МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет вычислительной техники

Кафедра МО и ПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

БАКАЛАВРСКАЯ РАБОТА

на тему

Мобильное приложение для онлайн-шопинга

Студент _____ Учинин Дмитрий Николаевич
(подпись, дата) (ФИО полностью)

Руководитель _____ Балашова И.Ю.
(подпись, дата) (фамилия, инициалы)

Нормоконтролер _____ 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	*Кафедра*
Вычислительной техники	МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____П.П. Макарычев
«__» _____2022 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент Учинин Дмитрий Николаевич гр. 18ВП2 факультета ВТ
<Фамилия, имя, отчество полностью>
2.Тема работы Мобильное приложение для онлайн-шопинга

Тема утверждена приказом ПГУ № 327/О от "18" апреля 2022 г.

3. Руководитель работы к.т.н., доцент кафедры «МО и ПЭВМ» Балашова И.Ю.

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

1) Назначение: мобильное приложение предназначено для проведения онлайн-шопинга в торговых центрах города.

2) Среда выполнения: операционная система Android с минимальной версией Android 5.1.

Языковые средства разработки: язык программирования Java, язык разметки XML.

Инструментальные средства: Android Studio, Firebase.

Технологии разработки ООП, UML.

3) Основные функции мобильного приложения:

а) регистрация и авторизация;

б) предоставление списка торговых центров;

в) предоставление информации о магазинах, актуальных акциях, новостях;

г) добавление и удаление торговых центров в закладках;

д) просмотр истории посещений торговых центров;

е) составление рейтинга торговых центров;

ж) возможность оценки пользователем торговых центров;

з) изменение данных о пользователе.

5. Перечень подлежащих разработке вопросов:

1) *Анализ предметной области и постановка задачи;*
2) *Проектирование;*
3) *Реализация;*
4) *Контроль качества программного обеспечения;*
5) *План разработки бюджета;*
6) *Оформление пояснительной записки.*

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и постановка задачи	12%	13.04.2022 – 18.04.2022	
2	Проектирование	20%	19.04.2022 – 07.05.2022	
3	Реализация	42%	08.05.2022 – 03.06.2022	
4	Контроль качества программного обеспечения	8%	04.06.2022 – 07.06.2022	
5	План разработки бюджета	3%	08.06.2022 – 10.06.2022	
6	Оформление пояснительной записки	15%	10.06.2022 – 17.06.2022	

Дата выдачи задания «22» ____декабря____2021 г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 77 листа, 36 рисунков, 28 таблиц, 14 использованных источников и 3 приложения.

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ ОНЛАЙН-ШОПИНГА, БАЗА ДАННЫХ, JAVA, FIREBASE, КЛИЕНТ-СЕРВЕР.

Объект разработки – мобильное приложение для онлайн-шопинга.

Цель выпускной квалификационной работы – разработать мобильное приложение для онлайн-шопинга, предоставляющее пользователям основную информацию о магазинах торговых центрах, актуальные новости, акции.

Технологии разработки – для разработки серверной составляющей приложения использовался облачный сервис Firebase, для разработки клиентской части был использован язык Java с использованием языка разметки XML.

Среда исполнения – Android Studio, мобильное приложение с операционной системой Android с минимальной версией 5.1.

Результаты работы – в ходе выполнения выпускной квалификационной работы было создано мобильное приложение для онлайн-шопинга. Все требования к разработке приложения соблюдены.

Содержание

Введение 6
1 Анализ предметной области мобильного приложения для онлайн-шопинга 7
1.1 Анализ программных продуктов аналогичного назначения 7
1.1.1 «СНТ Шоппинг» 7
1.1.2 «ТЦ Москвы» 8
1.1.3 «ZOON СПБ» 9
1.2 Сравнение аналогов 9
1.3 Анализ требований 11
1.3.1 Функциональные требования 11
1.3.2 Нефункциональные требования 20
2 Проектирование мобильного приложения мобильного приложения для онлайн-шопинга 21
2.1 Архитектура мобильного приложения 21
2.2 Проектирование структур данных 22
2.2.1 Концептуальная модель базы данных 22
2.2.2 Логическая модель базы данных 23
2.2.3 Физическая модель базы данных 27
2.3 Проектирование программных средств 28
2.3.1 Проектирование классов 28
2.3.2 Проектирование диаграммы деятельности 29
2.3.3 Проектирование диаграммы последовательности алгоритма авторизации мобильного приложения для онлайн-шопинга 31
2.3.4 Диаграмма кооперации приложения 32
2.3.5 Проектирования диаграммы состояний приложения 33
2.4 Проектирование пользовательского интерфейса приложения 34
3 Реализация программного обеспечения мобильного приложения 47
4 Контроль качества программного обеспечения 50
4.1 Функциональное тестирование 50

4.2 Метрики кода 52
4.3 Планирование разработки и оценка бюджета 53
Заключение 58
Список использованных источников 59
Приложение А. Глоссарий 61
Приложение Б. Листинг приложения 63
Приложение В. Результаты тестирования 73

Введение

Рынок электронной коммерции в России – один из главных двигателей торговли. По данным аналитических исследований последних лет объем рынка онлайн-торговли постоянно увеличивается [1]. На это влияет стремительный рост активных пользователей сети и изменение их потребностей.

Одним из трендов торговых центров является развитие цифровых форматов взаимодействия с покупателем, например, привлечение покупателей путем создания приложений для онлайн-шопинга по магазинам, кинотеатрам, развлекательным центрам данного торгового центра. Мобильные приложения онлайн-шопинга являются доступным способом донесения необходимой информации до клиента. Они удобнее в использовании, чем сайты, и обеспечивают постоянное присутствие торгового центра в смартфоне покупателя. Для пользователя основные преимущества использования мобильных приложений онлайн-шопинга заключаются в высвобождение времени на подбор покупки в данном торговом центре, персонификации товарного предложения, расширении возможности покупательского опыта. Таким образом, разработка мобильных приложений онлайн-шопинга является актуальной задачей.

Целью выпускной квалификационной работы является разработка мобильного приложения для онлайн-шопинга. Разработка включает в себя анализ требований, проектирование, реализация и тестирование программного обеспечения.

1 Анализ предметной области мобильного приложения для онлайн-шопинга

1.1 Анализ программных продуктов аналогичного назначения

В настоящее время существует множество ресурсов для онлайн-шопинга. Рассмотрим наиболее популярные интернет-ресурсы такого рода и мобильные приложения, проведем для них сравнительный анализ.

1.1.1 «СНТ Шоппинг»

«СТН Шоппинг» – мобильное приложение, включающее в себя все скидки и акции крупнейших торговых центров Нижнего Новгорода. Интерфейс главной страницы мобильного приложения «СТН Шоппинг» представлен на рисунке 1.

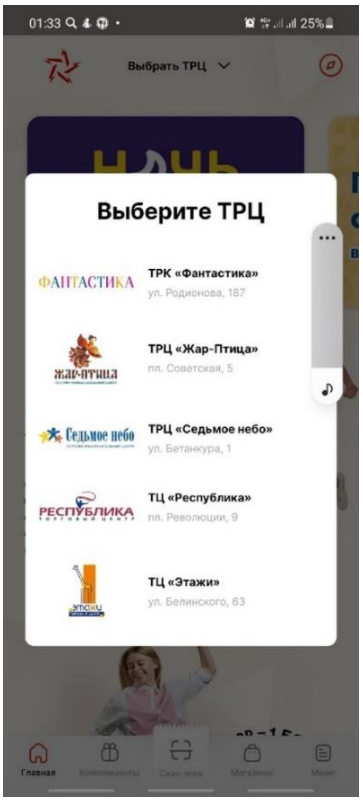


Рисунок 1 – Главная страница мобильного приложения «СТН Шоппинг»

Мобильное приложение «СТН Шоппинг» представляет пользователю выбор торговых центров. В приложении можно узнать информацию о брендах магазинах и акциях, и актуальных новостях. Из плюсов «СТН Шоппинг» можно выделить большое количество выбора торговых центров, понятный интерфейс. Недостатком приложения можно назвать отсутствие авторизации и регистрации пользователя, отсутствие добавления в закладки торговых центров и историю просмотров для конкретного пользователя.

1.1.2 «ТЦ Москвы»

«ТЦ Москвы» – представляет из себя сайт, где собраны крупные торговые центры г. Москвы. Главный экран представлен на рисунке 2.

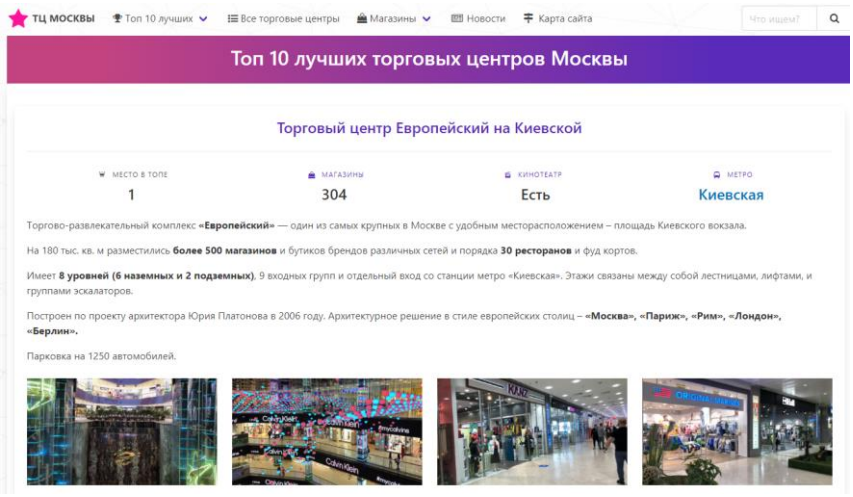


Рисунок 2 – Главный экран «ТЦ Москвы»

Плюсами данного сайта является:

– простой интерфейс, где нет ничего лишнего;
– большой список торговых центров.

К минусам можно отнести:

– отсутствие показа скидок для конкретных торговых центров;
– нет авторизации и регистрации для пользователей;
– отсутствует добавление в избранное;
– отсутствие истории просмотра;
– отсутствие мобильного приложения.

1.1.3 «ZOON СПБ»

«ZOON СПБ» – интернет-ресурс, где собраны торговые центры Санкт-Петербурга. главный экран сайта представлен на рисунке 3.

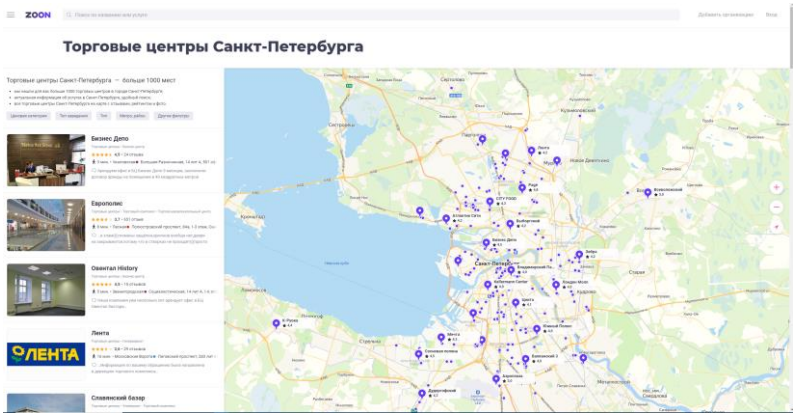


Рисунок 3 – Главный экран «ZOON СПБ»

Плюсами данного сайта является простой интерфейс, большой список торговых центров и большая карта с местоположением. К минусам можно отнести отсутствие показа скидок, новостей для конкретных торговых центров, отсутствует добавление в избранное и истории просмотра, отсутствие мобильного приложения.

1.2 Сравнение аналогов

Рассмотренные программные средства онлайн-шопинга обладают различным функционалом и характеристиками. Можно выделить критерии, напрямую влияющие на удобство применения вышеприведенных решений для осуществления онлайн-шопинга в торговых центрах города:

– регистрация и авторизация пользователей;

– наличие каталога торговых центров города;

– добавление торговых центров в закладки;

– предоставление информации о магазинах и брендах, представленных в торговом центре;

– просмотр новостей и действующих акций в торговом центре;

– сбор отзывов и оценок пользователей о торговом центре;

– формирование рейтинга торговых центров и предоставление его пользователю;

– поиск магазинов и брендов в торговых центрах города.

Для сравнения аналогов используем систему оценок, приведенную в таблице 1.

Таблица 1 – Расшифровка баллов

Балл	Описание
1	Функция реализована в приложении или на сайте
2	Функция частично реализована в приложении или на сайте
3	Функция полностью реализована в приложении или на сайте

Сравнительный анализ рассмотренных выше аналогов представлен в таблице 2.

Таблица 2 – Сравнение аналогов

	Приложения и интернет ресурсы		
Критерии	«СТН Шоппинг»	«ТЦ Москвы»	«ZOON СПБ»
Регистрация и авторизация пользователей	1	1	3
Наличие каталога торговых центров города	3	3	3
Добавление торговых центров в закладки	1	1	1
Предоставление информации о магазинах и брендах, представленных в торговом центре	3	3	3

Продолжение таблицы 2

	Приложения и интернет ресурсы		
Критерии	«СТН Шоппинг»	«ТЦ Москвы»	«ZOON СПБ»
Просмотр новостей и действующих акций в торговом центре	1	2	1
Сбор отзывов и оценок пользователей о торговом центре	1	2	2
Формирование рейтинга торговых центров и предоставление его пользователю	2	1	2
Поиск магазинов и брендов в торговых центрах города	1	3	3

Таким образом, в результате анализа систем аналогичного назначения можно сделать вывод, что приложение «СТН Шоппинг» в сумме набрало 13 баллов, «ТЦ Москвы» 16 баллов, «ZOON СПБ» 18 баллов. Таким образом было принято решение взять за прототип онлайн-сервис «ZOON СПБ». Разрабатываемое приложение будет выполнять все функции аналога, а также выполнять некоторые полезные отсутствующие функции.

1.3 Анализ требований

1.3.1 Функциональные требования

Разрабатываемое мобильное приложение для онлайн-шопинга по торговым центрам должно предоставлять необходимую информацию пользователям. В приложении собрана полная информация о магазинах и брендах, актуальных акциях и свежих новостях. Пользователям приложения не нужно будет лично посещать торговые центры, чтобы искать магазины. Всего лишь нужно иметь мобильный телефон с интернетом или wi-fi подключением. В приложении должен быть интуитивно понятный и простой интерфейс, предоставлена подробная информация о торговых центрах. Также должна быть реализована функция создания аккаунта и авторизации для входа в приложение. Благодаря приложению пользователи смогут оперативно

реагировать на последние события в торговых центрах, в личном кабинете пользователь может сделать закладки о интересующих новостях в выбранном торговом центре. Под каждый торговый центр должна быть выделена отдельная страница с разбивкой на бренды, магазины, развлечения, услуги.

Таким образом, в приложении должны быть реализованы следующие основные функции:

- регистрация и авторизация пользователей;
- каталог торговых центров;
- добавление торговых центров в закладки;
- предоставление информации о магазинах и брендах;
- просмотр новостей и действующих акций;
- сбор отзывов и оценок пользователей;
- поиск магазинов и брендов.

На основе функциональности мобильного приложения для онлайн-шопинга была создана диаграмма вариантов использования, описывающая процесс взаимодействия пользователя с приложением. Для каждого варианта использования было приведено подробное описание. Актером в данном случае является пользователь мобильного приложения для онлайн-шоппинга, всего выделено четырнадцать вариантов использования. Системой выступает само мобильное приложение для онлайн-шопинга.

Также отдельно была создана диаграмма вариантов использования регистрации пользователей, с подробным описанием процесса.

Диаграмма вариантов использования представлена на рисунке 4.

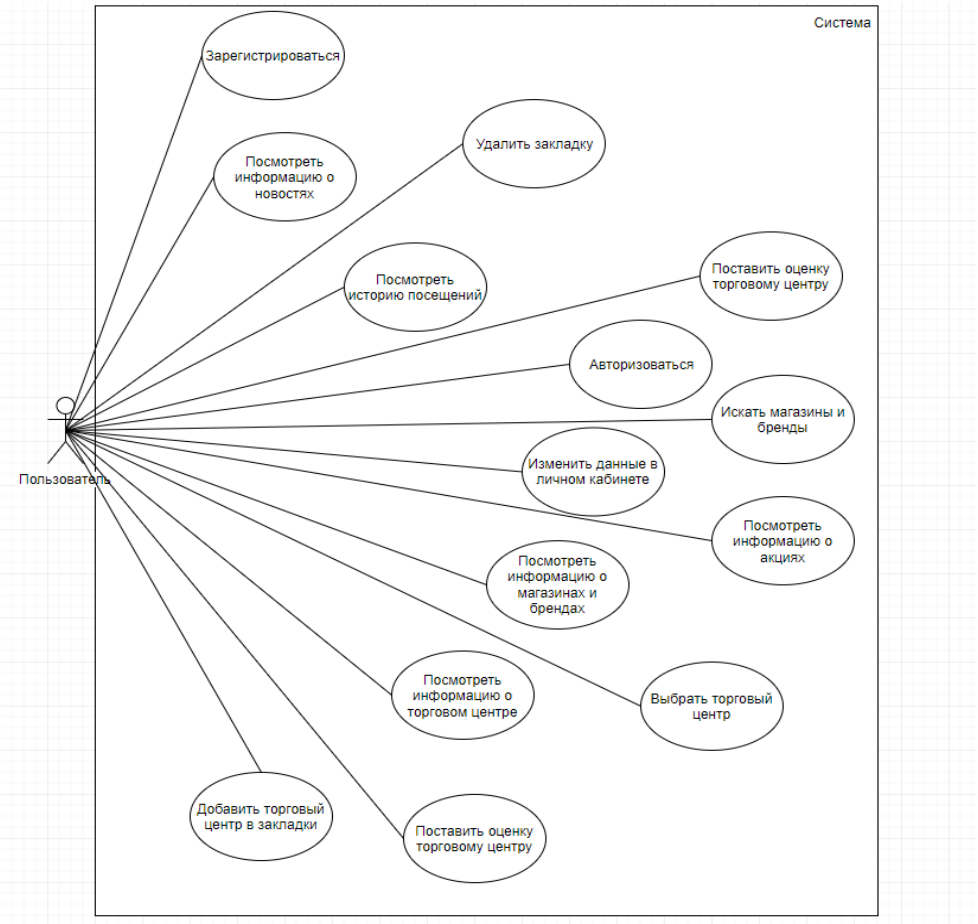


Рисунок 4 – Диаграмма вариантов использования

Отдельно представим диаграмму варианта использования «Зарегистрироваться» на рисунке 5.

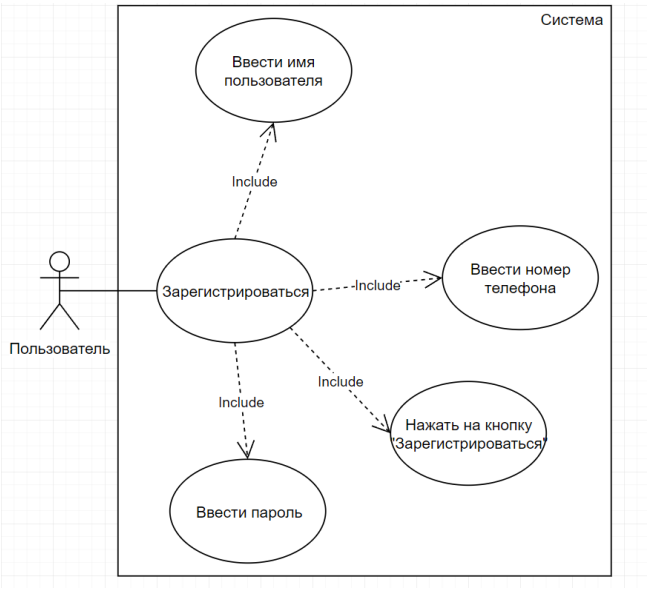


Рисунок 5 – Диаграмма варианта использования «Зарегистрироваться»

Вариант использования «Зарегистрироваться» описывает процесс создания аккаунта пользователя, показан в таблице 3.

Таблица 3 – Вариант использования «Зарегистрироваться»

Зарегистрироваться
Id: 1
Описание: Пользователь регистрируется в системе
Действующие лица: Пользователь
Предусловие: Пользователь открыл приложение
Основной поток: 1. Пользователь попадает на окно регистрации 2. Пользователь вводит имя, номер телефона, пароль 3. Данные о пользователе сохраняются в БД
Постусловие: пользователь зарегистрирован

Вариант использования «Авторизоваться» описывает процесс входа в аккаунт пользователем, показан в таблице 4.

Таблица 4 – Вариант использования «Авторизоваться»

Авторизоваться
Id: 2
Описание: Пользователь авторизуется в системе
Действующие лица: Пользователь
Предусловие: Пользователь зарегистрирован
Основной поток: 1. Пользователь попадает на окно авторизации 2. Пользователь вводит номер телефона и пароль 3. Номер телефона и пароль сравниваются с данными из БД
Постусловие: Пользователь авторизовался

Вариант использования «Посмотреть информацию о новостях» – описывает процесс просмотра пользователем новостей выбранного торгового центра, показан в таблице 5.

Таблица 5 – Вариант использования «Посмотреть информацию о новостях и акциях»

Посмотреть информацию о новостях и акциях
Id: 4
Описание: Пользователь просматривает информацию о новостях и акциях
Действующие лица: Пользователь
Предусловие: Пользователь выбрал торговый центр
Основной поток: 1. Пользователь выбрал вкладу «Новости» 2. Пользователь просматривает информацию и акции в торговом центре
Постусловие: Информация просмотрена

Ниже в таблице 6 приведено описание вариант использования «Посмотреть информацию о магазинах и брендах», показан в таблице 6.

Таблица 6 – Вариант использования «Посмотреть информацию о магазинах и брендах»

Посмотреть информацию о магазинах и брендах
Id: 5
Описание: Пользователь просматривает информацию о магазинах
Действующие лица: Пользователь
Предусловие: Пользователь выбрал торговый центр
Основной поток: 1. Пользователь выбрал вкладу «Магазины» 2. Пользователь просматривает информацию о магазинах и брендах
Постусловие: Информация просмотрена

Вариант использования «Посмотреть информацию о магазинах и брендах» описывает процесс просмотра пользователем магазинов и брендов в выбранном торговом центре.

Вариант использования «Поставить оценку» описывает процесс оценки пользователем торгового центра, показан в таблице 7.

Таблица 7 – Вариант использования «Поставить оценку»

Поставить оценку
Id: 6
Описание: Пользователь ставит оценку торговому центру
Действующие лица: Пользователь
Основной поток: 1. Пользователь ставит оценку торговому центру 2. Данные о оценке сохраняются в БД
Постусловие: Оценка поставлена

Вариант использования «Добавить торговый центр в закладки» описывает процесс добавление пользователем торгового центра в закладки, показан в таблице 8.

Таблица 8 – Вариант использования «Добавить торговый центр в закладки»

Добавить торговый центр в закладки
Id: 7
Описание: Пользователь добавляет торговый центр в закладки
Действующие лица: Пользователь
Основной поток: 1. Пользователь добавляет торговый центр в закладки 2. Закладка добавляется в БД
Постусловие: Оценка поставлена

Вариант использования «Поиск магазинов и брендов» описывает процесс поиска магазинов или брендов в выбранном торговом центре, показан в таблице 9.

Таблица 9 – Вариант использования «Поиск магазинов и брендов»

Поиск магазинов и брендов
Id: 8
Описание: Пользователь выполняет поиск магазинов или брендов соответствующего торгового центра
Действующие лица: Пользователь
Предусловие: Пользователь выбрал торговый центр
Основной поток: 1. Пользователь в строке поиска пишет название магазина или бренда 2. Пользователю выводится магазин или бренд
Постусловие: Пользователь выполнил поиск

Вариант использования «Удалить закладку» описывает процесс удаления пользователем закладки, показан в таблице 10.

Таблица 10 – Вариант использования «Удалить закладку»

Удалить закладку
Id: 9
Описание: Пользователь удаляет торговый центр из закладок
Действующие лица: Пользователь, Firebase
Основной поток: 1. Пользователь заходит в закладки 2. Пользователь удаляет торговый центр из закладок 3. Закладка удаляется из БД
Постусловие: закладка удалена

Вариант использования «Посмотреть историю посещений» описывает процесс просмотра пользователем истории посещений, показан в таблице 11.

Таблица 11 – Вариант использования «Посмотреть историю»

Посмотреть историю посещений
Id: 10
Описание: Пользователь просматривает историю посещений
Действующие лица: Пользователь
Предусловие: История посещений не пуста
Основной поток: 1. Пользователь заходит во вкладку «История» 2. Пользователь просматривает историю посещений торговых центров
Постусловие: Пользователь просмотрел историю

Вариант использования «Посмотреть информацию о торговом центре» описывает процесс просмотра пользователем информации о выбранном торговом центре, показан в таблице 12.

Таблица 12 – Вариант использования «Посмотреть информацию о торговом центре»

Посмотреть информацию о торговом центре
Id: 11
Описание: Пользователь просматривает информацию о торговом центре
Действующие лица: Пользователь
Предусловие: Пользователь выбрал торговый центр
Основной поток: 1. Пользователь заходит во вкладку «Информация» 2. Пользователь просматривает информацию о торговом центре
Постусловие: Пользователь просмотрел информацию о торговом центре

Вариант использования «Выбрать торговый центр» описывает процесс выбора торгового центра пользователем, показан в таблице 13.

Таблица 13 – Вариант использования «Посмотреть историю поиска»

Выбрать торговый центр
Id: 12
Описание: Пользователь выбирает торговый центр
Действующие лица: Пользователь, Firebase
Предусловие:
Основной поток: 3. Пользователь выбрал торговый центр 4. Данные о посещение добавляются в БД
Постусловие: Пользователь выбрал торговый центр

Вариант использования «Изменить данные в личном кабинете» описывает процесс изменения пользователем личных, показан в таблице 14.

Таблица 14 – Вариант использования «Посмотреть историю поиска»

Изменить данные в личном кабинете
Id: 13
Описание: Пользователь изменяет личные данные
Действующие лица: Пользователь
Предусловие: Пользователь зашел в личный кабинет
Основной поток: 1. Пользователь изменяет номер телефона 2. Пользователь изменяет имя 3. Пользователь изменяет пароль 4. Данные о пользователе в БД обновляются
Постусловие: Пользователь изменил данные

В результате была спроектирована и описана диаграмма вариантов использования.

1.3.2 Нефункциональные требования

Разработка приложения должна быть выполнена на языке программирования java, в среде программирования Android Studio. В качестве СУБД должна выступать облачная база данных Firebase. Приложение должно работать на мобильных устройствах с операционной системой не ниже Android Lollipop 5.1.

2 Проектирование мобильного приложения мобильного приложения для онлайн-шопинга

2.1 Архитектура мобильного приложения

Разрабатываемое мобильное приложение для онлайн-шопинга имеет архитектуру типа клиент-сервер. Архитектура мобильного приложения представлена на рисунке 6.

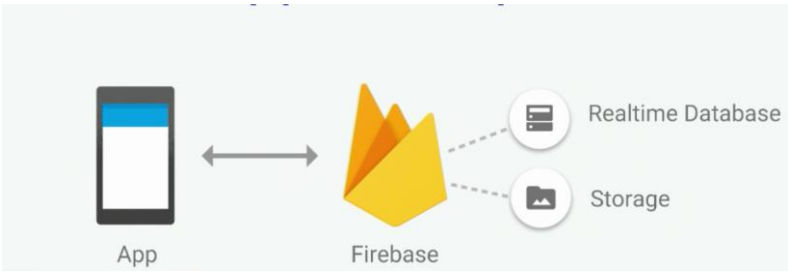


Рисунок 6 –Архитектура мобильного приложения

На рисунке 6 показаны основные компоненты клиент-серверного взаимодействия: Android приложение, сервер Firebase который включает в себя Realtime Database и Storage. Realtime Database – это облачная база данных NoSQL, которая позволяет хранить и синхронизировать данные между пользователями в режиме реального времени [2]. Storage – облачное хранилище предназначено для быстрого и удобного хранения и обслуживания пользовательского контента, такого как фотографии и видео [3].

В данном случае клиентом в архитектуре выступает Android приложение, т.е. пользователю нужно открыть приложение для онлайн-шопинга на своем мобильном устройстве. Процесс взаимодействия клиента с сервером представляет из себя обмен запросами и ответами. В данном случае сервер служит для хранения данных, а вся обработка и представление этих данных происходит на устройстве клиента, такое понятие имеет название «Толстый клиент» [4].

Сервер в свою очередь обрабатывает запросы пользователей такие, как

получение, изменение и проверка данных при авторизации и регистрации. База данных представляет из себя древовидную структуру. Хранилище используется для хранения медиа файлов.

2.2 Проектирование структур данных

2.2.1 Концептуальная модель 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 [5].

Графическое представление концептуальной модели базы данных в виде диаграммы «сущность-связь» для предметной области мобильного приложения представлено на рисунке 7.

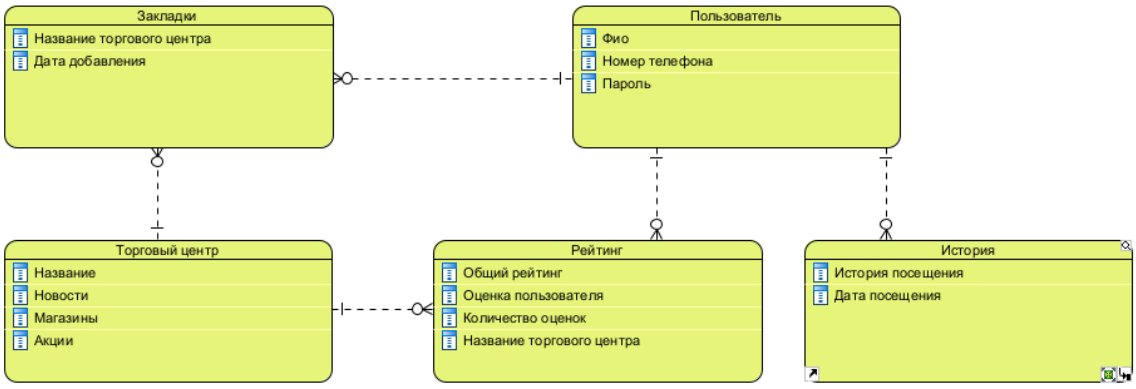


Рисунок 7 – Графическое представление концептуальной модели

Концептуальная модель построена в среде Visual paradigm. После построения концептуальной модели данных, можно перейти к конструированию логической модели данных.

2.2.2 Логическая модель 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 [6].

На основании концептуальной модели была сконструирована логическая модель базы данных на (рисунок 8).

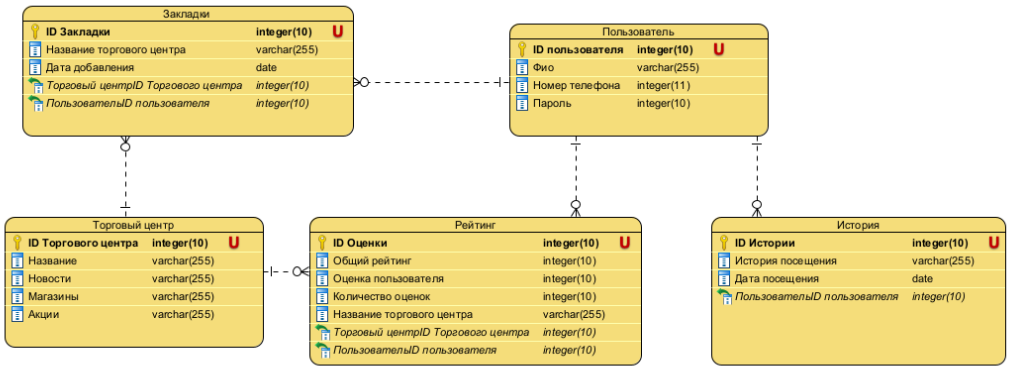


Рисунок 8 – Логическая схема БД

В таблице 15 приведено описание сущностей логической схемы базы данных.

Таблица 15 – Описание сущностей

№	Название	Описание	Количество экземпляров	Режим обновления
1	Пользователь	Информация о пользователе	30	Раз в день
2	Торговый центр	Информация о торговом центре	10	Раз в год и реже
3	Рейтинг	Оценка торговому центру	20	Раз в день
4	История	История поисковых запросов пользователя	55	Раз в день
5	Закладки	Закладки торговых центров	40	Раз в день

В таблице 16 приведено описание типов связи между сущностями логической схемы базы данных.

Таблица 16 – Описание типов связи между сущностями

№	Тип сущности 1	Тип сущности 2	Описание	Мощность	Обязательность
1	Пользователь	Закладки	Определяет закладки пользователя	1:M	Обязательная
2	Закладки	Торговый центр	Определяет торговые центры в закладках пользователя	1:M	Обязательная
3	Рейтинг	Пользователь	Определяет оценку торгового центра	1:M	Обязательная
4	Рейтинг	Пользователь	Определяет принадлежность оценки к пользователю	1:M	Обязательное
5	Пользователь	История	Определяет принадлежность истории поиска к пользователю	1:M	Обязательное

В таблице 17 приведены атрибуты сущности «Пользователь» логической схемы базы данных.

Таблица 17 – Атрибуты сущности «Пользователь»

Наименование	Описание	Тип данных	Размерность	ОДЗ	Nullable	Роль	Пример	Примечание
id	Уникальный номер пользователя	INT	10	0-100000	-	PK	1	
ФИО	ФИО пользователя	VARCHAR	255	А-яA-z	-		Иванов Иван Иванович	
Номер телефона	Номер телефона пользователя	INT	11	0-9-	-		89042644788	
Пароль	Пароль пользователя	INT	10	А-яA-z0-9-	-		Add8934	

В таблице 18 приведены атрибуты сущности «Закладки» логической схемы базы данных.

Таблица 18 – Атрибуты сущности «Закладки»

Наименование	Описание	Тип данных	Размерность	ОДЗ	Nullable	Роль	Пример	Примечание
id	Уникальный номер закладки	INT	10	1-100	-	PK	1	
Название торгового центра	Название торгового центра	VARCHAR	255	А-яA-z	-		Суворовский	
Дата добавления	Дата добавления торгового центра	DATE		0-9-	-		21.06.2022	

В таблице 19 приведены атрибуты сущности «Рейтинг» логической схемы базы данных.

Таблица 19 – Атрибуты сущности «Рейтинг»

Наименование	Описание	Тип данных	Размерность	ОДЗ	Nullable	Роль	Пример	Примечание
id	Уникальный номер администратора	INT	11	1-100000	-	PK	1	
Оценка пользователя	Оценка торговому центру	INT	10	1-10	-		5	
Количество оценок	количество оценок	INT	10	1-100000	-		25	
Общий рейтинг	Общее рейтинг оценок	INT	10	1-100000	-		400	
Название торгового центра	Название торгового центра	VARCHAR	255	А-яA-z	-		Суворовский	

В таблице 20 приведены атрибуты сущности «История» логической схемы базы данных.

Таблица 20 – Атрибуты сущности «История»

Наименование	Описание	Тип данных	Размерность	ОДЗ	Nullable	Роль	Пример	Примечание
id	Уникальный номер администратора	INT	10	1-100000	-	PK	1	
История посещения	Название Истории	VARCHAR	255	А-яA-z	-		Коллаж	
Дата посещения	Дата посещения	DATE		0-9-	-		21.06.2022	

В таблице 21 приведены атрибуты сущности «Торговый центр» логической схемы базы данных.

Таблица 21 – Атрибуты сущности «Торговый центр»

Наименование	Описание	Тип данных	Размерность	ОДЗ	Nullable	Роль	Пример	Примечание
id	Уникальный номер матча	INT	6	1-100000	-	PK	1	
Название	Название торгового центра	VARCHAR	255	А-яA-z	-		Суворовский	
Магазины	Название магазина	VARCHAR	255	А-яA-z	-		Pullbear	
Новости	Информация о новостях	VARCHAR	255	А-яA-z	-		Открытие магазина Pullbear	
Акции	Информация о акциях	VARCHAR	255	А-яA-z	-		Купи вещь вторую получи в подарок	

В результате были описаны атрибуты сущности и связи между ними. От данного уровня модели БД нужно перейти к физической модели.

2.2.3 Физическая модель базы данных

Физическая модель – логическая модель базы данных, выраженная в терминах языка описания конкретной СУБД. Физические модели баз данных определяют способы размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне. В качестве СУБД будет использован PostgreSQL версии 12.

Плюсами данной СУБД является то, что она свободно распространяемая и имеет реализацию для множества UNIX-подобных платформ. Также данная СУБД имеет поддержку множества различных типов данных, такие как JSON, XML, массивы, uuid идентификаторы и т.д. [7].

На основании логической модели была разработана физическая модель базы данных. Физическая модель БД представлена на рисунке 9.

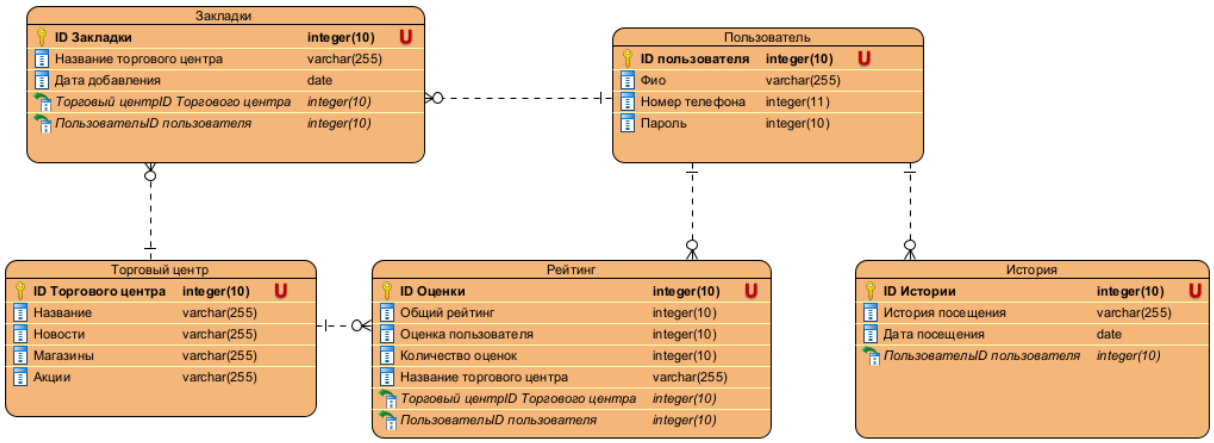


Рисунок 9 – Физическая схема БД

В результате проектирования была разработана концептуальная модель данных. По концептуальной модели данных была разработана логическая и в последствии физическая модель данных. На этапе проектирования логической модели были описаны сущности и связи между ними. Все модели были построены с помощь программы Visual Paradigm [8].

2.3 Проектирование программных средств

2.3.1 Проектирование классов

Для мобильного приложения для онлайн-шопинга была спроектирована диаграмма классов, представленная на рисунке 10.

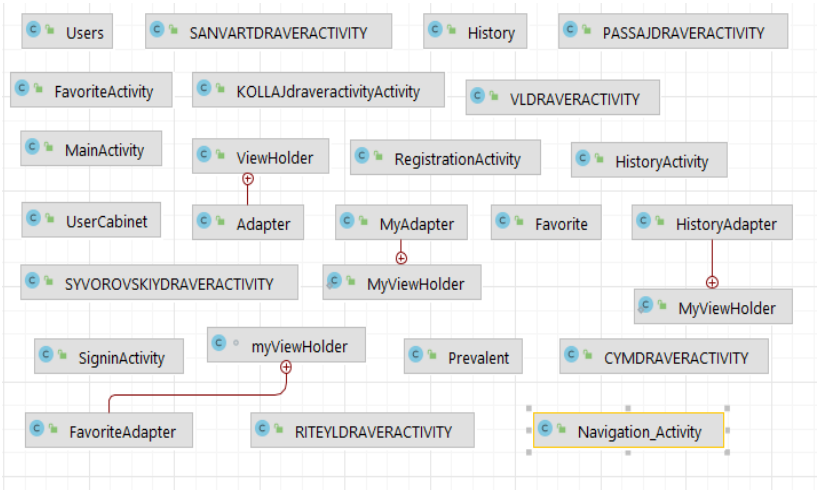


Рисунок 10 – Диаграмма классов

Описание классов мобильного приложения описано в таблице 22.

Таблица 22 – Спецификация класса мобильного приложения

№	Класс	Описание
1	MainActivity	Точка входа в программу. На активности находятся 2 кнопки, первая для авторизации если пользователь раннее был зарегистрирован, и кнопка регистрации если пользователь не зарегистрирован в приложении
2	RegistrationActivity	Активность регистрации пользователей. На активности есть 3 поля ввода, а именно поля для имени, номера телефона, пароля, а также кнопка регистрации по нажатию которой, данные о пользователе добавляются в базу данных
3	SigninActivity	Активность авторизации. На активности 2 поля ввода, а именно номера телефона и пароля, а также кнопка входа в приложение которая проверяет введенные данные и данные в БД
4	Navigation_Activity	NavigationDraver Активность. На активности находится RecycleView список с торговыми центрами, по клику на который происходит переход на активность торгового центра. Так же свайпом вправо открывается меню, в котором пользователь может перейти в закладки, историю, личный кабинет
5	Adapter	Класс, предназначенный для заполнения RecycleView в активности Navigation_Activity

Продолжение таблицы 22

№	Класс	Описание
6	FavoriteActivity	Активность Закладки. Содержит список с закладками пользователя, каждый элемент списка содержит название и кнопку для удаления из списка
7	MyAdapter	Класс, предназначенный для заполнения RecycleView в активности FavoriteActivity
8	Favorite	Класс, предназначенный для получения данных о закладке, содержит метод get для получения данных
9	HistoryActivity	Активность История. Содержит список, каждый элемент списка содержит название и дату посещения торгового центра
10	HistoryAdapter	Класс, предназначенный для заполнения RecycleView в активности HistoryActivity
11	History	Класс, предназначенный для получения данных о истории, содержит метод get для получения данных
12	UserCabinet	Активность Личный кабинет. Содержит 3 поля ввода для изменения данных пользователя и кнопку сохранения или отмены
13	KollajDraverActivity	Активность Торгового центра. Содержит информацию, магазины, акции, новости торгового центра, а также рейтинг торгового центра, raitingbar для оценки торгового центра и кнопку добавления в закладки. Остальные торговые центра имеют такую же структуру

При разработке приложения были реализованы классы, изображенные на рисунке 10. Разрабатываемые классы можно разделить на классы ответственные за описание визуальной части и классы, отвечающие за обработку данных.

2.3.2 Проектирование диаграммы деятельности

Диаграмма деятельности представляет, по существу, блок-схему, которая показывает, как поток управления переходит от одной деятельности к другой. Диаграммы деятельности, как правило, применяются, чтобы промоделировать последовательные и при необходимости параллельные шаги вычислительного процесса. С помощью диаграмм деятельности можно также моделировать жизнь объекта, когда он переходит из одного состояния в другое в разных точках потока управления [9].

Одной из функцией мобильного приложения является авторизация пользователи в системе. Для перехода на страницу информации пользователь

должен авторизоваться или зарегистрироваться в системе, для авторизации пользователь должен ввести номер телефона и пароль, если пользователь не был раннее зарегистрирован, то пользователю нужно будет пройти регистрацию для этого нужно ввести имя пользователя и пароль. Если аккаунт с введенным номером телефона уже раннее был зарегистрирован в системе, то выведется соответствующее сообщение.

Диаграмма деятельности авторизации пользователя представлена на рисунке 11.

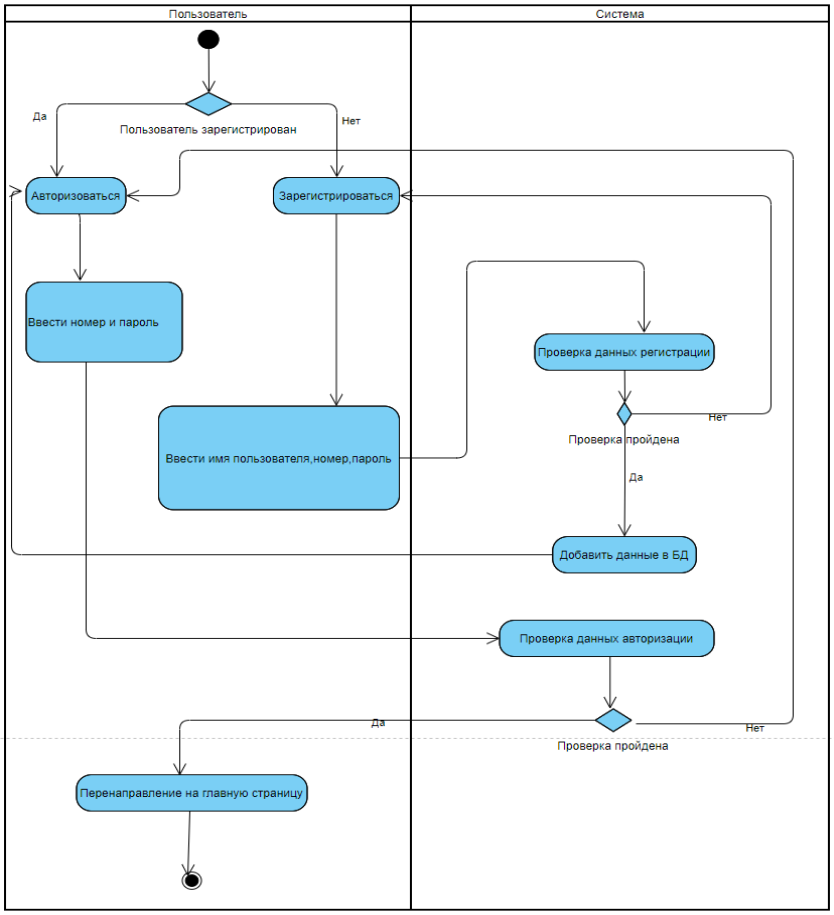


Рисунок 11 – Диаграмма деятельности «Авторизация пользователя»

После успешной авторизации пользователя, происходит перенаправление на главную страницу, где будет доступен полный функционал данного мобильного приложения.

2.3.3 Проектирование диаграммы последовательности алгоритма авторизации мобильного приложения для онлайн-шопинга

Диаграмма последовательности описывает коммуникацию между основными объектами системы. На данной диаграмме описан процесс авторизации и регистрации пользователей (рисунок 12).

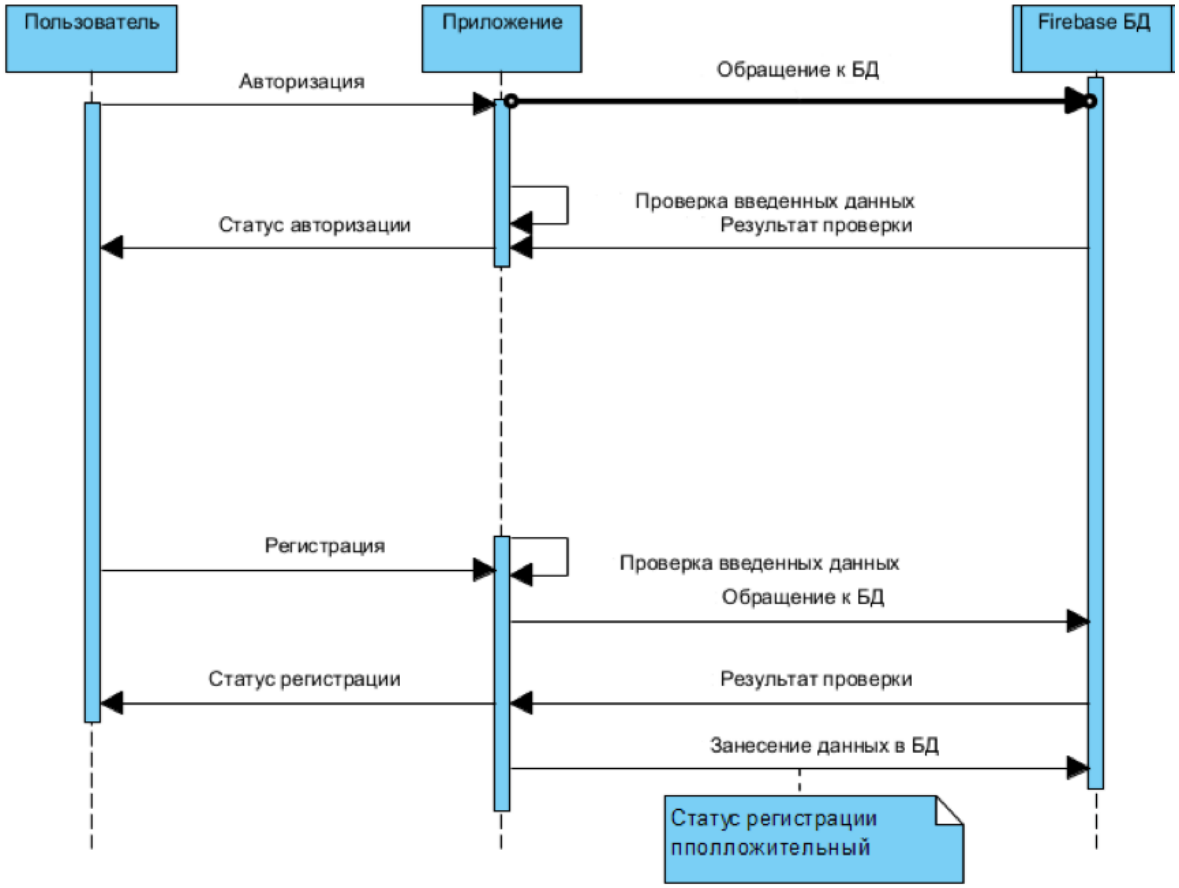


Рисунок 12 – Диаграмма последовательности

Диаграмма последовательности содержит три объекта, а именно пользователя, приложение и база данных Firebase. После того, как пользователь открыл мобильное приложение, он попадает в окно, где ему предстоит авторизоваться в приложении или же зарегистрироваться.

Если пользователь выбрал авторизацию в приложении, то происходит перенаправление на страницу авторизации с двумя полями ввода. После ввода данных пользователь должен нажать на кнопку войти. После данные проверяются на корректность ввода, а затем данные проверяются с записью в

базе данных. Если данные совпадают с записью в базе данных, то происходит перенаправление на главную страницу приложения, иначе происходит перенаправление на страницу регистрации.

На странице регистрации пользователю нужно будет ввести имя пользователя, номер телефона и пароль. После данные проверяются на корректность ввода, если данные введены корректно, то введенные данные сравниваются с записью в базе данных. Если данные отсутствуют в базе данных, то регистрация проходит успешно, иначе происходит перенаправление на страницу авторизации.

2.3.4 Диаграмма кооперации приложения

Диаграмма кооперации уровня спецификации показывает роли, которые играют участвующие во взаимодействии элементы (рисунок 13). Элементами кооперации на этом уровне являются классы и ассоциации, которые обозначают отдельные роли классификаторов и ассоциации между участниками кооперации [10].

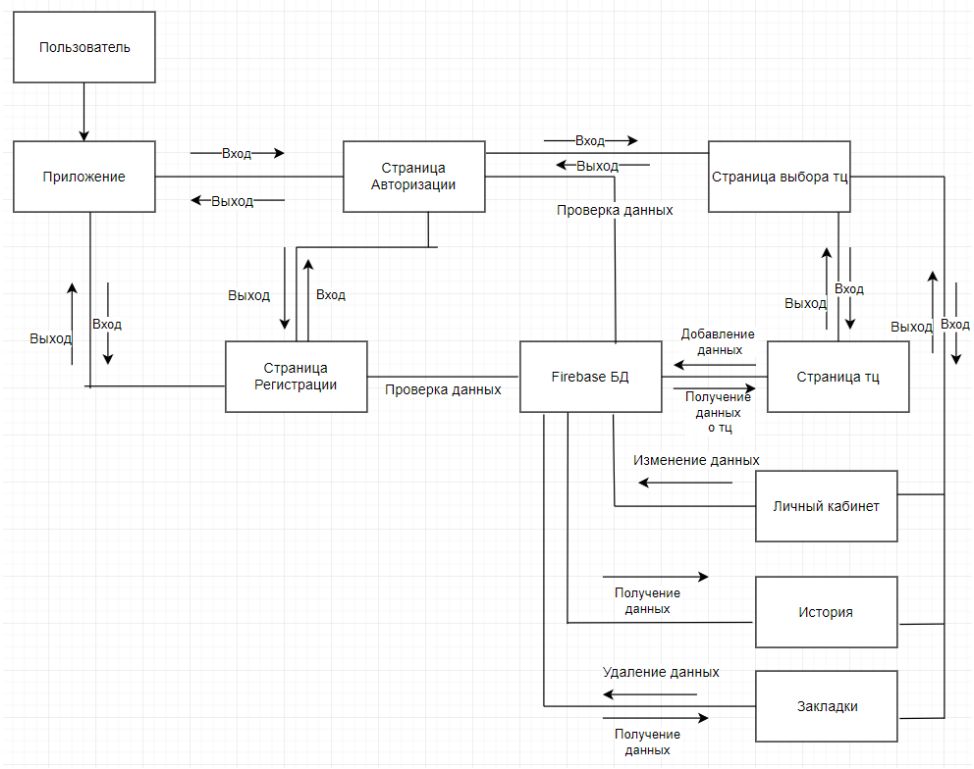


Рисунок 13 – Диаграмма кооперации мобильного приложения

Диаграмма кооперации отражает последовательность взаимодействия объектов, показанных на рисунке 13 в виде прямоугольников. После того как пользователь вошел в приложение, он должен пройти процесс авторизации и регистрации с проверкой введенных данных в базе данных. Авторизовавшись, пользователь перенаправляется на страницу выбора желаемого торгового центра. При выборе торгового центра из списка пользователь перенаправляется на страницу торгового центра с подробной информацией о нем. На странице приложения, пользователь может добавить в закладки, оставить оценку, посмотреть рейтинг. После чего оценка и закладки добавляются в базу данных. Также на странице выбора свайпом вправо открывается меню с возможностью выбора перехода в личный кабинет, закладки или историю.

На странице закладок находится список с раннее добавленными торговыми центрами с возможностью удаления. На странице истории находится список элементов, загруженный из базы данных. Каждый элемент списка содержит информацию о посещении и дате посещения. На странице личного кабинета находятся три поля ввода для изменения данных о пользователе.

2.3.5 Проектирования диаграммы состояний приложения

Диаграмма состояний предназначена для представления жизненного цикла объектов [11]. Основными элементами в данной диаграмме являются «Состояние» и «Переход». Диаграммы состояний служат для определения состояний, в которых может находится объект. На диаграмме показано одно сложное состояние «Ввод данных» и два простых. Основное отличие заключается в том, что сложное включает в себя несколько. На диаграмме состояний на рисунке 14 показан процесс жизни аккаунта на этапе регистрации в мобильном приложении.

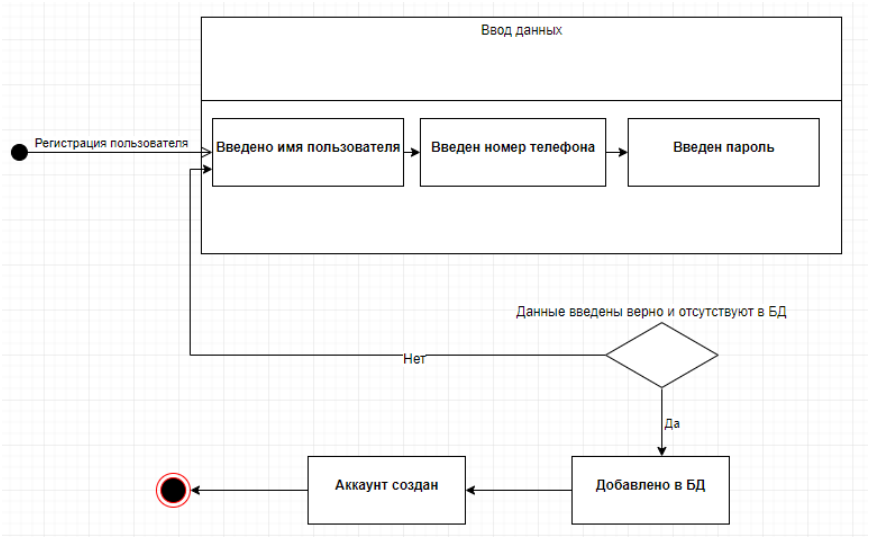


Рисунок 14 – Диаграмма состояний мобильного приложения

Для создания аккаунта, пользователю нужно зайти на страницу регистрации затем, чтобы попасть в состояние ввода данных как показано на диаграмме нужно ввести данные, затем происходит проверка на корректность данных и наличие записи в базе данных, если исход успешен, то данные добавляются в базу данных и происходит переход в создание аккаунта. При не удачном исходе валидации происходит переход на состояние ввода данных.

2.4 Проектирование пользовательского интерфейса приложения

По диаграмме вариантов использования был спроектирован графический интерфейс для разрабатываемого мобильного приложения.

При проектировании графического интерфейса были выделены основные особенности:

– доступность, т.е. чтобы каждый элемент интерфейса был интуитивно понятен любому пользователю;

– минимализм – минимально-допустимое количество элементов на каждой странице;

– соответствие контексту, каждая страница должна содержать соответствующий контент;

– привлекательность интерфейса – делает использование приложения более приятным нежели обычный стандартный набор стилей интерфейса.

На рисунке 15 представлен интерфейс начальной страницы.

Рисунок 15 – Начальная страница приложения

Основные элементы интерфейса страницы входа в приложение, представлены на рисунке 15:

1 – кнопка авторизации в приложении, компонент Button;

2 – кнопка регистрации в приложении, компонент Button;

3 – название приложения, компонент TextView.

При нажатии на кнопку 1 происходит перенаправление на страницу авторизации, при нажатии на кнопку 2 происходит перенаправление на страницу регистрации рисунок 16.

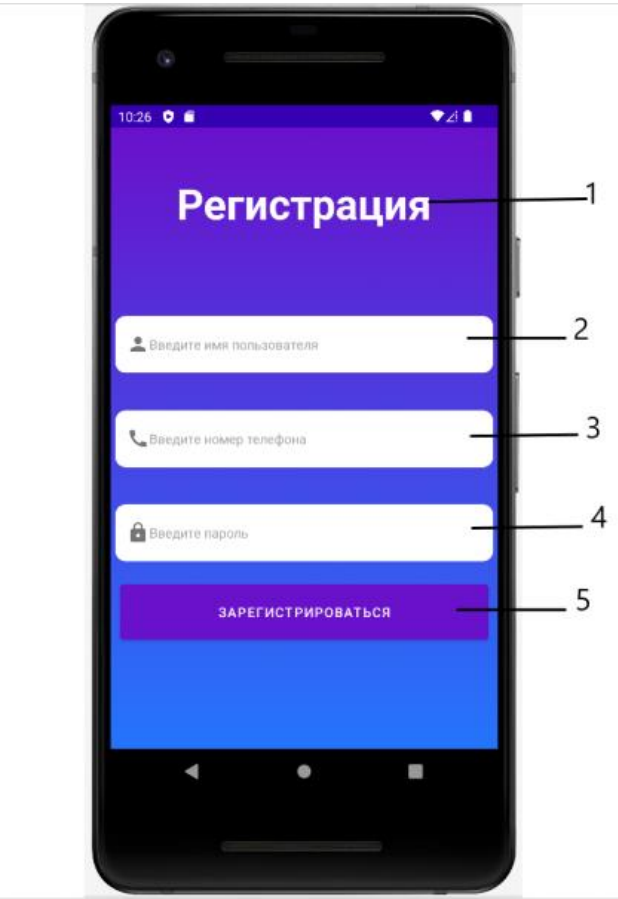


Рисунок 16 – Страница авторизации

Основные элементы интерфейса страницы регистрации, представлены на рисунке 16:

1 – название страницы приложения, компонент TextView;

2 – поле ввода имя пользователя, компонент EditText;

3 – поле ввода номера телефона пользователя, компонент EditText;

4 – поле ввода пароля пользователя, компонент EditText;

5 – кнопка регистрации в приложении, компонент Button.

После заполнения всех данных пользователя и клику по кнопке 5 происходит создание аккаунта и перенаправление на страницу авторизации (рисунок 17).

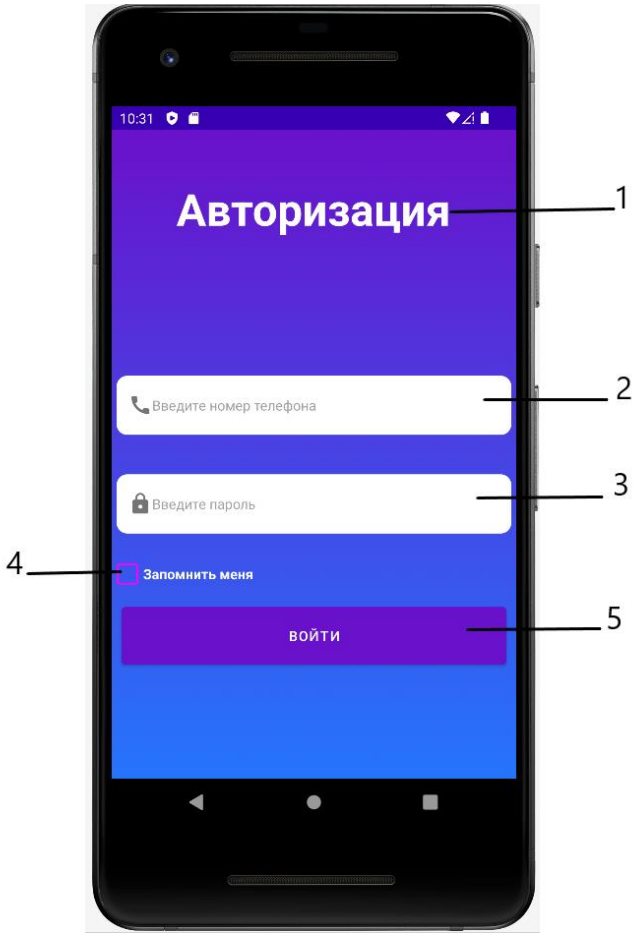


Рисунок 17 – Страница авторизации

Основные элементы интерфейса страницы авторизации представлены на рисунке 17:

1 – название страницы приложения, компонент TextView;

2 – поле ввода номера телефона пользователя, компонент EditText;

3 – поле ввода пароля пользователя, компонент EditText;

4 – флажок для запоминания номера телефона и пароля, компонент CheckBox;

5 – кнопка авторизации в приложении, компонент Button.

После заполнения всех данных пользовать может нажать на флажок 4, чтобы заходить в приложение без ввода данных, по клику по кнопке 5 происходит авторизация и перенаправление на главную страницу выбора торговых центров (рисунок 18).

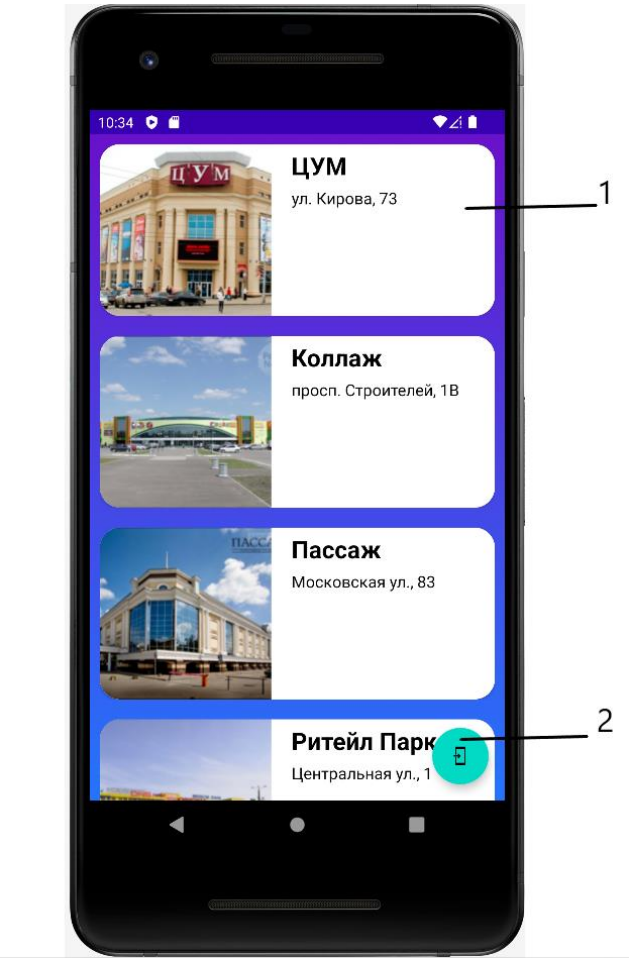


Рисунок 18 – Главная страница выбора торговых центров

Основные элементы интерфейса страницы выбора торгового центра представлены на рисунке 18:

1 – список торговых центров каждый элемент которого содержит название и адрес, компонент RecycleView;

2 – плавающая кнопка выхода из приложения, компонент FloatButton.

По клику на элемент списка происходит перенаправление на страницу торгового центра рисунок 19, также можно пролистывать элементы списка, по клику на кнопку 2 происходит выход из приложения. Так же по свайпу вправо открывается меню (рисунок 20).

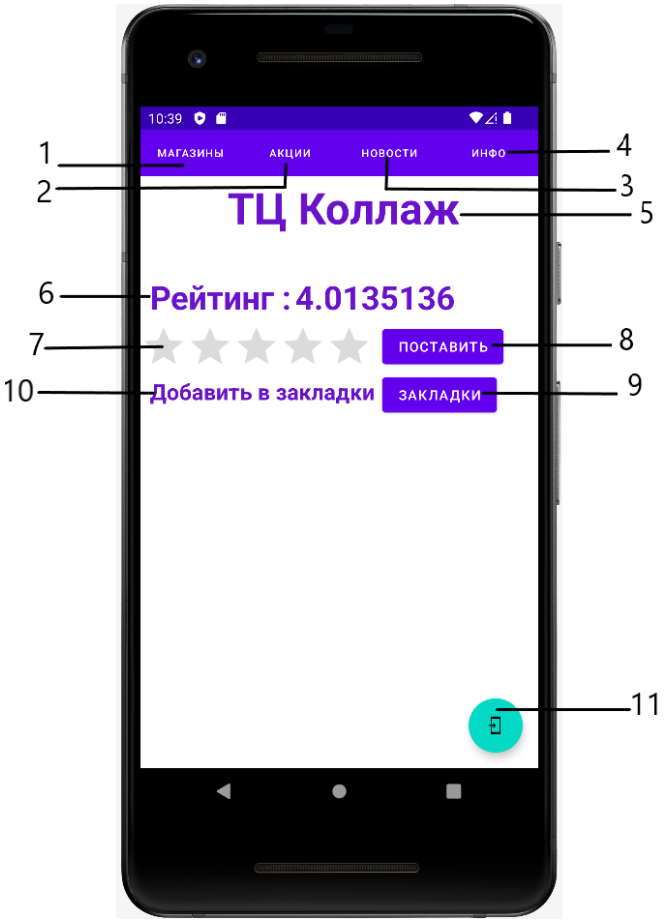


Рисунок 19 – Страница торгового центра

Основные элементы интерфейса страницы торгового центра, представлены на рисунке 19:

1 – кнопка для открытия магазинов торгового центра, компонент Button;

2 – кнопка для открытия акций торгового центра, компонент Button;

3 – кнопка для открытия новостей торгового центра, компонент Button;

4 – кнопка для открытия информации по торговому центру, компонент Button;

5 – название торгового центра, компонент TextView;

6 – рейтинг торгового центра, компонент TextView;

7 – колонка для оценки торгового центра, компонент RaitingBar;

8 – кнопка для оценки торгового центра, компонент Button;

9 – кнопка для добавления торгового центра в закладки, компонент Button;

10 – титл «Добавить в закладки», компонент TextView;

11 – плавающая кнопка выхода из приложения, компонент FloatButton.

По клику на кнопки 1, 2, 3, 4 пользователь будет перенаправлен на соответствующую страницу. Чтобы поставить оценку торговому центру, нужно кликнуть по звездам в RaitingBar – количество звезд соответствует оценке по пяти бальной шкале затем, нужно нажать на кнопку 8 для обработки оценки. После того как оценка поставлена, произойдет обновление рейтинга и элементы 8 и 9 примут статус Invisible, т.е. станут невидимы. Для добавления торгового центра в закладки нужно нажать на кнопку 9, после чего элементы 9 и 10 примут статус Invisible.

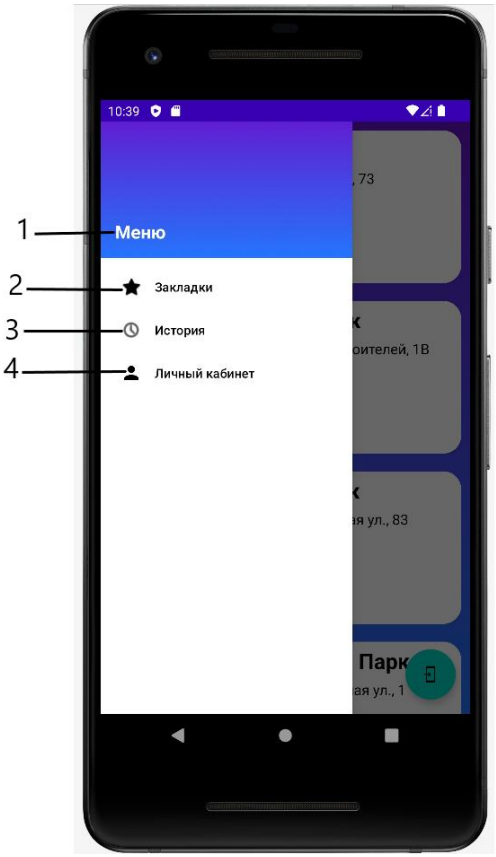


Рисунок 20 – Меню

Основные элементы интерфейса меню, представлены на рисунке 20:

1 – титл «Меню», компонент TextView;

2 – титл «Закладки», компонент TextView;

3 – титл «История», компонент TextView;

4 – титл «Личный кабинет», компонент TextView.

При нажатии на элементы 2, 3, 4 произойдет переход на соответствующую им страницу. Интерфейс страницы закладки изображен на рисунке 21.

Рисунок 21 – Страница закладок

Основные элементы интерфейса страницы закладок, представлены на рисунке 21:

1 – название страницы, компонент TextView;

2 – список закладок каждый элемент которого содержит название и кнопку удаления, компонент RecycleView;

3 – кнопка для удаления закладки, компонент Button.

При нажатии на кнопку 3 пользователь может удалить закладку. Интерфейс страницы истории представлен на рисунке 22.

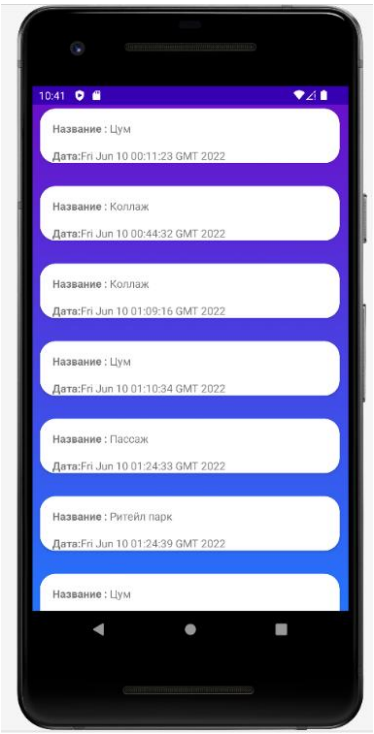


Рисунок 22 – Страница истории

Интерфейс данной страницы на рисунке 24 включает в себя список истории посещений каждый, элемент которого содержит название и дату посещения, компонент RecycleView.

Интерфейс страницы личного кабинета представлен на рисунке 23.

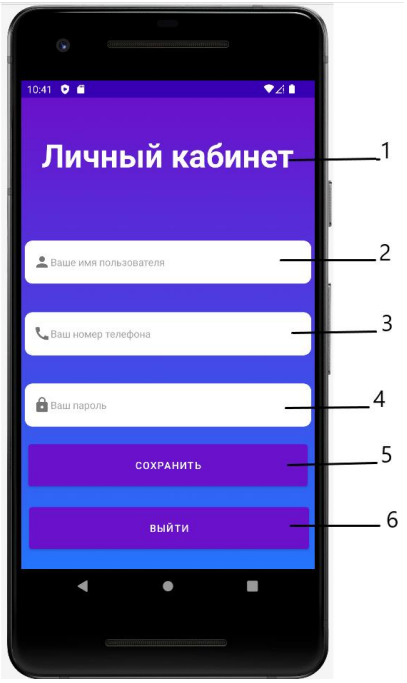


Рисунок 23 – Личный кабинет

Основные элементы интерфейса страницы личного кабинета, представлены на рисунке 23:

1 – название страницы приложения, компонент TextView;

2 – поле ввода имя пользователя, компонент EditText;

3 – поле ввода номера телефона пользователя, компонент EditText;

4 – поле ввода пароля пользователя, компонент EditText;

5 – кнопка сохранения данных, компонент Button;

6 – кнопка выхода без сохранения данных, компонент Button.

Для изменения данных аккаунта нужно ввести данные в поле, после чего нажать на кнопку 5, для выхода нажать на кнопку 6.

Страница магазинов для выбранного торгового центра, представлена на рисунке 24.

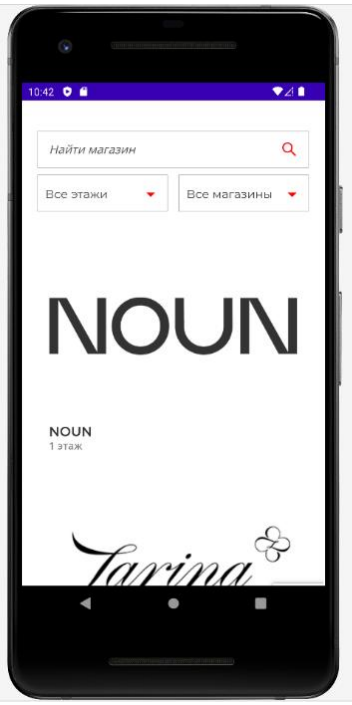


Рисунок 24 – Страница магазины

Интерфейс данной страницы на рисунке 24 включает в себя встроенную веб-страницу магазинов торгового центра, компонент Webview.

Страница содержит WebView для отображения html-страницы. Страница новости для выбранного торгового центра, представлена на рисунке 24.

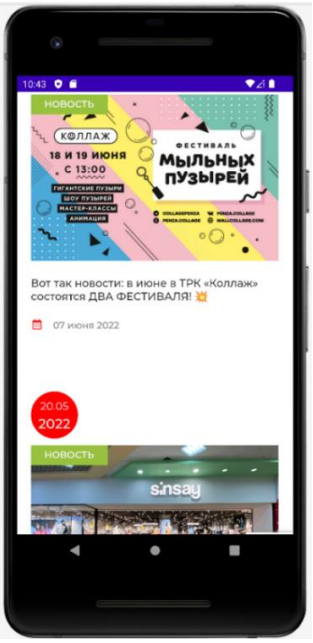


Рисунок 25 – Страница новости

Интерфейс данной страницы на рисунке 25 включает в себя встроенную. веб-страницу магазинов торгового центра, компонент Webview.

Страница содержит WebView для отображения html-страницы. Страница информации для выбранного торгового центра, представлена на рисунке 25.

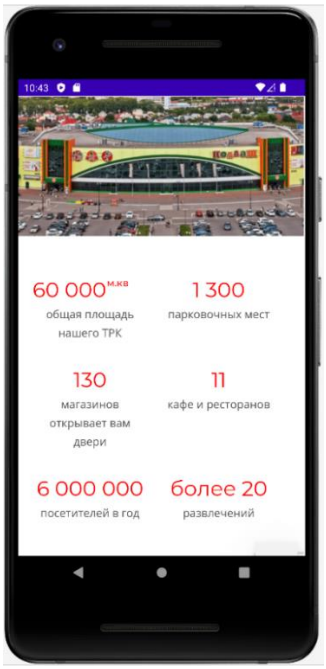


Рисунок 26 – Страница информации

Интерфейс данной страницы на рисунке 26 включает в себя встроенную. веб-страницу магазинов торгового центра, компонент Webview.

Страница содержит WebView для отображения html-страницы. Страница акций для выбранного торгового центра представлена на рисунке 27.

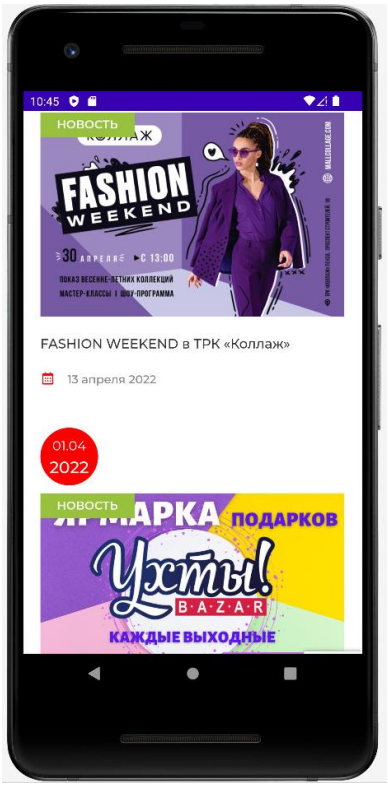


Рисунок 27 – Страница акции

Интерфейс данной страницы на рисунке 27 включает в себя встроенную веб-страницу магазинов торгового центра, компонент Webview.

Таким образом мобильное приложение для онлайн-шопинга имеет страницы авторизации, регистрации, личного кабинета, истории, закладок, страницу выбора торгового центра, главную страницу торгового центра и вкладки для перехода на страницы магазина, акций, новостей, информации по выбранному торговому центру. Проектирование интерфейса было выполнено в соответствии с требованиями, выделенными раннее.

3 Реализация программного обеспечения мобильного приложения

Программные средства мобильного приложения для онлайн-шопинга состоят из ряда компонентов. Компоненты приложения представлены на рисунке 28.

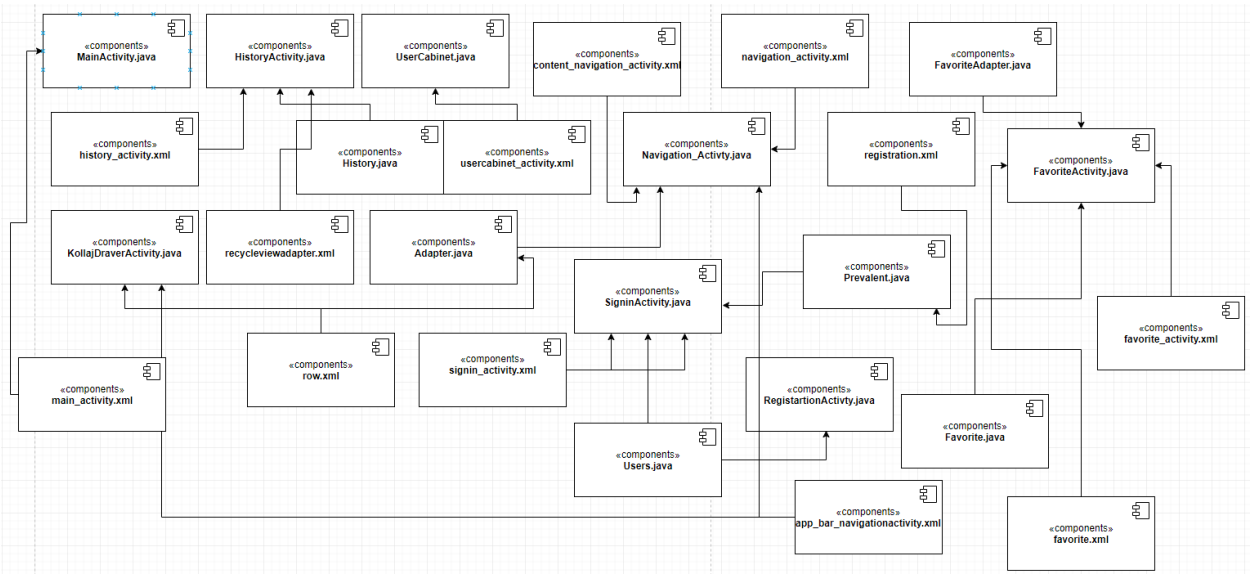


Рисунок 28 – Диаграмма компонентов

Ниже в таблице 23, приведено описание компонентов мобильного приложения для онлайн-шопинга.

Таблица 23 – Описание компонентов системы

№	Название	Описание
1	HistoryActivity.java	Функциональность страницы истории.
2	UserCabinet.java	Функциональность страницы личного кабинета.
3	SigninActivity.java	Функциональность страницы авторизации.
4	RegistrationActivity.java	Функциональность страницы регистрации.
5	NavigationActivity.java	Функциональность страницы выбора торгового центра.
6	Favoritedrawer.java	Адаптер для записи в список данных из базы данных
7	User.java	Описание данных пользователя.
	History.java	Описание данных истории.
9	Favorite.java	Описание данных закладок.
10	KollajDraverActivity.java	Функциональность страницы торгового центра.
11	MainActivity.java	Функциональность начальной страницы приложения, точка входа в программу.
12	Adapter.java	Адаптер для записи в список торговых центров

Продолжение таблицы 23

№	Название	Описание
13	Prevalent.java	Описание данных пользователя, использующего приложение.
14	FavoriteActivity.java	Функциональность страницы закладки.
15	history_activity.xml	Описание графического интерфейса страницы истории.
16	main_activty.xml	Описание графического интерфейса начальной страницы.
17	recycleviewadapter.xml	Графическое описание элемента таблицы для списка.
18	row.xml	Графическое описание элемента таблицы для списка.
19	signin_activivty.xml	Описание графического интерфейса страницы авторизации.
20	usercabinet_activity.xml	Описание графического интерфейса страницы личного кабинета.
21	content_navigation_activity.xml	Описание контента, находящегося на странице.
22	navigation_activity.xml	Описание графического интерфейса страницы экрана выбора торгового центра.
23	registration.xml	Описание графического интерфейса страницы регистрации.
24	app_bar_navigationactivity.xml	Описание графического интерфейса меню.
25	favorite_activity.xml	Описание графического интерфейса страницы закладок.
26	favorite.xml	Графическое описание элемента таблицы для списка.

На рисунке 29 представлена диаграмма развертывания мобильного приложения для онлайн-шопинга, которая изображает основные элементы системы.

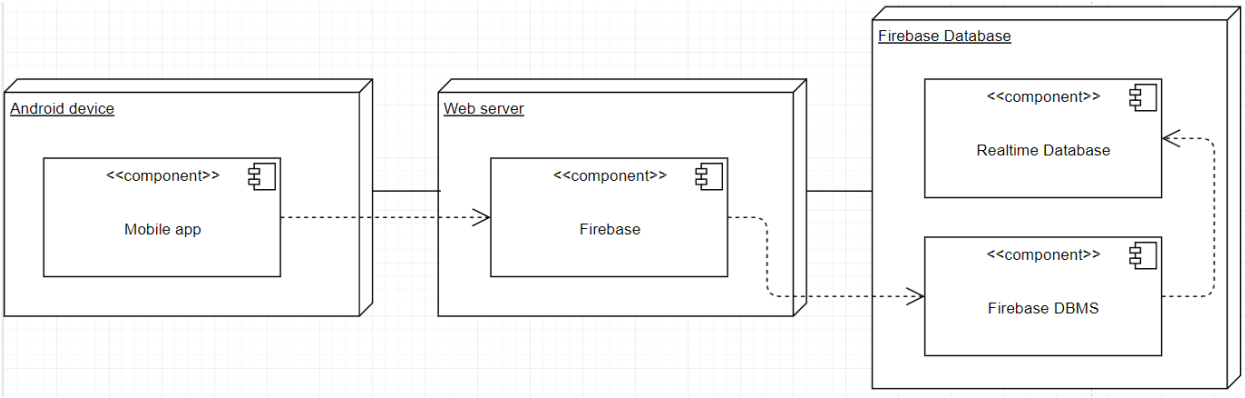


Рисунок 29 – Диаграмма развертывания

Архитектура приложения состоит из трех основных элементов: мобильное устройство на операционной системе Android, веб сервер Firebase и база данных Firebase. Используя мобильное устройство пользователь взаимодействует с мобильным приложением. В свою очередь, Firebase сервер принимает запросы пользователей, затем обращается к базе данных, после получает ответ и отправляет ответ пользователю приложения в формате JSON.

4 Контроль качества программного обеспечения

4.1 Функциональное тестирование

Функциональное тестирование – это один из видов тестирования ПО, проверяющий систему на соответствие функциональным требованиям, такой метод тестирования позволяет обнаружить расхождения поведения функций мобильного приложения [12]. Функциональное тестирование проведено для процесса регистрации пользователей.

Таблица 24 – Классы эквивалентности

Поле номер телефона	Правильные классы	Неправильные классы
Длина номера	11	[0,10), (12, +]
Номер телефона	Состоит из цифр	Содержит другие символы
Занятость номера	Свободен	Занят
Первые 2 символа	89	Содержит другие символы

Для построения тестов воспользуемся методом функциональных диаграмм. Для регистрации пользователя в приложении построена функциональная диаграмма, представленная на рисунке 30.

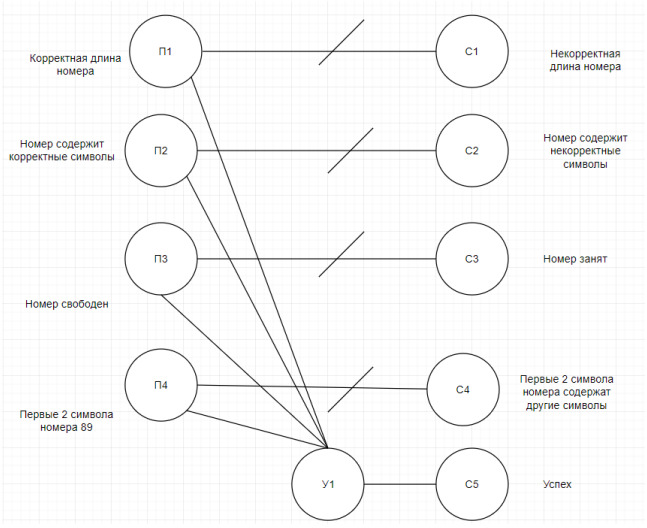


Рисунок 30 – Функциональная диаграмма

На основе диаграммы была построена таблица решений. Таблица решений представлена в таблице 25.

Таблица 25 – Таблица решений

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
П1	0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	1
П2	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1	1
П3	0	0	1	1	0	0	1	0	0	1	1	1	0	0	1	1
П4	0	1	0	1	0	1	0	0	1	0	1	1	0	1	0	1
С1	1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
С2	1	1	1	1	0	0	0	1	1	1	1	0	0	0	0	0
С3	1	1	0	0	1	1	0	1	1	0	0	0	1	1	0	0
С4	1	0	1	0	1	0	0	1	0	1	0	0	1	0	1	0
С5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

В таблице 26 представлен набор тестов, составленные на основе таблицы решений.

Таблица 26 – Набор тестов

№	Входные данные	Ожидаемый результат	Результат
1	Номер телефона: 11345addfger78910	(С1, С2, С3, С4)	Тест пройден
2	Номер телефона: 891243255344	(С1, С2, С3)	Тест пройден
3	Номер телефона: 12345678910	(С1, С2, С4)	Тест пройден
4	Номер телефона: 1a	(С1, С2)	Тест пройден
5	Номер телефона: 123456fhgh	(С1, С3, С4)	Тест пройден
6	Номер телефона: 1adf	(С1, С3)	Тест пройден (рисунок В.2)
7	Номер телефона: 8a	(С1, С4)	Тест пройден (рисунок В.3)
8	Номер телефона: wertyu2ox456	(С2, С3, С4)	Тест пройден
9	Номер телефона: 89rbvf567102	(С2, С3)	Тест пройден
10	Номер телефона: asfghgj4e5323	(С2, С4)	Тест пройден
11	Номер телефона: 8904264478-	(С2)	Тест пройден
12	Номер телефона: 8904	(С1)	Тест пройден (рисунок В.1)
13	Номер телефона: wetrye2	(С3, С4)	Тест пройден
14	Номер телефона: 89042644788	(С3)	Тест пройден (рисунок В.3)
15	Номер телефона: 12345678910	(С4)	Тест пройден (рисунок В.4)
16	Номер телефона: 89042644785	(С5)	Тест пройден (рисунок В.5)

При функциональном тестировании приложения было выявлено, что приложение соответствует функциям, все тесты успешно завершились. Результаты тестирования представлены в Приложении В.

4.2 Метрики кода

Для расчета метрик кода используем программу LocMetrics. Программа анализирует код программного проекта [13].

Результаты представлены на рисунке 31.

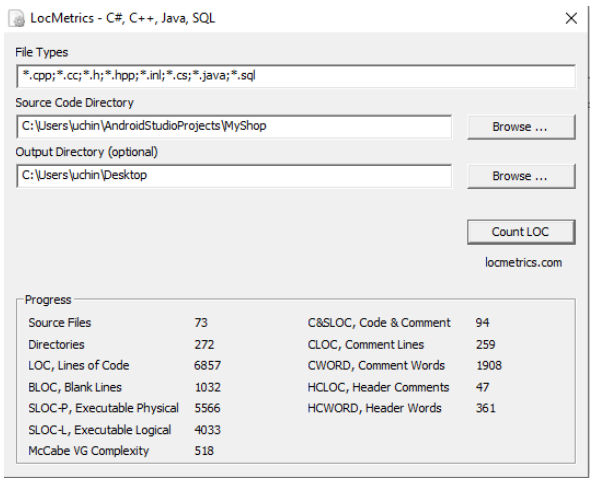


Рисунок 31 – Метрика кода

По результатам анализа программного проекта мобильного приложения можно увидеть на рисунке 32, что количество исходных файлов равно 73 (Source Files), 6857 строк кода (Lines of Code), 1032 пустых строки (Blank Lines).

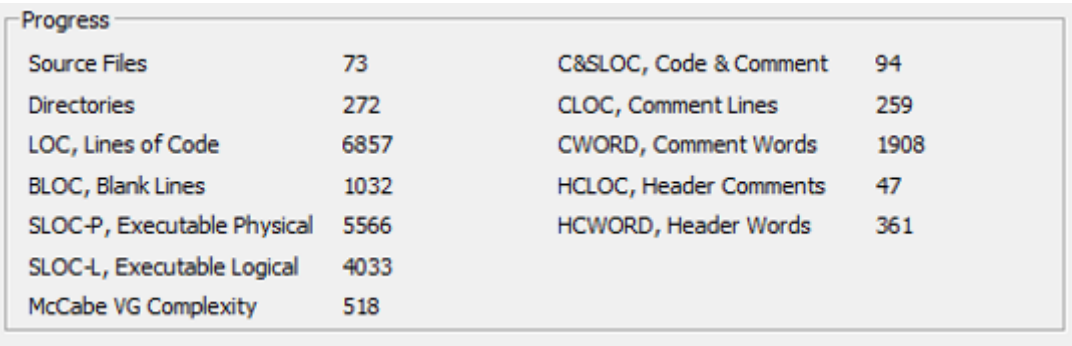


Рисунок 32 – Количественные метрики

Цикломатическая сложность – это количество линейно независимых маршрутов через программный код, в нашем случае цикломатическая сложность равна значению 518 (McCabe VG Complexity), что означает, что программа является сложной.

Так же при подключении плагина Statistic в Android Studio появляется функция сбора статистики всего программного проекта, на рисунке 33 показана статистика приложения для онлайн-шопинга.

Extension	Count	Size SUM	Size MIN	Size MAX	Size AVG
bat *(BAT files)*	1x	2kB	2kB	2kB	2kB
gitignore *(GITIGNORE files)*	2x	0kB	0kB	0kB	0kB
gradle *(GRADLE files)*	2x	0kB	0kB	0kB	0kB
java *(Java classes)*	28x	117kB	0kB	11kB	4kB
json *(JSON files)*	1x	1kB	1kB	1kB	1kB
plantuml *(PLANTUML files)*	7x	16kB	0kB	12kB	2kB
pro *(PRO files)*	1x	0kB	0kB	0kB	0kB
properties *(Java properties files)*	3x	1kB	0kB	1kB	0kB
webp *(WEBP files)*	10x	33kB	0kB	7kB	3kB
xml *(XML configuration file)*	98x	117kB	0kB	5kB	1kB

Рисунок 33 – статистка приложения

На рисунке 32 видно, что чаще всего используется язык xml его количество равно 98 и язык Java его количество равно 28.

4.3 Планирование разработки и оценка бюджета

Планирование и разработка бюджета была выполнена в программе Microsoft Project [14]. В процессе планирования были выделены основные этапы проекта, а именно 24 пункта.

Список этапов планирования разработки программного проекта изображен на рисунке 34.

Название задачи	Длительность	Начало	Окончание	Предшественни	Названия ресурсов
Начало проекта	0 дней	Ср 13.04.22	Ср 13.04.22		
Анализ требований	**12 дней**	**Ср 13.04.22**	**Чт 28.04.22**		
Анализ аналогов	3 дней	Ср 13.04.22	Чт 21.04.22	1	Программист1[50%];К
Анализ предметной области	3 дней	Пт 22.04.22	Ср 27.04.22	3	Программист2[50%]; Программист1[50%];К
Принятие окончательного решени	1 день	Чт 28.04.22	Чт 28.04.22	4	Программист1;Програ
Анализ требований завершен	0 дней	Чт 28.04.22	Чт 28.04.22	5	
Выбор средств реализации проек	**10 дней**	**Ср 13.04.22**	**Вт 26.04.22**		
Анализ возможных программных	3 дней	Ср 13.04.22	Вт 19.04.22	1	Программист1[50%];П
Выбор программных средств	2 дней	Ср 20.04.22	Пт 22.04.22	8	Программист1[50%];П
Выбор средств завершен	2 дней	Пн 25.04.22	Вт 26.04.22	9	
Разработка проекта	**16 дней**	**Пт 29.04.22**	**Пт 20.05.22**		
Разработка структуры	3 дней	Пт 29.04.22	Пт 06.05.22	10;6	Программист1[25%];П
Разработка интерфейса	4 дней	Пн 09.05.22	Чт 12.05.22	12	Программист1[50%];П
Разработка макета интерфейса	4 дней	Пн 09.05.22	Чт 12.05.22	12	Программист1[50%]; Программист2[50%];М
Реализация приложения	6 дней	Пт 13.05.22	Пт 20.05.22	14	Программист1[70%];П
Разработка приложения завершено	0 дней	Пт 20.05.22	Пт 20.05.22	15	
Отладка	**22 дней**	**Чт 19.05.22**	**Пт 17.06.22**		
Тестирование интерфейса	2 дней	Пн 23.05.22	Чт 26.05.22	16	Программист1[60%]; Программист2[70%];К
Исправление ошибок	2 дней	Пт 27.05.22	Пн 30.05.22	18	Программист1;Програ
Тестирование завершено	0 дней	Пн 30.05.22	Пн 30.05.22	19	
Внедрение програмного средства	2 дней	Вт 31.05.22	Ср 01.06.22	20	Программист1; Программист2;Компь
Исправление ошибок внедрения	2 дней	Чт 02.06.22	Чт 09.06.22	21	Программист1[50%]; Программист2[50%];И
Внедрение завершено	0 дней	Чт 09.06.22	Пт 17.06.22	22	Программист1;Програ
Конец проекта	0 дней	Пт 17.06.22	Пт 17.06.22	23	

Рисунок 34 – Список этапов проекта

Описание этапов разработки представлены в таблице 33. Во время планирования были выделены основные ресурсы для выполнения программного проекта изображенные в таблице 34.

В таблице 27 представлены задачи, которые представляют этапы создания проекта. Для каждой задачи были описаны предшественники и длительность каждой задачи в днях

Таблица 27 – Перечень задач проекта

№	Название задачи	Длительность	Предшественники
1	Начало проекта	-	-
2	Анализ требований	12 дней	-
3	Анализ аналогичных решений	3 дня	1
4	Анализ предметной области	3 дня	3
5	Принятие окончательного решения	1 день	4
6	Анализ требований завершен	-	5
7	Выбор средств реализации проекта	10 дней	-

Продолжение таблицы 27

№	Название задачи	Длительность	Предшественники
8	Анализ возможных программных средств	3 дня	1
9	Выбор программного средства для реализации	2 дня	8
10	Выбор средств завершен	2 дня	9
11	Реализация проекта	16 дней	-
12	Разработка структуры	3 дня	6; 10
13	Разработка интерфейса	4 дня	-
14	Разработка макета интерфейса приложения	4 дня	12
15	Реализация	6 дней	14
16	Реализация проекта завершена	-	11; 16
17	Отладка	22 дня	-
18	Тестирование пользовательского интерфейса	2 дня	16
19	Исправление ошибок	2 дня	18
21	Тестирование завершено	-	19
21	Внедрение программного средства	7 дней	20
22	Исправления ошибок внедрения	7 дней	21
23	Внедрение завершено	-	22
24	Конец проекта	-	23

Из таблицы, представленной выше видно, что для реализации проекта необходимо выполнить 24 задачи.

Таблица 28 – Список ресурсов проекта

Название ресурса	Тип	Стандартная ставка	Ставка сверхурочных
Программист1	Трудовой	200 Р/ч	250 Р/ч
Программист2	Трудовой	300 Р/ч	350 Р/ч
Дизайнер	Трудовой	250 Р/ч	300Р/ч
Интернет	Затраты	1000р	

Продолжение таблицы 28

Название ресурса	Тип	Стандартная ставка	Ставка сверхурочных
Компьютер	Материальный	1000Р	
Телефон	Материальный	1000 Р	

В результате планирования списка задач и списка ресурсов была построена диаграмма Ганта рисунок 35.

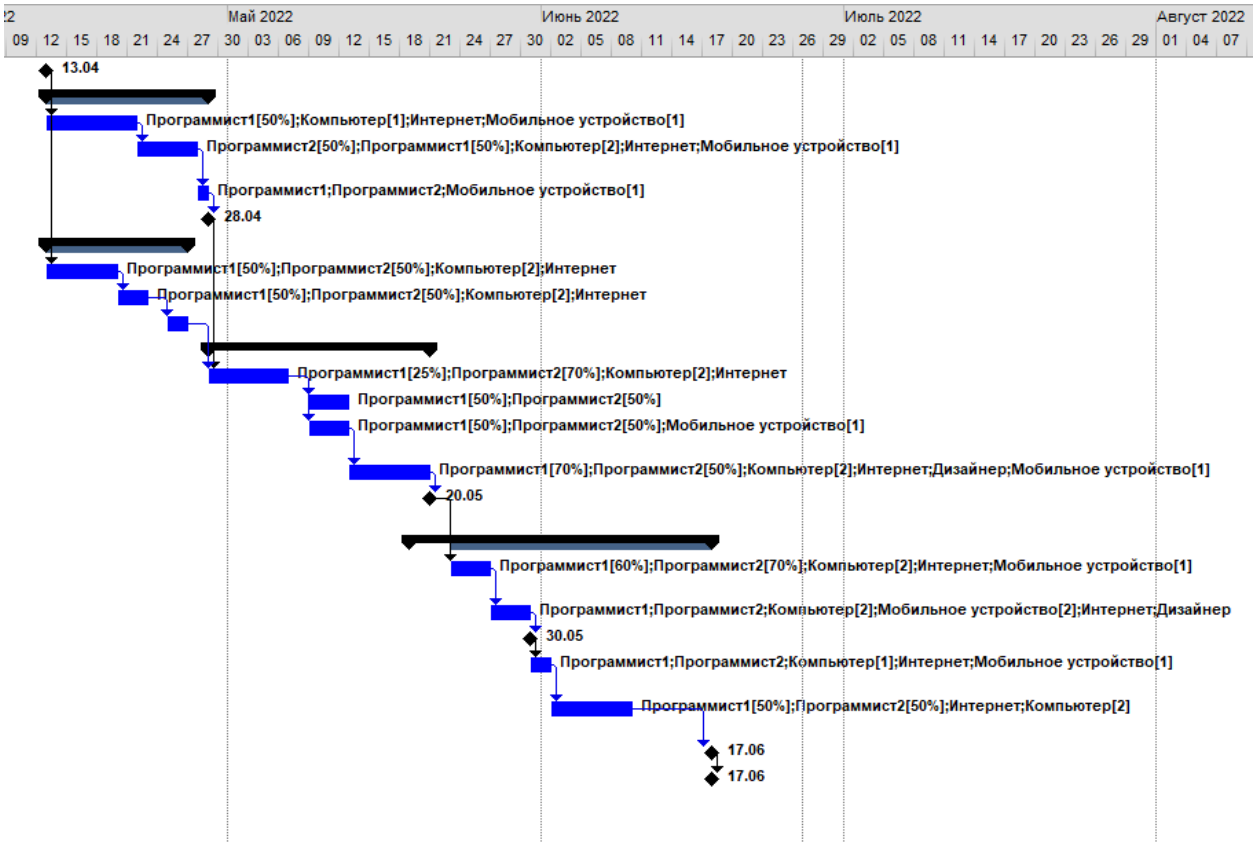
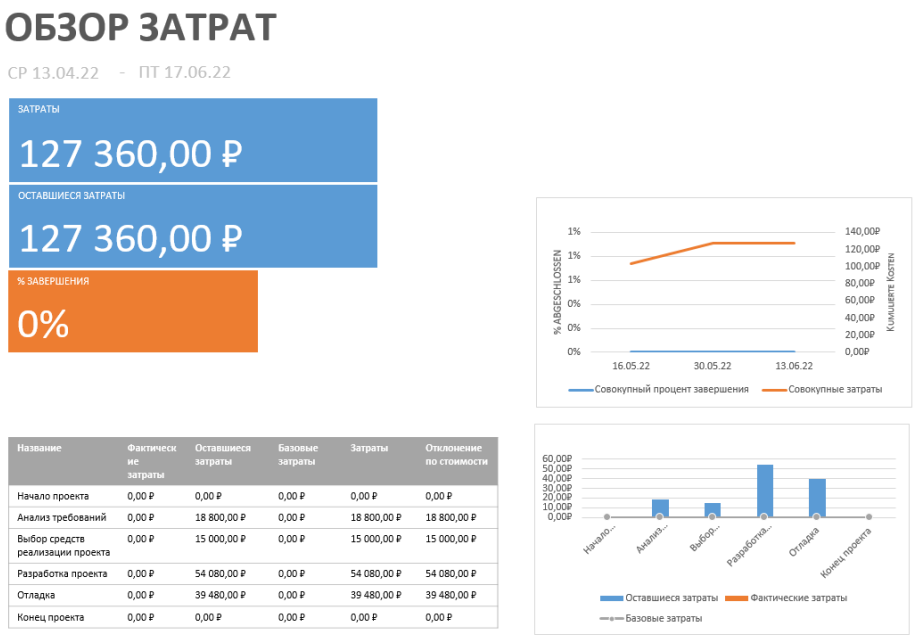


Рисунок 35 – Диаграмма Ганта

На рисунке 36 представлен обзор программного проекта. Из рисунка видно, что общая длительность составляет 50 дней, а предполагаемые затраты 127 360 руб.



Название	Фактические затраты	Оставшиеся затраты	Базовые затраты	Затраты	Отклонение по стоимости
Начало проекта	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽
Анализ требований	0,00 ₽	18 800,00 ₽	0,00 ₽	18 800,00 ₽	18 800,00 ₽
Выбор средств реализации проекта	0,00 ₽	15 000,00 ₽	0,00 ₽	15 000,00 ₽	15 000,00 ₽
Разработка проекта	0,00 ₽	54 080,00 ₽	0,00 ₽	54 080,00 ₽	54 080,00 ₽
Отладка	0,00 ₽	39 480,00 ₽	0,00 ₽	39 480,00 ₽	39 480,00 ₽
Конец проекта	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽

Рисунок 36 – Обзорный отчет проекта

В результате планирования разработки и оценки бюджета были выделены основные этапы разработки определены ресурсы, произведена оценка затрат и длительность выполнения программного проекта.

Заключение

В ходе выполнения выпускной квалификационной работы разработано мобильное приложение для онлайн-шопинга. При разработке использовался язык программирования Java, язык разметки xml для создания графического интерфейса, в роли базы данных выступает Firebase.Приложение работает на мобильных устройствах с минимальной версией Android 5.1.

Проведен анализ программных продуктов подобного назначения, выделен основной функционал для разрабатываемого приложения, описаны варианты использования, построена и описана диаграмма классов.

Архитектура приложения имеет тип клиент-сервер, в роли клиента выступает приложение для онлайн-шопинга, сервером является облачный сервис Firebase.

Выполнено функциональное тестирование. Все тесты успешно выполнены. Вычислены метрики программного проекта, определена цикломатическая сложность равная 518, что говорит о сложности приложения.

Проведено планирование и оценка бюджета для разрабатываемого приложения в программе Microsoft Project. В результате было выявлено, что длительность разработки равна 50 дней, затраты составили 127 360 руб.

Таким образом, в результате выполнения выпускной квалификационной работы разработано мобильное приложение для онлайн-шопинга, отвечающее всем требованиям технического задания.

Список использованных источников

1. Firebase Realtime Database [Электронный ресурс]. URL: https://firebase.google.com/products/realtime-database (дата обращения: 28.05.2022)

2. Firebase Storage [Электронный ресурс]. URL: https://firebase.google.com/products/storage (дата обращения: 28.05.2022)

3. NoSQL подход [Электронный ресурс]. URL: https://www.oracle.com/cis/database/nosql/what-is-nosql/ (дата обращения: 28.05.2022)

4. Понятие толстого и тонкого клиента [Электронный ресурс]. URL: https://testmatick.com/ru/ponyatie-tonkogo-i-tolstogo-klienta/ (дата обращения: 28.05.2022)

5. Концептуальная модель базы данных [Электронный ресурс]. URL: http://wiki.it-wiki.org.ua/doku.php/db_intro (дата обращения: 28.05.2022)

6. Логическая модель базы данных [Электронный ресурс]. URL: https://dl.nure.ua/pluginfile.php/550/mod_resource/content/2/content/content4.html(дата обращения: 28.05.2022)

7. Физическая модель базы данных [Электронный ресурс]. URL: https://scienceforum.ru/2017/article/2017030333 (дата обращения: 28.05.2022)

8. Разработка баз данных в Visual Paradigm [Электронный ресурс]. URL: http://sp.cs.msu.ru/ooap/exerb2020.html (дата обращения: 28.05.2022)

9. Диаграмма деятельности проектирование [Электронный ресурс]. URL: https://nationalteam.worldskills.ru/skills/proektirovanie-diagrammy-deyatelnosti-uml-activity-diagram/ (дата обращения: 28.05.2022)

10. Диаграмма кооперации [Электронный ресурс]. URL: http://www.telenir.net/uchebniki/samouchitel_uml/p9.php (дата обращения: 28.05.2022)

11. Диаграмма состояний [Электронный ресурс]. URL: https://nationalteam.worldskills.ru/skills/proektirovanie-diagrammy-sostoyaniy-uml-statechart-diagram/ (дата обращения: 28.05.2022)

12. Функциональное тестирование [Электронный ресурс]. URL: https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/ (дата обращения: 28.05.2022)

13. LockMetrics документирование кода [Электронный ресурс]. URL: https://mydocx.ru/12-19950.html (дата обращения: 28.05.2022)

14. Microsoft Project управление программными проектами [Электронный ресурс]. URL: https://habr.com/ru/post/151593/ (дата обращения: 28.05.2022)

Приложение А

ГЛОССАРИЙ

(справочное)

База данных (БД) – Это модель данных позволяющая организовывать информацию о группе объектов, имеющих одинаковые параметры.

Unified Modeling Language (UML) – графический язык моделирования, служащий для моделирования процессов.

Система управления базами данных (СУБД) – Различные языки, программы и языковые средства, обеспечивающие управлением и созданием базы данных.

Свайп (Swipe) – специальный жест, суть которого заключается в том, чтобы положить палец на экран и вести в нужную сторону, при этом не отрывая палец от экрана.

Приложение Б

ЛИСТИНГ ПРИЛОЖЕНИЯ

(рекомендуемое)

Листинг Б.1 – Файл Adapter.java

```
public class Adapter {

    String  data1[], data2[];
    int  images[];
    private  String  using = " Users ";
    String username;
    Context   context;
    public  Adapter
(Context  ct, String Sname[], String  Sdescription [],
 int img[] ){
        context = ct;
        data1 =  Sname;
        data2 = Sdescription;
        images  = img;
    }
    @NonNull
    @Override
    public ViewHolder  onCreateViewHolder
(@NonNull  ViewGroup parent, int viewType) {

        LayoutInflater  inflater =
LayoutInflater.from(context);
        View view =
 inflater.inflate(R.layout.my_row,parent,false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder (@NonNull ViewHolder holder,
@SuppressLint("RecyclerView")
 int position) {
    holder. myText1. setText(data1[position]);
        holder. myText2. setText(data2[position]);
        holder. myImage. setImageResource
(images[position]);

        holder.mainLayout.setOnClickListener
(new View.OnClickListener() {
            @Override
            public void onClick
(View view) {
             if (images [position] == R.drawable.vl){
                    SaveHistoryToDaDatabase
("Высшая лига");
                    Intent intent =
new Intent
(context, VLDRAVERACTIVITY.class);
                    context.startActivity(intent);

                }
                else if (images[position] == R. drawable.cym){
                    SaveHistoryToDaDatabase
("Цум");
                            Intent intent =
new Intent(context,CYMDRAVERACTIVITY.class);
                            context. startActivity(intent);
                        }
                                            else if
(images[position]==R.drawable.kollaj){
                                                Intent intent = new
```

```
Intent(context, KOLLAJdraveractivityActivity.class);
    SaveHistoryToDaDatabase
("Коллаж");

context.startActivity(intent);
                                                }
  else if
(images [position]==R.drawable.passaj){

SaveHistoryToDaDatabase
("Пассаж");
       Intent intent =
 new Intent(context, PASSAJDRAVERACTIVITY.class);

context.startActivity(intent);
                                                              }

   else if
(images[position] == R.drawable.riteyl){

SaveHistoryToDaDatabase
("Ритейл парк");

 Intent intent =
new Intent(context, RITEYLDRAVERACTIVITY.class);

context.startActivity(intent);
                                                       }
        else if(images[position] == R.drawable.sanmart){
       SaveHistoryToDaDatabase
("Сан март");//
                              Intent intent =
new Intent(context, SANVARTDRAVERACTIVITY.class);
                 context.
startActivity(intent);
                         }
                         else if (images[position]==R.drawable.syvorovski){
                    Intent intent =
 new Intent(context,
SYVOROVSKIYDRAVERACTIVITY. class);
   SaveHistoryToDaDatabase
("Суворовский");
                    context.startActivity(intent);
                }
            }
        });

    }

    public void SaveHistoryToDaDatabase
(String name) {
        final DatabaseReference Root;
        Root=
FirebaseDatabase.getInstance().getReference().child("Users").child(Prevalent.
UserCurrentEmail).child("History");
        Root.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot  snapshot) {
                Date currentTime =
 Calendar.getInstance().getTime();
                String date =
currentTime.toString();
                HashMap<String,Object> UserDataMap =
```

```
new HashMap<>();
                UserDataMap .put(" NameHistory ",name);
                UserDataMap.put("Date", date);
                Root.child(date).updateChildren(UserDataMap.
addOnCompleteListener (new OnCompleteListener <Void>() {
                    @Override
                    public void onComplete(@NonNull Task<Void> task) {
                        if (task. isSuccessful()){

                        }
                        else {

                        }
                    }
                });

            }

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

            }
        });
    }

    @Override
    public int getItemCount() {

        return images.length;
```

Листинг Б.2 – Файл FavoriteActivity.java

```
public class FavoriteActivity {

    RecyclerView recyclerView;
    ArrayList<Favorite> list;

    DatabaseReference databaseReference;
    MyAdapter adapter;

recyclerView =
(RecyclerView) findViewById(R.id.list_favorite);
databaseReference = FirebaseDatabase.getInstance().
getReference().child("Users").child(Prevalent.UserCurrentEmail).child("Favori
te");
list = new ArrayList < Favorite>();
recyclerView.setLayoutManager(new LinearLayoutManager(this));
adapter = new MyAdapter(this,list);
recyclerView.setAdapter(adapter);

databaseReference. addValueEventListener
(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot snapshot) {
        ;
            for
(DataSnapshot dataSnapshot : snapshot.getChildren()) {
   Favorite users =  dataSnapshot .getValue (Favorite. class);
               list.add(users);
            }
        adapter. notifyDataSetChanged
```

```
();
    }
() {
        super. onBackPressed();

startActivity (
new Intent
(FavoriteActivity
. this,Navigation_Activity. class)
);
        finish();
    }
```

Листинг Б.3 – Файл HistoryActivity.java

```
public class  HistoryActivity {
    RecyclerView  recyclerView;
    ArrayList <History> list ;
    DatabaseReference  databaseReference;
    HistoryAdapter  adapter;


    public void onBackPressed() {
        super.onBackPressed();
        startActivity
(new Intent
(HistoryActivity.this, Navigation_Activity.class));
         finish();
    }

    //FavoriteAdapter mainAdapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate
(savedInstanceState);
        setContentView ( R.layout.activity_history );


        recyclerView  = (RecyclerView)
 findViewById(R.id.list_history);
        databaseReference =
FirebaseDatabase.getInstance().getReference().child("Users").child(Prevalent.
UserCurrentEmail).child("History");
        list = new ArrayList<History>();
        recyclerView.setLayoutManager
(new LinearLayoutManager
(this));
        adapter =
new HistoryAdapter (this, list);
        recyclerView. setAdapter (adapter);
        databaseReference.addValueEventListener
( new  ValueEventListener () {
            @Override
            public void onDataChange(@NonNull DataSnapshot  snapshot) {
                ;
                for
(DataSnapshot  dataSnapshot :
 Snapshot .getChildren()) {
        History users =
dataSnapshot. getValue(History.class);
                    list.add(users);
                }
```

```
                adapter.notifyDataSetChanged();
            }

          });

    }

}
```

Листинг Б.4 – Файл KollajDraverActivity.java

```
public class KOLLAJdraveractivityActivity

private String namefavorit = "Kollaj";
RatingBar ratingBar;
Button btSubmit;

TextView  raiting;
TextView  titlefavorite;
Button favotritButton ;
Button  magazine;

Button info;
Button news;
Button action;
public void OutRaiting(){
    final DatabaseReference Root;
    Root =
 FirebaseDatabase.getInstance(). getReference().child
("Kollaj Raiting");
    Root.
addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(
@NonNull DataSnapshot
snapshot) {
            raiting.setText
(snapshot.getValue()
.toString());

        }

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

        }
    });
}
public void ChekFavorite() {
    final DatabaseReference Root;
    Root = FirebaseDatabase.getInstance().getReference()
.child("Users").child(Prevalent.UserCurrentEmail).child("Favorite");
    Root.
addValueEventListener( new ValueEventListener() {
        @Override
        public void onDataChange (
@NonNull DataSnapshot snapshot) {
            if (
snapshot.child("Коллаж").exists()) {
                Toast. makeText
(KOLLAJdraveractivityActivity. this,
```

```
 "Нашлось", Toast. LENGTH_SHORT).show();
                favotritButton. setVisibility(View.GONE);
                titlefavorite.setVisibility(View.GONE);
            }
        }
public void AddFavorit() {
    final DatabaseReference Root;
    Root= FirebaseDatabase.
getInstance().getReference().child("Users").child(Prevalent.UserCurrentEmail)
.child("Favorite");
    Root.
addListenerForSingleValueEvent
( new ValueEventListener() {
        @Override
        public void onDataChange
(@NonNull  DataSnapshot snapshot) {
            Date currentTime
 = Calendar. getInstance().getTime();
            String date = currentTime .toString();
            HashMap<String,Object> UserDataMap  =
new HashMap<>();
            UserDataMap .put
("NameFavorite","Коллаж");
            UserDataMap.put
("Date", date);
            Root.child
("Коллаж").updateChildren (UserDataMap)
```

Листинг Б.5 – Файл PassajDraverActivity.java

```
public void updateraiting (String sanmart)
{
    FirebaseDatabase db
 = FirebaseDatabase.getInstance();
    DatabaseReference ref
= db.getReference("raitingall Passaj"); // Key
    DatabaseReference ur =  db.getReference ("usercount Passaj"); // Key
    ur.addListenerForSingleValueEvent (new ValueEventListener()

        public void onDataChange
(@NonNull DataSnapshot  snapshot) {
            String user =
 snapshot.getValue(String.class);
            ref.addListenerForSingleValueEvent
(new ValueEventListener() {

                @Override
                public void onDataChange
(DataSnapshot dataSnapshot) {
    String raiting = dataSnapshot.getValue(String.class);
 float parseraiting = Float.parseFloat(raiting);
 float usercount =
 Float.parseFloat(user);//кол-во   float raitinmag =
Float.parseFloat(sanmart);// оценка из
                    usercount =
(float) (usercount + 1);//прибавляем колво    float main;
                    String newusercount =  String. valueOf( usercount);
      main = raitinmag + parseraiting;
                    String newraiting = String.valueOf(main);
       FirebaseDatabase db =
 FirebaseDatabase.getInstance();
                    DatabaseReference ref = db. getReference
```

```
("raitingall Passaj");
                    ref.setValue(newraiting );
                    DatabaseReference ur  = db. getReference("usercount
Passaj");
                    ur.setValue(newusercount ); // Value
                    float raitineg =
main/usercount;
                    String mainraiteneng =
 String.valueOf(raitineg);
                    DatabaseReference all =
db.getReference("Passaj Raiting");
                    all.setValue(mainraiteneng );
                }
            });
        }

        public void  ChekFavorite ()
 {
    final  DatabaseReference Root;
    Root =
FirebaseDatabase.getInstance().getReference().child("Users").child(Prevalent.
UserCurrentEmail). child("Favorite");
    Root.addValueEventListener
(new ValueEventListener() {
        @Override
        public void onDataChange
(@NonNull DataSnapshot  snapshot) {
            if ( snapshot.child("Пассаж").exists()) {
                favotritButton .setVisibility
(View.GONE);
            }
        }
        public void onCancelled (@NonNull DatabaseError error) {

        }
    });
}
public void AddFavorit()
 {
    final DatabaseReference Root;
    Root=
FirebaseDatabase.getInstance().getReference().child("Users").child(Prevalent.
UserCurrentEmail)
.child("Favorite");
    Root.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange
(@NonNull DataSnapshot snapshot) {
            Date currentTime =
Calendar.getInstance(). getTime();
            String date = currentTime.toString();
            HashMap<String,Object> UserDataMap =
new HashMap<>();
            UserDataMap. put
("NameFavorite","Пассаж");
            UserDataMap. Put
 ("Date", date);
            Root.child ("Пассаж").
updateChildren(UserDataMap).addOnCompleteListener(new
OnCompleteListener<Void>() {
                @Override
                public void onComplete(@NonNull Task<Void> task) {
                    if (task
```

```
. isSuccessful()){

                    }
                    else {

                    }
                }
            });
        }
```

Листинг Б.6 – Файл Navigation_Activity.java

```
public class Navigation_Activity extends
{

    private AppBarConfiguration  mAppBarConfiguration;
    private  ActivityNavigation2Binding binding;
    String Sname[] ,Sdescriptions [];
    int
image[]={R.drawable.cym,R.drawable.kollaj,R.drawable.passaj,R.drawable.riteyl
,
            R.drawable .sanmart,R.drawable. syvorovski,R. drawable.vl};
    RecyclerView recyclerView;

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

        binding =
ActivityNavigation2Binding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());
        binding. appBarNavigation. fab.setOnClickListener(new
View.OnClickListener () {
            public void onClick
(View view) {
                Snackbar. make(
view, "Replace with your own action", Snackbar.LENGTH_LONG)
                        .setAction("Action", null).show();
            }
        });
        DrawerLayout drawer =
 findViewById(R.id.drawer_ layout);
        NavigationView navigationView =
findViewById(R.id .nav_view);
        navigationView.setNavigationItemSelectedListener (this);
            FloatingActionButton fab =
 findViewById(R.id.fab);
            fab.setOnClickListener(new  View.OnClickListener() {
                @Override
                public void onClick (View  view) {
                    Paper.book(). destroy();
                    Intent logoutIntent = new Intent(Navigation_Activity
.this,MainActivity.class);
                    startActivity(logoutIntent);
                }
            });
        recyclerView =
 findViewById(R.id.recycleView);
        Sname = getResources(). getStringArray
```

```
(R.array.list_tc);
        Sdescriptions = getResources().getStringArray(R.array.descriptions);

        Adapter  adapter =
new Adapter(this,Sname,Sdescriptions,image);
        recyclerView. setAdapter(adapter);
        recyclerView. setLayoutManager (new   LinearLayoutManager( this))
;
    }
    public boolean onNavigationItemSelected
(@NonNull MenuItem item) {
        int id =
 item.getItemId();
        if
(id ==  R.id.nav_zakladki)
{
  Intent Favorite =
 new Intent(
Navigation_Activity.this, FavoriteActivity .class
);
            startActivity(Favorite);
        }
        else if
 (id == R.id .nav_history){
            Intent History =
 new Intent(Navigation_Activity .this, HistoryActivity .class);
            startActivity
(History);
        }
        else if
(id == R.id. nav_lichkabinet){
            Intent UserCabinet =
 new Intent
(Navigation_Activity.this,UserCabinet .class);
            startActivity(UserCabinet);
        }
        DrawerLayout drawerLayout =
findViewById(R. id. drawer_layout);
        drawerLayout.
closeDrawer (GravityCompat.START);
        return  false;
    }
}
```

Приложение В

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

(рекомендуемое)

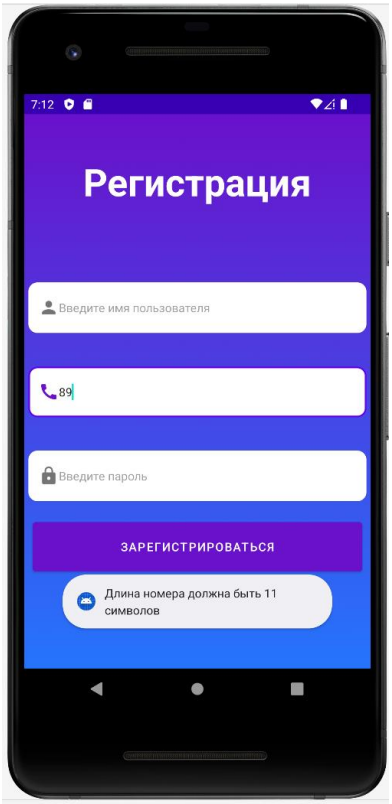


Рисунок В.1 – Тест проверки длины номера

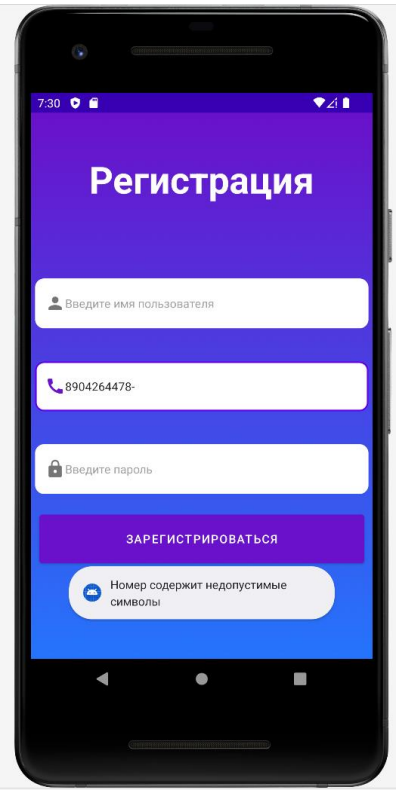


Рисунок В.2 – Тест проверки содержания недопустимых символов

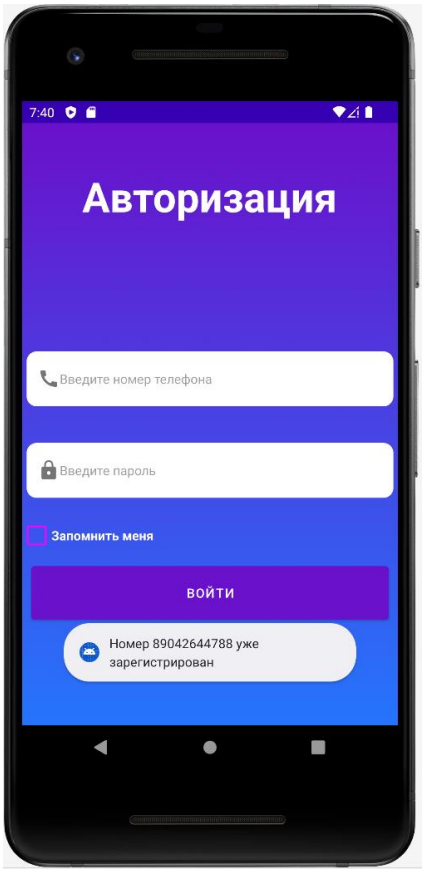


Рисунок В.3 – Тест для проверки свободного номера

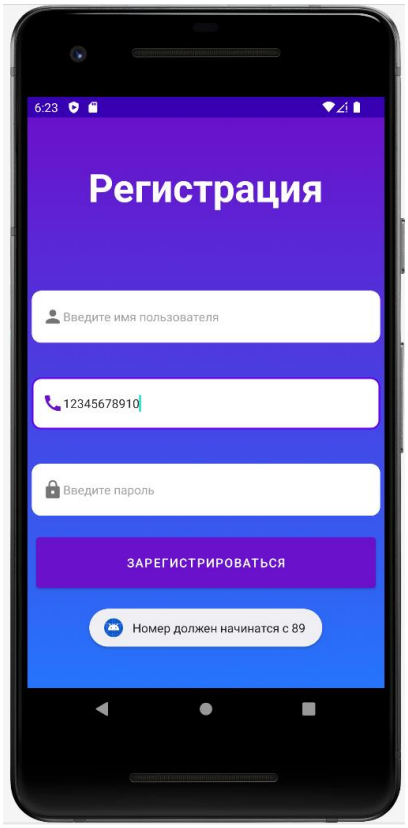


Рисунок В.4 – Тест проверки первых символов

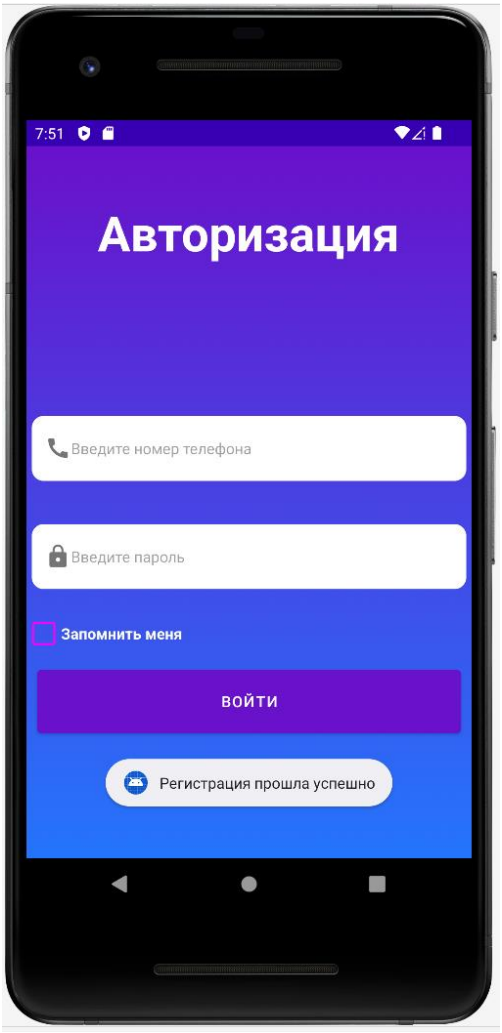


Рисунок В.5 – Тест успешной регистрации