. Иванов. – Рн/Д: Феникс, 2018. – 102 с.
17. Гонтарева, И.В. Управление проектами / И.В. Гонтарева, Р.М. Нижегородцев, Д.А. Новиков. – М.: КД Либроком, 2018. – 384 с.
18. Ньютон, Р. Управление проектами от А до Я / Р. Ньютон. – М.: Альпина Паблишер, 2018. – 180 с.
19. Вольфсон, Б.Л. Гибкое управление проектами и продуктами / Б.Л. Вольфсон. - СПб.: Питер, 2019. – 208 с.
20. Скороход, С.В. Управление проектами средствами Microsoft Project. Курс лекций. / С.В. Скороход. – М.: Интуит НОУ, 2016. – 319 стр.
21. Куперштейн, В.И. Microsoft Project 2013 в управлении проектами / В.И. Куперштейн. – СПб:БХВ-Петербург, 2013. – 432 с.
22. Джонсон, Т., Четфилд К. Microsoft Project 2016. Шаг за шагом / Т.Джонсон, К.Четфилд. – М., Изд-во "КТК Галактика", 2018. – 566 с.
23. Акатова, Н.А. Планирование и контроль выполнения проектов с использованием MS Project / Н.А. Акатова, М.Г. Коротких. – М.: МИСиС, 2019. – 74 с.
24. Унифицированный процесс Rational (RUP) // Интернет-технологии.ру [Электронный ресурс]. URL: [https://www.internet-](https://www.internet-технологии.ру)

technologies.ru/articles/newbie/unificirovannyy-process-rational-rup.html (дата обращения: 25.05.2022).

25. Арлоу, Д. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование, 2-е издание / Д. Арлоу, А. Нейштадт – СПб: Символ Плюс, 2007. – 624 с.

26. Web API // Metanit.com [Электронный ресурс]. URL: <https://metanit.com/sharp/mvc/12.1.php> (дата обращения: 25.05.2022).

27. Документация case-средства Visual Paradigm [Электронный ресурс]. URL: <https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/> (дата обращения 23.05.2022).

28. Firebase Realtime Database Management API [Электронный ресурс]. URL: <https://firebase.google.com/docs/reference/database/database-management/rest?ur> (дата обращения 28.05.2022).

29. Программный код и его метрики // Habr.com [Электронный ресурс]. URL: <https://habr.com/ru/company/intel/blog/106082/> (дата обращения: 25.05.2022).

30. Аверьянов, А.В. Применение метрик Холстеда для количественного оценивания характеристик программ ЭВМ / А.В. Аверьянов, И.Н. Кошель, В.В. Кузнецов // Известия высших учебных заведений. Приборостроение. – 2019. – №11. – С. 970 – 975.

31. Полякова, Э.Р. Исследование программного обеспечения встраиваемой вычислительной системы метриками Холстеда / Э.Р. Полякова, А.А. Поляков, А.Л.Шустов, Д.Г. Придачкин // Cloud of Science. – 2020. – №4. – С. 950 – 959.

Приложение А
ГЛОССАРИЙ
(справочное)

Android – мобильная операционная система для смартфонов, планшетов и других мобильных устройств, разрабатываемая и выпускаемая американской компанией Google и используемая на подавляющем большинстве современных мобильных устройств.

RUP (Rational Unified Process) – методология разработки программного обеспечения, в основе которой лежит итеративная модель разработки.

Firebase – облачная платформа для разработки, управления и запуска приложений.

Realtime Database – хранение и синхронизация данных в реальном времени .

Приложение Г
ИСХОДНЫЙ КОД МОБИЛЬНОГО ПРИЛОЖЕНИЯ
(рекомендуемое)

Листинг Г.1 – Файл MainActivity.java

```

package com.example.diplom;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private Button personalArea, findButton;
    private EditText whereOut, whereIn, date_out, date_in;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        init();

        personalArea.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new
Intent("android.intent.action.Personal");
                startActivity(intent);
            }
        });

    }

    private void init() {
        personalArea = findViewById(R.id.personalArea);
        findButton = findViewById(R.id.findButton);
        whereOut = findViewById(R.id.whereOut);
        whereIn = findViewById(R.id.whereIn);
        date_out = findViewById(R.id.date_out);
        date_in = findViewById(R.id.date_in);
    }

    public void find_rice(View view) {
        if (!TextUtils.isEmpty(whereOut.getText().toString()) &&
!TextUtils.isEmpty(whereIn.getText().toString()) &&
!TextUtils.isEmpty(date_out.getText().toString())) {
            Intent intent = new
Intent("android.intent.action.ResultFind");
            intent.putExtra("whereOut",
whereOut.getText().toString());

```

```

        intent.putExtra("whereIn",
whereIn.getText().toString());
        intent.putExtra("date_out",
date_out.getText().toString());
        intent.putExtra("date_in",
date_in.getText().toString());
        startActivity(intent);
    }
    else {
        Toast.makeText(getApplicationContext(), "Ошибка",
Toast.LENGTH_SHORT).show();
    }
}

    public void censellTicket(View view) {
        Intent intent = new
Intent("android.intent.action.Cansel");
        startActivity(intent);
    }

    public void findTicket(View view) {

    }
}

```

Листинг Г.2 – Файл Cansel.java

```

package com.example.diplom;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

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;

public class Cansel extends AppCompatActivity {

    private EditText cansel_edit_text;
    private Button cansel_button;
    private DatabaseReference dataRef;
    private String TIKETS_KEY = "tickets";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cansel);
        init();
    }
}

```

```

    }

    private void init() {
        cancel_edit_text = findViewById(R.id.cancel_edit_text);
        cancel_button = findViewById(R.id.cancel_button);
        dataRef =
        FirebaseDatabase.getInstance().getReference("tickets");
    }

    public void cancelOnClick(View view) {
        ValueEventListener valueEventListener = new
        ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot
            snapshot) {
                for (DataSnapshot ds : snapshot.getChildren()) {
                    Ticket ticket = ds.getValue(Ticket.class);
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError
            error) {
            }
        };
    }
}

```

Листинг Г.3 – Файл Flight.java

```

package com.example.diplom;

public class Flight {
    public String id, dispatch_city, whereIn, date_out, date_in;

    public Flight() {
    }

    public Flight(String id, String dispatch_city, String
    whereIn, String date_out, String date_in) {
        this.id = id;
        this.dispatch_city = dispatch_city;
        this.whereIn = whereIn;
        this.date_out = date_out;
        this.date_in = date_in;
    }
}

```

Листинг Г.4 – Файл Personal.java

```

package com.example.diplom;

```

```

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

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;
import com.google.firebase.auth.FirebaseUser;

public class Personal extends AppCompatActivity {

    private FirebaseAuth mAuth;
    private TextView register_text_view, back_text_view;
    private EditText input_email, input_password;
    private Button next_button, register_button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_personal);
        init();

        register_text_view.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                next_button.setVisibility(View.INVISIBLE);
                register_button.setVisibility(View.VISIBLE);
                back_text_view.setVisibility(View.VISIBLE);

register_text_view.setVisibility(View.INVISIBLE);
            }
        });

        back_text_view.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                next_button.setVisibility(View.VISIBLE);
                register_button.setVisibility(View.INVISIBLE);
                back_text_view.setVisibility(View.INVISIBLE);
                register_text_view.setVisibility(View.VISIBLE);
            }
        });
    }
}

```

```

private void init() {
    register_text_view
findViewById(R.id.register_text_view);
    next_button = findViewById(R.id.next_button);
    register_button = findViewById(R.id.register_button);
    back_text_view = findViewById(R.id.back_text_view);
    input_email = findViewById(R.id.input_email);
    input_password = findViewById(R.id.input_password);
    mAuth = FirebaseAuth.getInstance();
}

public void on_click_registration(View view) {
    if (!TextUtils.isEmpty(input_email.getText().toString())
    && !TextUtils.isEmpty(input_password.getText().toString())) {
        if
(!input_email.getText().toString().equals("email@mail.ru")) {

mAuth.createUserWithEmailAndPassword(input_email.getText().toStr
ing(),
input_password.getText().toString()).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull
Task<AuthResult> task) {
        if (task.isSuccessful()) {

Toast.makeText(getApplicationContext(), "Успешная регистрация",
Toast.LENGTH_SHORT).show();
        Intent intent = new
Intent("android.intent.action.PersonalArea");
        startActivity(intent);
        }
        else {

Toast.makeText(getApplicationContext(), "Неуспешная
регистрация", Toast.LENGTH_SHORT).show();
        }
        }
    });
}

    else {
        Toast.makeText(getApplicationContext(), "Такая
почта уже зарегистрирована", Toast.LENGTH_SHORT).show();
    }
}
    else {
        Toast.makeText(getApplicationContext(), "Введите
данные регистрации", Toast.LENGTH_SHORT).show();
    }
}

public void on_click_avtorization(View view) {

```

```

        if (!TextUtils.isEmpty(input_email.getText().toString())
        && !TextUtils.isEmpty(input_password.getText().toString())) {

mAuth.signInWithEmailAndPassword(input_email.getText().toString(
),
input_password.getText().toString()).addOnCompleteListener(this,
new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult>
task) {
        if (task.isSuccessful()) {
            Toast.makeText(getApplicationContext(),
"Успешный вход", Toast.LENGTH_SHORT).show();
            Intent intent = new
Intent("android.intent.action.PersonalArea");
            startActivity(intent);
        }
        else {
            Toast.makeText(getApplicationContext(),
"Неуспешный вход", Toast.LENGTH_SHORT).show();
        }
    }
});
    }
}
}

```

Листинг Г.5 – Файл ResultFind.java

```

package com.example.diplom;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;

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;
import java.util.List;

public class ResultFind extends AppCompatActivity {

    private String a,b,c,d, FLIGHT_KEY = "flight";
    private DatabaseReference dataRef;
    private List<String> list_rise;
    private List<Flight> list_rise_flight;

```

```

private ListView find_list_rise;
private ArrayAdapter<String> adapter;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_result_find);
    init();
    getDataDB();
    setOnClickItem();
}

private void init(){
    a = getIntent().getStringExtra("whereOut");
    b = getIntent().getStringExtra("whereIn");
    c = getIntent().getStringExtra("date_out");
    d = getIntent().getStringExtra("date_in");

    find_list_rise = findViewById(R.id.list_rise);
    dataRef =
    FirebaseDatabase.getInstance().getReference("flight");
    list_rise = new ArrayList<>();
    list_rise_flight = new ArrayList<>();
    adapter = new
    ArrayAdapter<>(this,
    android.R.layout.simple_list_item_1, list_rise);
    find_list_rise.setAdapter(adapter);
}

private void getDataDB(){
    ValueEventListener valueEventListener = new
    ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot
    snapshot) {
            if (list_rise.size() > 0)list_rise.clear();
            if (list_rise_flight.size() >
    0)list_rise_flight.clear();
            for (DataSnapshot ds : snapshot.getChildren()) {
                Flight flight = ds.getValue(Flight.class);
                assert flight != null;
                list_rise.add(flight.dispatch_city);
                list_rise_flight.add(flight);
            }
            adapter.notifyDataSetChanged();
        }

        @Override
        public void onCancelled(@NonNull DatabaseError
    error) {

        }
    };
    dataRef.addValueEventListener(valueEventListener);
}

```

```

        private void setOnClickItem(){
            find_list_rise.setOnItemClickListener(new
            AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> adapterView,
                View view, int i, long l) {
                    }
            });
        }

        private void check() {
            //proverka
        }
    }

```

Листинг Г.6 – Файл Ticket.java

```

package com.example.diplom;

public class Ticket {
    public String id, FIO, email, price, type;

    public Ticket() {
    }

    public Ticket(String id, String FIO, String email, String
    price, String type) {
        this.id = id;
        this.FIO = FIO;
        this.email = email;
        this.price = price;
        this.type = type;
    }
}

```

Листинг Г.7 – Файл User.java

```

package com.example.diplom;

import com.google.firebase.database.DatabaseReference;

public class User {
    public String id, login, email, password;

    public User() {
    }

    public User(String id, String login, String email, String
    password) {
        this.id = id;
    }
}

```

```

        this.login = login;
        this.email = email;
        this.password = password;
    }
}

```

Листинг Г.8 – Файл Booking_find.java

```

package com.example.diplom;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class booking_find extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_booking_find);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        display(mNumberOfCoffees);
        displayPrice(mNumberOfCoffees * 5);
    }

    public void submitOrder(View view) {
        display(mNumberOfCoffees);
        displayPrice(mNumberOfCoffees * 5);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        mInfoTextView = findViewById(R.id.current_pick);

        RadioButton redRadioButton =
        findViewById(R.id.radio_red);

        redRadioButton.setOnClickListener(new RadioButtonClickListener());

        RadioButton greenRadioButton =
        (RadioButton) findViewById(R.id.radio_green);

        greenRadioButton.setOnClickListener(new RadioButtonClickListener());

        RadioButton blueRadioButton =
        findViewById(R.id.radio_blue);
    }
}

```

```

blueRadioButton.setOnClickListener(radioButtonClickListener);

        RadioButton                grayRadioButton                =
findViewById(R.id.radio_gray);

grayRadioButton.setOnClickListener(radioButtonClickListener);
    }

    View.OnClickListener    radioButtonClickListener    =    new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            RadioButton rb = (RadioButton)v;
            switch (rb.getId()) {
                case                R.id.radio_red:
mInfoTextView.setBackgroundColor(Color.parseColor("#ff0000"));
                break;
                case                R.id.radio_green:
mInfoTextView.setBackgroundColor(Color.parseColor("#0000ff"));
                break;
                case                R.id.radio_blue:
mInfoTextView.setBackgroundColor(Color.parseColor("#00ff00"));
                break;
                case                R.id.radio_gray:
mInfoTextView.setBackgroundColor(Color.parseColor("#666666"));
                break;

                default:
                break;
            }
        }
    };
}

private void display(int number) {
    quantityTextView = findViewById(R.id.countTV);
    quantityTextView.setText("" + number);
}

private void display(int number) {
    TextView                priceTextView                =                (TextView)
findViewById(R.id.priceTV);

priceTextView.setText(NumberFormat.getCurrencyInstance(Locale.US
).format(number));
}

public void decrement(View view) {
    if(mNumberOfCoffees > 0){
        mNumberOfCoffees = mNumberOfCoffees - 1;
        quantityTextView = findViewById(R.id.countTV);
        quantityTextView.setText("" + mNumberOfCoffees);
    }
}

```

```

    }

    public void increment(View view) {
        mNumberOfCoffees = mNumberOfCoffees + 1;
        quantityTextView = findViewById(R.id.countTV);
        quantityTextView.setText("" + mNumberOfCoffees);
    }
}

```

Листинг Г.9 – Файл AndroidManifest.xml

```

<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.diplom">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Diplom">
        <activity
            android:name=".Cansel"
            android:exported="false" >
            <intent-filter>
                <action
android:name="android.intent.action.Cansel" />

                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".info"
            android:exported="false" />
        <activity
            android:name=".booking_find"
            android:exported="false">
            <intent-filter>
                <action
android:name="android.intent.action.booking_find" />

                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".ResultFind"
            android:exported="false">
            <intent-filter>
                <action
android:name="android.intent.action.ResultFind" />

```

```

        <category
android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity
    android:name=".PersonalArea"
    android:exported="false">
    <intent-filter>
        <action
android:name="android.intent.action.PersonalArea" />
    </action>
    <category
android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity
    android:name=".Personal"
    android:exported="false">
    <intent-filter>
        <action
android:name="android.intent.action.Personal" />
    </action>
    <category
android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity
    android:name=".MainActivity"
    android:exported="true">
    <intent-filter>
        <action
android:name="android.intent.action.MAIN" />
    </action>
    <category
android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
</application>

</manifest>

```

Листинг Г.10 – Файл Activity_result_find.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=".ResultFind">

    <TextView
        android:id="@+id/fide_rise_text"
        android:layout_width="wrap_content"

```

```

        android:layout_height="wrap_content"
        android:text="Найденные рейсы"
        android:textSize="30dp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

<ListView
    android:id="@+id/list_rise"
    android:layout_width="409dp"
    android:layout_height="694dp"
    android:layout_marginTop="2dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintTop_toBottomOf="@+id/fide_rise_text" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

Листинг Г.11 – Файл Activity_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    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">

    <Button
        android:id="@+id/findButton"
        android:layout_width="250dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="600dp"
        android:layout_marginBottom="16dp"
        android:onClick="find_rice"
        android:text="@string/selection"
        tools:layout_editor_absoluteX="46dp" />

    <Button
        android:id="@+id/personalArea"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/personal_area_text" />

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/personalArea"
        android:layout_marginTop="200dp"
        android:orientation="horizontal">

```

```

<EditText
    android:id="@+id/whereOut"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_weight="1"
    android:inputType="textPersonName"
    android:hint="@string/city_mandrel" />

<EditText
    android:id="@+id/whereIn"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_weight="1"
    android:inputType="textPersonName"
    android:hint="@string/arrival_city" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/personalArea"
    android:layout_marginTop="250dp"
    android:orientation="horizontal">

    <EditText
        android:id="@+id/date_out"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp"
        android:layout_weight="1"
        android:inputType="date"
        android:hint="@string/departure_date" />

    <EditText
        android:id="@+id/date_in"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp"
        android:layout_weight="1"
        android:inputType="date"
        android:hint="@string/arrival_date" />
</LinearLayout>

<Button
    android:id="@+id/find_info_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_marginStart="10dp"

```

```

        android:layout_marginLeft="10dp"
        android:layout_marginEnd="10dp"
        android:layout_toStartOf="@+id/button3"
        android:layout_toEndOf="@+id/personalArea"
        android:layout_toRightOf="@+id/personalArea"
        android:onClick="findTicket"
        android:text="Найти" />

<Button
    android:id="@+id/button3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_alignParentEnd="true"
    android:layout_marginLeft="330dp"
    android:layout_marginTop="0dp"
    android:layout_marginEnd="8dp"
    android:onClick="censellTicket"
    android:text="ОТМЕНИТЬ" />

</RelativeLayout>

Листинг Г.12 – Файл Activity_personal.xml

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    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=".Personal">

    <TextView
        android:id="@+id/personalAreaText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp"
        android:gravity="center"
        android:text="@string/personal_area_text"
        android:textSize="30dp"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/input_email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/personalAreaText"
        android:layout_marginHorizontal="100dp"
        android:layout_marginTop="150dp"
        android:ems="10"
        android:hint="Почта"
        android:inputType="text" />

    <EditText

```

```

        android:id="@+id/input_password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_below="@+id/input_email"
        android:layout_marginHorizontal="100dp"
        android:layout_marginTop="30dp"
        android:hint="Пароль" />

```

```

<TextView
    android:id="@+id/register_text_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/input_password"
    android:layout_marginTop="20dp"
    android:gravity="center"
    android:text="Зарегистрироваться?"
    android:textSize="15dp" />

```

```

<TextView
    android:id="@+id/back_text_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/input_password"
    android:layout_marginTop="20dp"
    android:gravity="center"
    android:visibility="invisible"
    android:text="Назад"
    android:textSize="15dp" />

```

```

<Button
    android:id="@+id/next_button"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/register_text_view"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="200dp"
    android:gravity="center"
    android:onClick="on_click_avtorization"
    android:visibility="visible"
    android:text="Войти" />

```

```

<Button
    android:id="@+id/register_button"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/register_text_view"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="200dp"
    android:gravity="center"
    android:onClick="on_click_registration"
    android:text="Зарегистрироваться"
    android:visibility="invisible" />

```

```

</RelativeLayout>

```

Приложение Д
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ СЕРВЕРНОЙ ЧАСТИ
(рекомендуемое)

Search by email address, phone number, or user UID

Add user

↺ ⋮

Identifier	Providers	Created ↓	Signed In	User UID
myemail@mail.ru	✉	Jun 14, 2022	Jun 14, 2022	hbyFm6AAtUOkYhHU0QxqbQl3fUy1
qwerty@mail.ru	✉	Jun 14, 2022	Jun 14, 2022	ShcOcoDUgWeQ1hnGRStW9OHba...
vsm_mil@inbox.ru	✉	Jun 14, 2022	Jun 14, 2022	DZDDnQgxZSbD2xEA3Uz3kLrwm...
qwerty123@mqil.ru	✉	Jun 14, 2022	Jun 14, 2022	8FRL5luqznTT0kY00AedPOHBPV...

Рисунок Д.1 – Успешная регистрация в системе

▼ tickets

▼ ticket1

FI0: "Петровович Петр Петрович"

email: "myemail@mail.ru"

id: 21512

price: 26000

type: "бизнесс"

Рисунок Д.2 – Успешное добавление билета

Приложение Е
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ
(рекомендуемое)

11:57

Личный кабинет

myemail@mail.ru

.....

ФИО

Паспорт

Номер телефона

ИЗМЕНИТЬ

Успешная регистрация

Рисунок Е.1 – Успешная регистрация в системе

11:55

Личный кабинет

myemail@mail.ru

qwer1234

Назад

ЗАРЕГИСТРИРОВАТЬСЯ

Такая почта уже зарегистрирована

Рисунок Е.2 – Почта занята

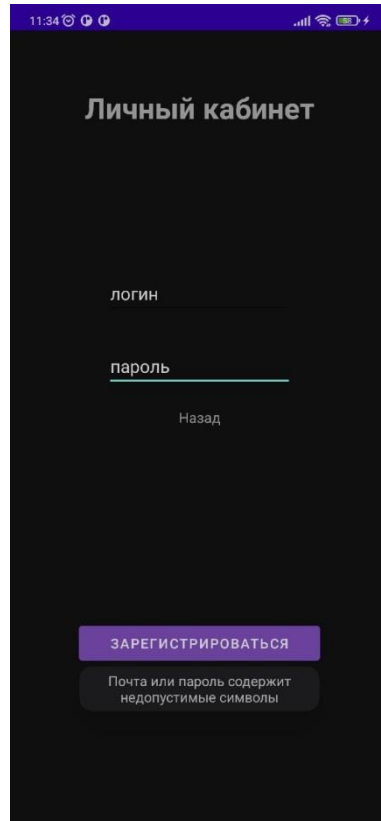


Рисунок Е.3 – Почта и пароль содержат недопустимые символы

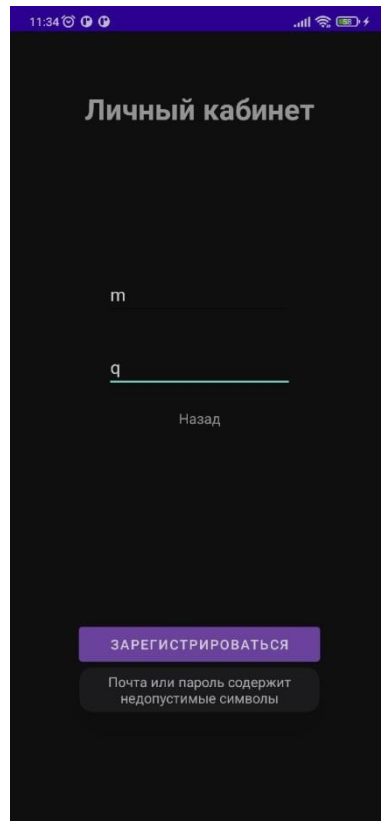


Рисунок Е.4 – Почта и пароль недопустимой длины

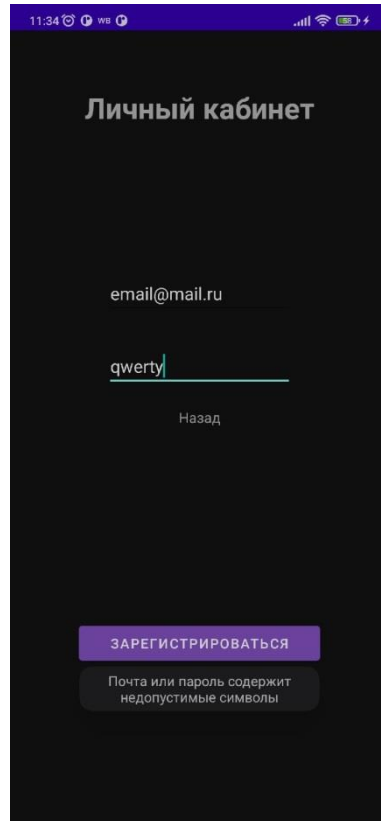


Рисунок Е.5 – Пароль содержит недопустимые символы

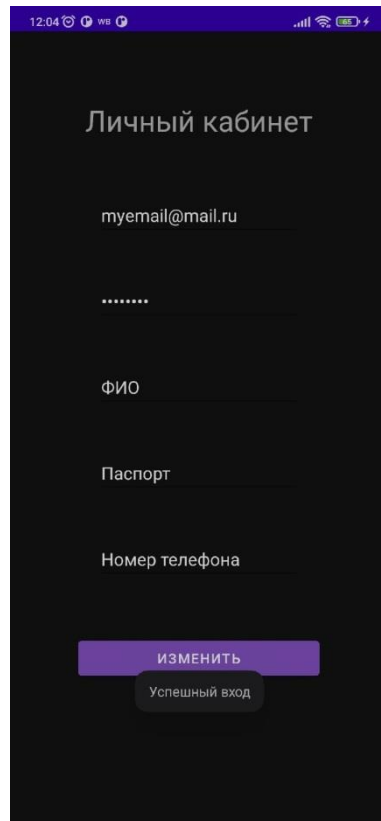


Рисунок Е.6 – Успешный вход в аккаунт

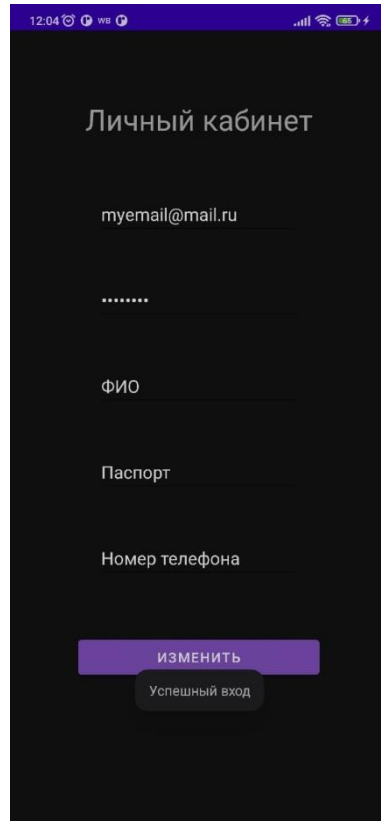


Рисунок Е.6 – Успешный вход в аккаунт

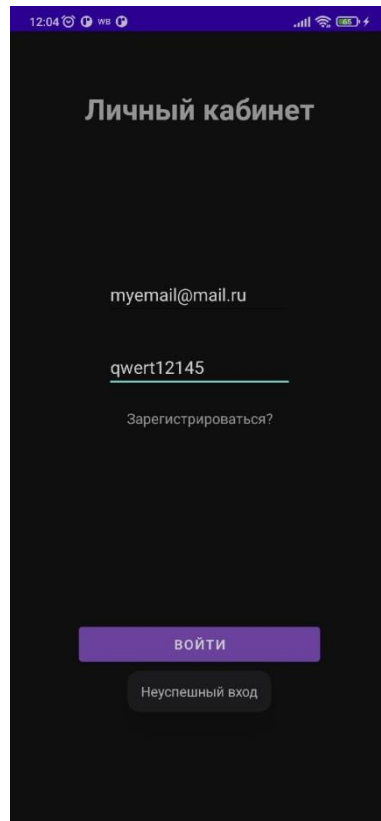


Рисунок Е.7 – Неуспешный вход в аккаунт

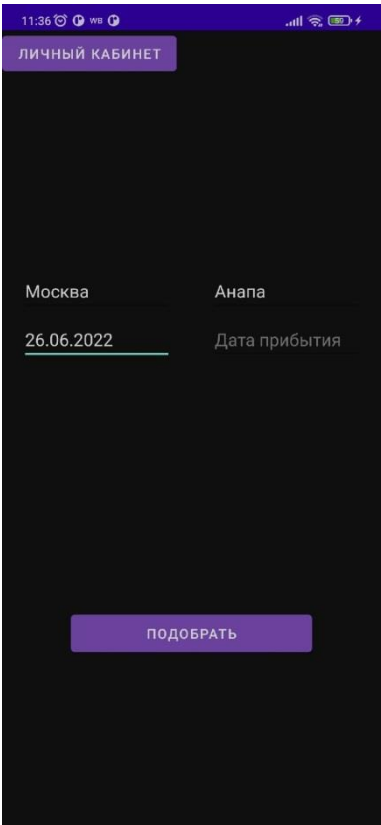


Рисунок Е.8 – Поиск рейсов

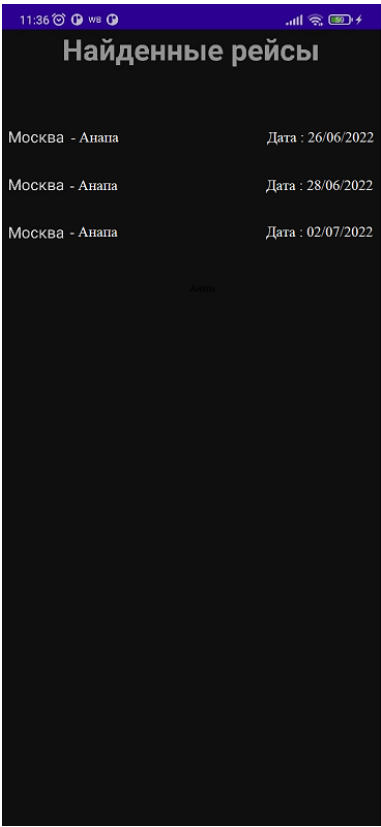


Рисунок Е.9 – Поиск рейсов