МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
заведующий кафедрой компьютерной безопасности и прикладной алгебры ЧелГУ, к.ф.-м.н.
_____ А.Н. Ручай

“___”_____ 2019 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н., профессор
_____ Л.Б. Соколинский

“___”_____ 2019 г.

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ КЛИЕНТА СЕРВИСА СОВМЕСТНЫХ ЗАКУПОК

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.02. 2019.115-016.ВКР

Научный руководитель
к.ф.-м.н.
_____ Т.Ю. Маковецкая

Автор работы,
студент группы КЭ-405
_____ М.М. Тимеева

Ученый секретарь
(нормоконтролер)
_____ О.Н. Иванова

“___”_____ 2019 г.

Челябинск-2019

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский

09.02.2019

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-405
Тимеевой Марии Михайловне,
обучающемуся по направлению
09.03.02 «Информационные системы и технологии»

1. Тема работы (утверждена приказом ректора от «25» Марта 2019 № 899)

Разработка мобильного приложения для клиента сервиса совместных закупок

2. Срок сдачи студентом законченной работы: 05.02.2019.

3. Исходные данные к работе

3.1 Харди Б., Филлипс Б. Программирование под Android. – СПб.: Питер, 2014. – 592 с.
3.2. Браун. И. Веб-разработка с применением Node и Express. – Санкт-Петербург: Питер, 2017. – 336 с.

4. Перечень подлежащих разработке вопросов

4.1. Провести анализ предметной области.
4.2. Спроектировать и разработать архитектуру приложения.
4.3. Спроектировать и реализовать мобильное приложение.
4.4. Провести тестирование.

5. Дата выдачи задания: 09.02.2019.

Научный руководитель
доцент кафедры СП Т.Ю. Маковецкая
Задание принял к исполнению М.М. Тимеева

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

АКТУАЛЬНОСТЬ ТЕМЫ РАБОТЫ

Покупки в интернете через посредников давно перестали быть чем-то необычным и стали совершенно привычным способом приобретения товаров для большого количества пользователей. Всё больше людей, совершающих покупки, делают свой выбор в пользу более выгодных закупок из различных магазинов, объединяясь друг с другом.

В социальных сетях набирают популярность сообщества и группы, занимающиеся совместными покупками.

Но в настоящее время клиент подобных сервисов не имеет возможности просматривать информацию о своих заказах, не связываясь при этом с администратором. Поэтому нами было решено разработать мобильное приложение для клиента сервиса совместных закупок.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью данной работы является разработка мобильного приложения для клиента сервиса совместных закупок. Для достижения цели передо мной были поставлены следующие задачи.

1. Изучить предметную область.
2. Проанализировать аналоги.
3. Провести проектирование интерфейса приложения.
4. Реализовать мобильное приложение.
5. Провести тестирование приложения.

ОБЪЕМ И СТРУКТРУРА РАБОТЫ

Работа состоит из введения, четырех глав, заключения, списка литературы и приложения. Объем работы составляет 29 страниц, объем списка литературы – 15 источников, объём приложения – 2 страницы.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первой главе описан анализ предметной области. Он включает в себя обзор существующих аналогов и анализ существующих решений для создания мобильного приложения.

Вторая глава посвящена проектированию системы. В ней описаны функциональные и нефункциональные требования, представлены диаграмма вариантов использования с описанием и диаграмма последовательности, а также произведено проектирование интерфейса приложения.

Третья глава - «Реализация» содержит описание инструментов, используемых при реализации, реализации компонентов системы и их взаимодействие.

В четвертой главе «Тестирование» представлены результаты функционального тестирования мобильного приложения.

Заключение содержит основные результаты, полученные при выполнении выпускной квалификационной работы.

Приложение содержит примеры окон реализованного мобильного приложения, запущенного в эмуляторе.

АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Предметная область проекта

Совместные закупки – это возможность выгоднее приобрести понравившейся товар в популярных магазинах. Так же они позволяют делать покупки с зарубежных сайтов.

Их схема работы заключается в следующем: объединив заказы нескольких клиентов, администратор производит закупку, а после доставляет товары в город заказчиков. Клиент при этом занимается только выбором товара, его оплатой и получением.

Итоговая стоимость товара для клиента включает в себя цену данного товара в магазине закупки, накрутку посредника, стоимость доставки, рассчитываемой при получении и изменения курса валюты, в которой производился выкуп товаров.

1.2. Обзор аналогов

Поиск аналогов мобильного приложения для клиентов совместных закупок осуществлялся в двух наиболее популярных магазинах приложений: Play Market и App Store [13].

Полных аналогов найдено не было, однако существует несколько частичные аналогов. Можно выделить наиболее популярные:

«СП 2.0» – мобильное приложение, позволяющее выбрать товар непосредственно через него, отложить понравившееся в корзину и через «Сообщения» связаться с посредником. Оповещение об изменении статуса заказа происходит в формате уведомлений. Для клиента отсутствует возможность просмотра любых его прошлые заказы. Проанализировав комментарии к данному приложению, оставленные пользователями в магазинах приложений, хотелось бы отметить следующий частоупоминаемый недостаток – слишком долгий ответ посредников. Данная проблема возникает по причине отсутствия в приложении уведомлений для посредников о поступлении нового заказа. Пример интерфейса приведён на рисунке 1.

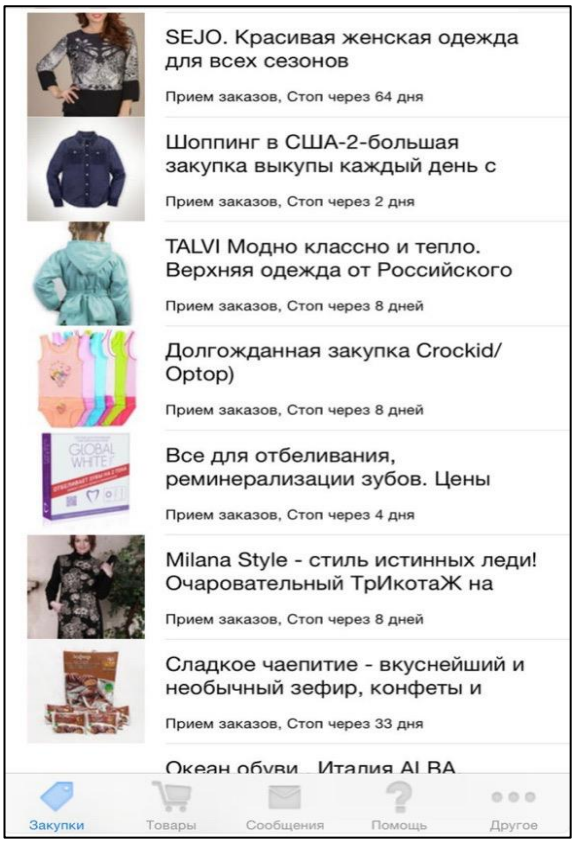

Рис. 1. Интерфейс приложения «СП 2.0»

«SuperPuper: совместные закупки» отображает все произведённые клиентом заказы, разделяя их на ожидаемые заказы, неоплаченные и новые. Для клиента совместных закупок в данном приложении видны планируемые в ближайшее время и осуществлённые ранее закупки. Во вкладке «Уведомления» содержится информация о статусе заказа. Изменить свои личные данные пользователь может после перехода во вкладку «Обо мне». Однако у приложения «SuperPuper: совместные закупки», как и у приведенного ранее аналога, можно выделить несколько достаточно крупных недостатков. Одним из них является отсутствие возможности просматривать подробную информацию о заказах. Кроме того, не предусмотрено добавление своих комментариев к заказам и добавление фотографий. Пользователи данного приложения, согласно отзывам в маркетах, отмечают в качестве недостатка неудобство отсутствия ссылок на магазин. Пример интерфейса приведен на рисунке 2.

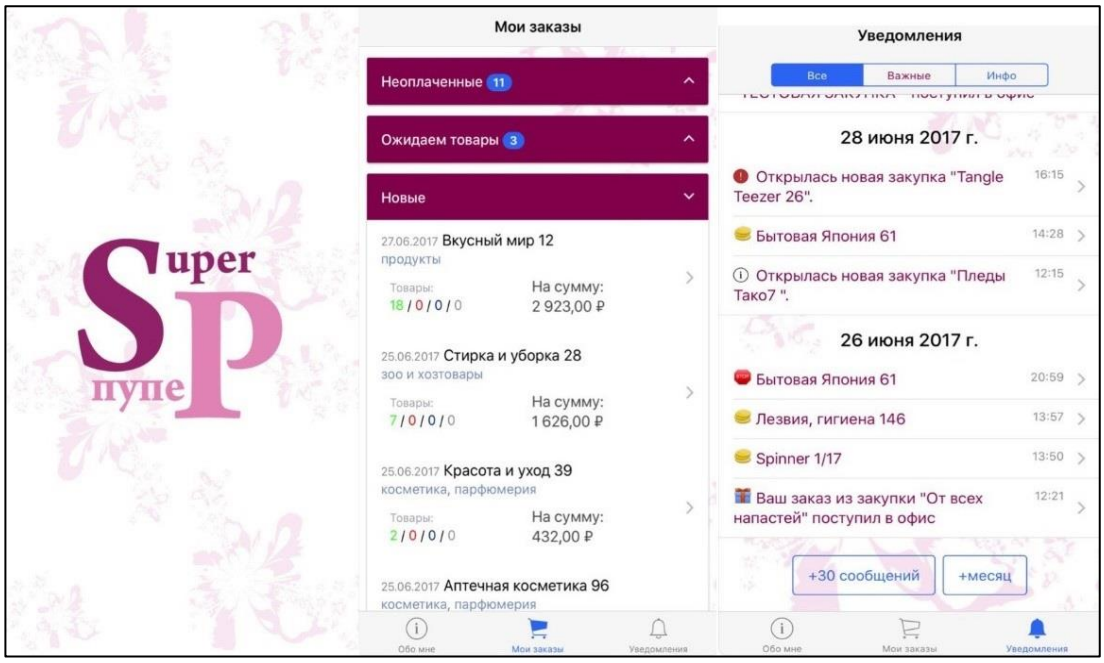

Рис. 2. Интерфейс приложения «SuperPuper: совместные закупки»

Таким образом, проведя обзор существующих аналогов мобильного приложения для клиентов сервиса совместных закупок, было решено реализовать приложение, в котором клиент мог бы просматривать подробную информацию о своих заказах, видеть их статус, а также добавлять свои комментарии с фотографиями.

1.3. Анализ существующих решений

Разработка мобильных приложений уже давно перестала быть новым направлением в сфере IT. Она настолько популярна, что нередко мобильные приложения разрабатываются намного раньше, чем сами вебсайты. И по мере того, как мобильная связь становится все более доступной, способов разработки приложений для использования в абсолютно любых обстоятельствах становится все больше.

Специалисты выделяют три вида мобильных приложений: гибридные решения, нативные и web-приложения [6].

Нативные приложения (от английского native – родной) – это такие приложения, которые были разработаны на “родном” для платформы язы-

ке. Так, для Android родным языком является Java, а для iOS – objective-C. Нативные приложения для любой конкретной платформы пишутся на тех языках программирования, которые утверждают разработчики ПО. Для загрузки приложений пользователи могут использовать так называемые магазины приложений: App Store, Google Play и другие. Такие магазины как правило установлены в программное обеспечение смартфона [7].

Мы можем отметить следующие особенности нативных приложений.

1. Им доступна аппаратная часть устройства, это означает, что такие приложения могут использовать такие функции смартфона, как камера, микрофон, определение геолокации.

2. Способны работать частично или даже совсем без интернет- соединения.

3. Каждое такое приложение полностью задействует весь аппаратный ресурс устройства, оно оптимизировано под используемую операционную систему.

Web-приложения – приложения, не требующие установки, они работают в браузере телефона. По существу, это просто мобильная версия сайта, обладающая более широким интерактивом. Реализация таких приложений происходит без привязки к какой-то конкретной платформе, а само приложение не имеет возможности использовать функции (например, камеру, микрофон и др.) смартфона.

Можно выделить следующие отличия web-приложений.

1. Аппаратная часть телефона в таких приложениях не используется.

2. Скачивание таких приложений, ориентированных для того или иного устройства из магазинов приложений не представляется возможным.

3. Web-приложение работает на любых устройствах и платформах

Гибридными мобильными приложениями называют такие кроссплатформенные приложения, которые взаимодействуют с ПО смартфона. Данный тип приложения представляют собой симбиоз некоторых преиму-

ществ как нативного, так и web-приложения. Фреймворк, выбранный разработчиком для написания приложения, определяет возможности гибридных приложений.

Можно выделить следующие особенности гибридных мобильных приложений.

1. Загружаются из магазина приложений, при этом имеют возможность независимого (автономного) обновления информации;

2. Для работы требуют подключение к интернету, так как при отсутствии интернет-соединения веб-функции не работают;

3. Позволяют компаниям объединять положительные качества как родных (нативных) приложений, так и долговечность и технологическую актуальность последних веб-технологий.

Приложения всегда разделяются между собой в зависимости от того, для какой операционной системы они были реализованы.

Ниже приведена статистика по популярности различных мобильных ОС в мире на январь 2019 г. [5].

- Android – 74,450%;
- iOS – 22,850%;
- KaiOS – 22,850%;
- WindowsPhone – 0,30%;
- Samsung – 0,280%;
- others – 0,410%.

Разработчики, выбирая операционную систему, для которой будет вестись разработка руководствуются таким важнейшим критерием, как потенциальный коммерческий успех приложения.

Ещё пару лет назад специалисты по разработке для iOS зарабатывали более чем в четыре раза больше, нежели Android разработчики того же уровня. Об этом свидетельствует получаемая статистика по монетизации приложений [14].

Однако сегодня ситуация в корне поменялась. Положение Android-разработчиков становится все стабильнее и уровень доходов от создания приложений для Android нивелируется с доходами разработчиками для iOS. Поэтому все больше специалистов выбирают разработку под операционную систему Android [10].

Проведя анализ используемых в мобильных устройствах операционных систем, разработку мобильного приложения для сервиса совместных закупок я решила проводить для операционной системы Android.

Вывод по первой главе

В данной главе был произведён анализ предметной области, включая анализ существующих аналогичных проектов и существующих решений. В результате было принято решение разрабатывать мобильное приложение для ОС Android.

2. ПРОЕКТИРОВАНИЕ

2.1. Определение требований к проектируемой системе

В ходе анализа требований к проектируемой системе были определены как функциональные, так и нефункциональные требования.

Выделим функциональные требования.

1. Мобильное приложение должно давать пользователю возможность аутентификации.
2. Мобильное приложение должно давать пользователю возможность просматривать список произведённых заказов.
3. Мобильное приложение должно давать пользователю возможность просматривать информацию о каждом заказе.
4. Мобильное приложение должно давать пользователю возможность добавлять комментарий с фотографией к выполненным заказам.
5. Мобильное приложение должно давать пользователю возможность выйти из профиля.

Выделим нефункциональные требования.

1. Мобильное приложение должно быть разработано для использования на смартфонах с операционной системой Android, начиная с версии 5.0.
2. Мобильное приложение должно корректно отображаться в книжной ориентации устройства.
3. Мобильное приложение должно быть реализовано на языке Java в интегрированной среде разработки Android Studio.

2.2. Варианты использования системы

В ходе анализа проектируемого приложения были выявлены варианты использования, представленные на Рис. 3. В системе определён один актёр, взаимодействующий с приложением. Пользователь – клиент сервиса совместных закупок, использующий приложение.

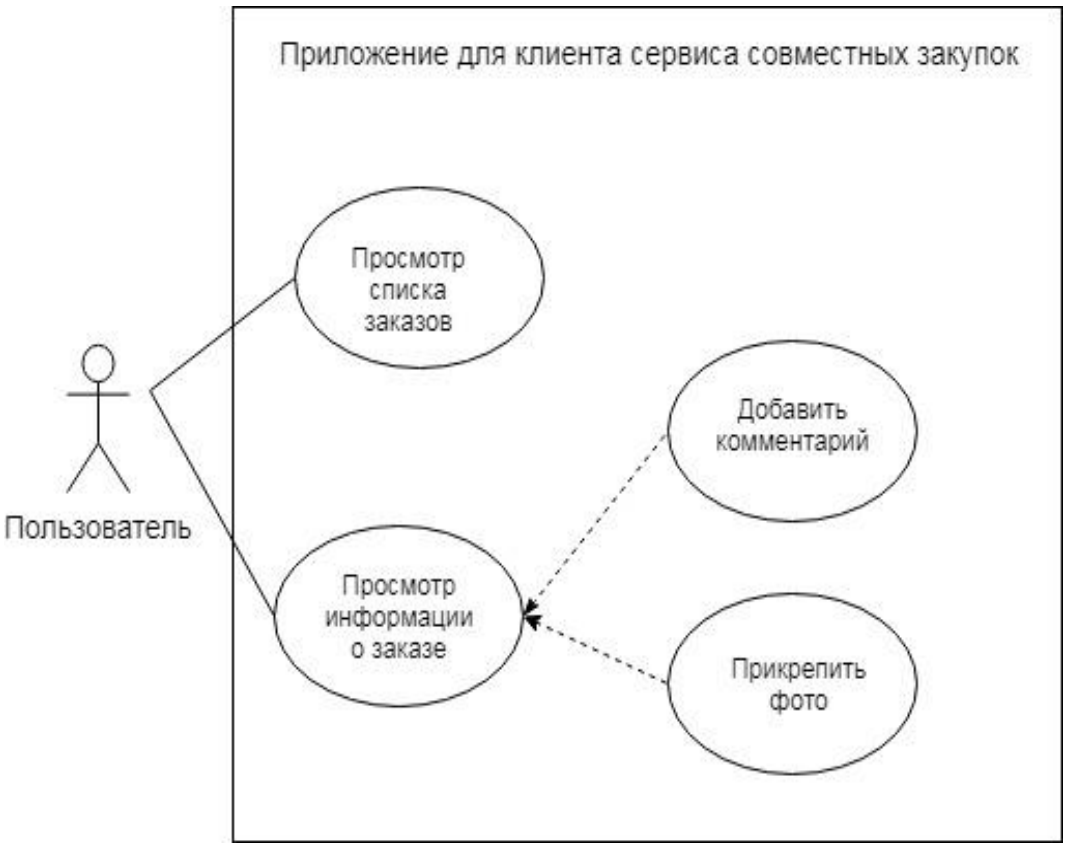

Рис. 3. Диаграмма вариантов использования

Для Пользователя возможны следующие варианты использования системы.

1. Просмотр списка заказов – пользователь имеет возможность просматривать список совершённых им заказов: формирующихся, выкупленных и завершенных.

2. Просмотр информации о заказе – пользователь имеет возможность просматривать подробную информацию о каждом заказе, включая информацию по оплате.

3. Добавить комментарий – пользователь может добавлять комментарий к завершённому заказу. Является расширением варианта использования «Просмотр информации о заказе».

4. Прикрепить фото – пользователь может прикрепить фото к завершённому заказу. Является расширением варианта использования «Просмотр информации о заказе».

2.3. Проектирование интерфейса

Дизайн приложения производился в графическом редакторе Adobe Illustrator cc 2019. Один из этапов разработки дизайна приложения – это создание макета экранов.

Форма 1 «Авторизация» - стартовый экран при первом запуске приложения. Содержит формы для логина и пароля. Представлена на рисунке 4.

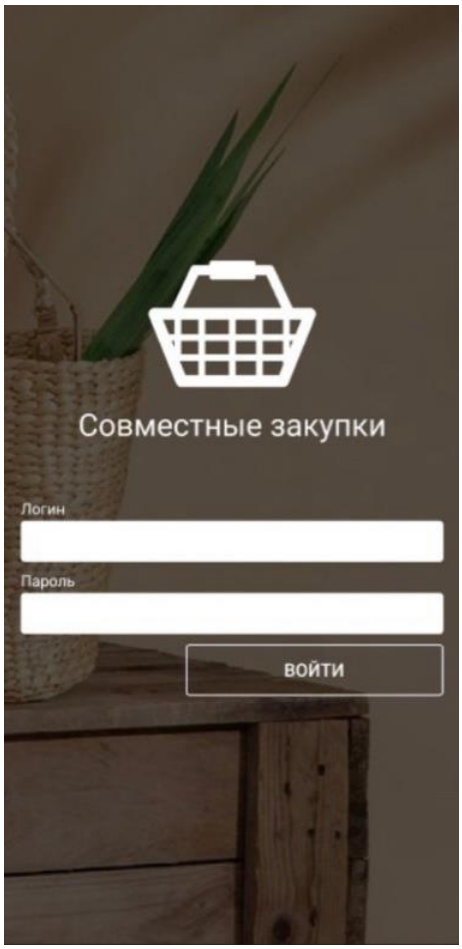

Рис. 4. Форма «Авторизация»

Форма 2 «Мои заказы» - экран, отображающий все формирующиеся, выкупленные и завершённые заказы клиента. Видны кликабельные название товара, магазин и дата заказа. Изображена на рисунке 5.

Форма 3 «Заказ». Отображает информацию по конкретному заказу, совершённому клиентом: размер, дату заказа, статус, ссылку на магазин. Так же в данной форме предполагается отображение подробной информа-

ции об оплате, которая включается в себя аванс, курс валюты, в которой производится выкуп товара, доплату за доставку, определение стоимости и оплата которой производится клиентом непосредственно при получении заказа. Представлена на рисунке 6.

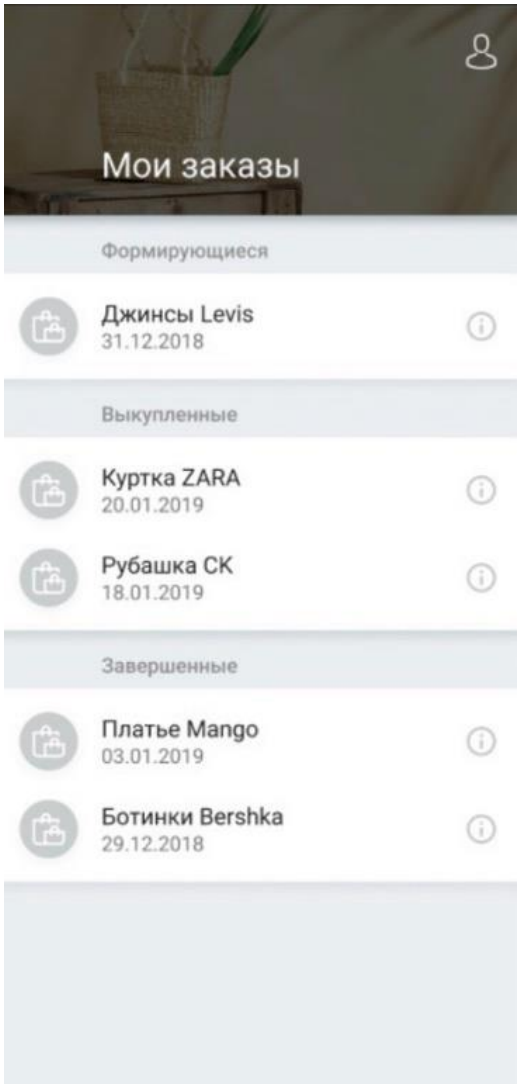

Рис. 5. Форма «Мои заказы»

Форма 4 «Комментарий». Включает в себя форму для добавления комментария клиентом к его завершённому заказу. Для загрузки фотографии предусмотрена кнопка «Прикрепить фото». Для редактирования уже созданного ранее комментария пользователь может стереть в поле комментария прошлый отзыв, написать новый и воспользоваться кнопкой «Изменить фото» для смены фотографии. Представлена на рисунке 7.

Рис. 6. Форма «Заказ»

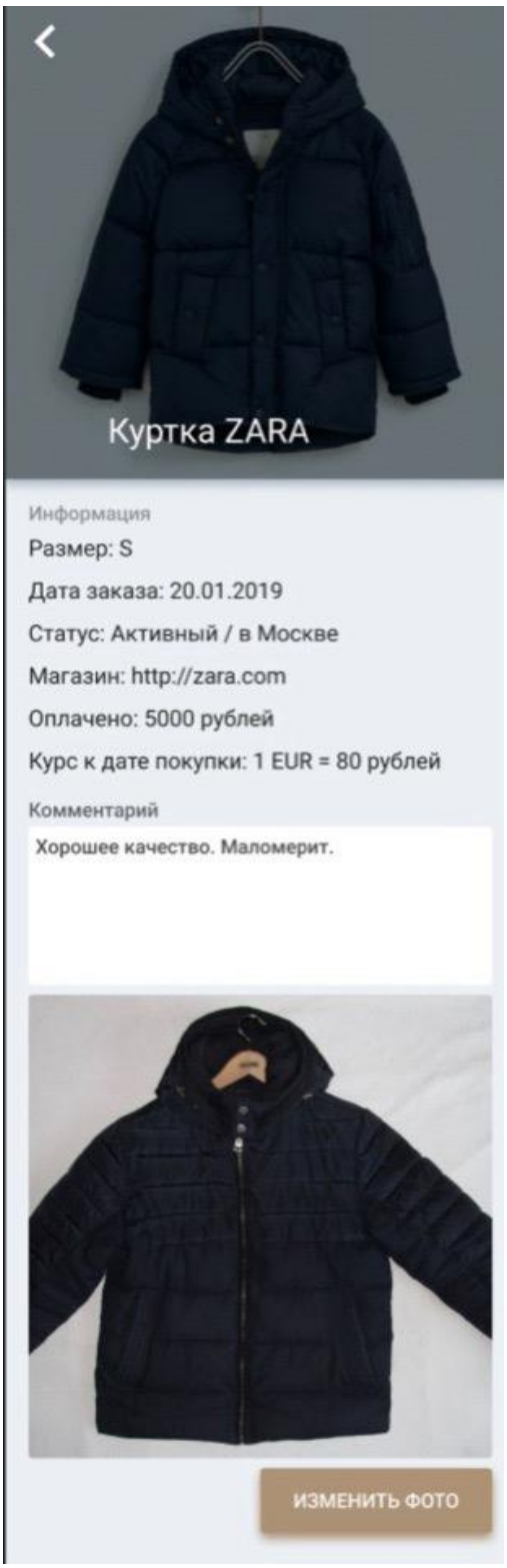

Рис. 7. Формы «Комментарий»

2.4. Диаграмма последовательности

Для проектирования мобильного приложения был использован язык UML.

Проектируемое мобильное приложение имеет несколько форм, взаимодействие которых отображено на диаграмме последовательности вызова на рисунке 7.

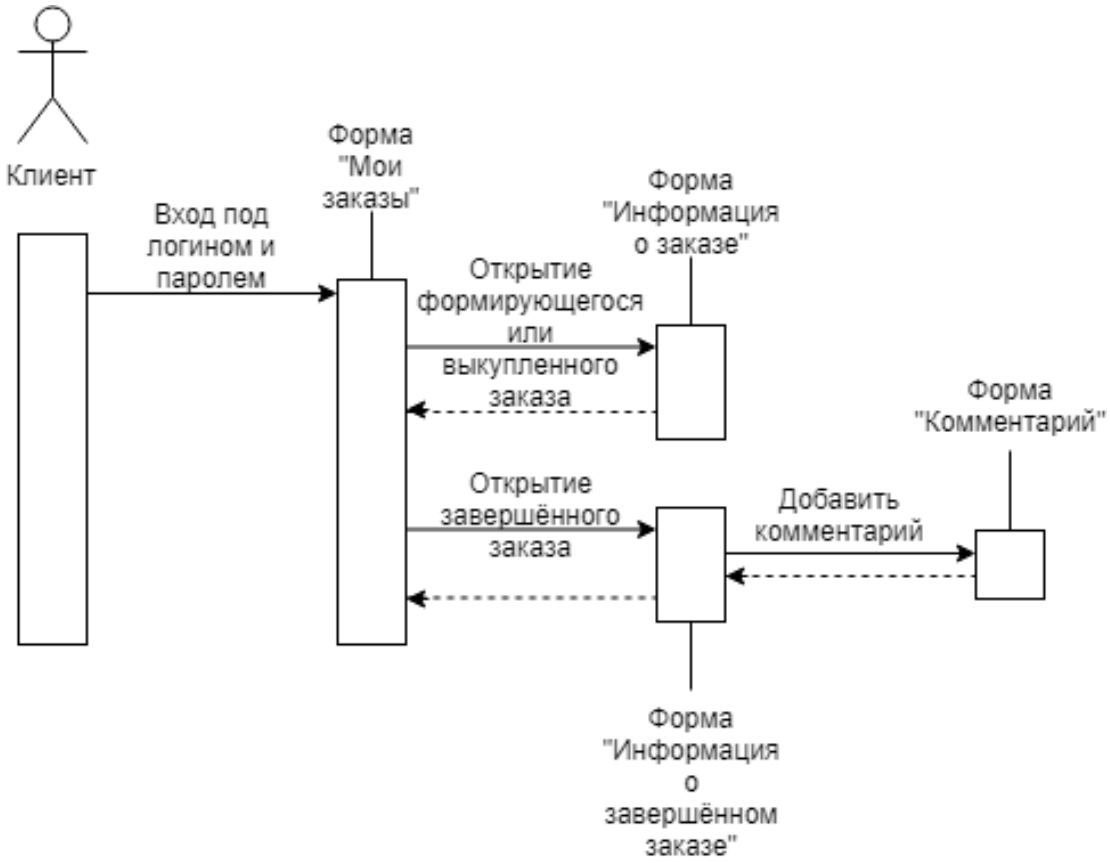

Рис. 7. Диаграмма последовательности вызова

Вывод по второй главе

Сформированы функциональные и нефункциональные требования к проектируемой системе, описаны диаграмма вариантов использования и диаграмма последовательности. Спроектирован интерфейс всех форм разрабатываемого приложения.

3. РЕАЛИЗАЦИЯ

3.1. Инструменты, используемые при реализации

Для реализации приложения на платформе Android был выбран язык программирования Java и интегрированная среда разработки Android Studio.

Java – объектно-ориентированный язык программирования. Существует уже более 22 лет. За это время приобрел большую популярность среди программистов. Он был изобретен компанией Sun Microsystems, но позднее OpenSource перекупила у них права на язык Java. Основными преимуществами Java является его простота, надежность, безопасность, удобство, производительность, кроссплатформенность [8].

Android Studio часто выбирается программистами, так как она является официальной средой разработки для работы с платформой Android. Принадлежит компании Google. Является популярной средой разработки благодаря полной интегрированности, простоте и большого количества готовых шаблонов [9].

В качестве стандарта обмена данными между сервером и клиентским приложением использована библиотека JSON. Для работы с REST API выбрана библиотека Retrofit. Удобство загрузки и кэширования фотографий обеспечивает библиотека Picasso.

Серверная часть модуля реализована на языке программирования TypeScript и программной платформе Node.js.

TypeScript – язык программирования, разработанный корпорацией Microsoft в 2012 году. Обратно совместим с JavaScript [3]. Часто используется совместно с серверной платформой Node.js [11].

В качестве веб-фреймворка для запуска сервера и создания программного интерфейса (API) был выбран Express. Он предоставляет достаточно возможностей для запуска и работы, не требуя много времени на подготовку. Для работы с базой данных: сопоставление таблиц и их отно-

шений с классами было решено использовать ORM библиотеку Sequalize [4] .

Для хранения данных о пользователях, заказах и закупках была использована СУБД PostgreSQL [2].

3.2. Взаимодействие компонентов

Разрабатываемое мобильное приложение является частью общей системы, которая так же содержит сервер, базу данных и десктопное приложение для работы администратора, который занимается заполнением данных о заказах, закупках и клиентах. Десктопное приложение, схема базы данных и SQL запрос к базе данных приложения, возвращающий баланс, отсортированный по виду закупки или дате, были реализованы параллельно в рамках выпускной квалификационной работы другого студента. Схема базы данных представлена на рисунке 8.

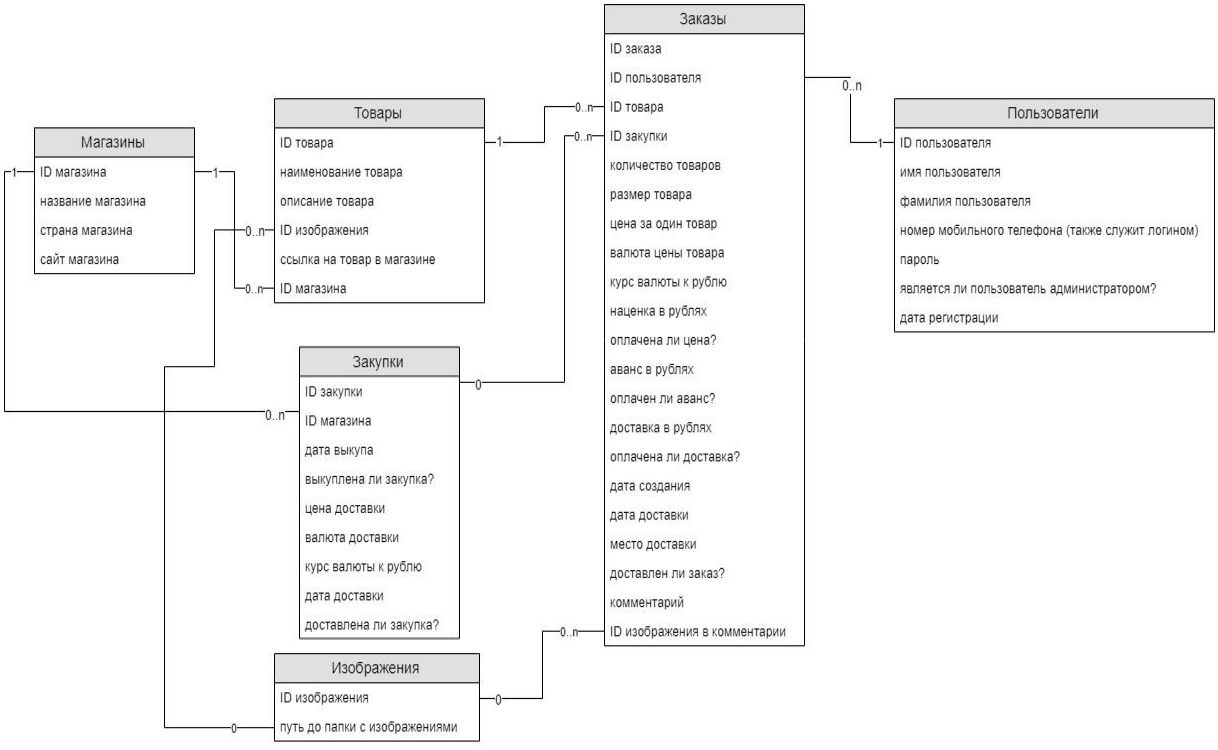

Рис. 8. Схема базы данных

Мобильное приложение взаимодействует с сервером по средствам Restful API [12].

Сервер отправляет SQL-запросы к базе данных и получает ответ. В качестве сервиса для хранения файлов (изображений) было решено ис-

пользовать Amazon S3. Сервер расположен на облачной PaaS-платформе Heroku [1].

На рисунке 9 представлена диаграмма компонентов разрабатываемой системы.

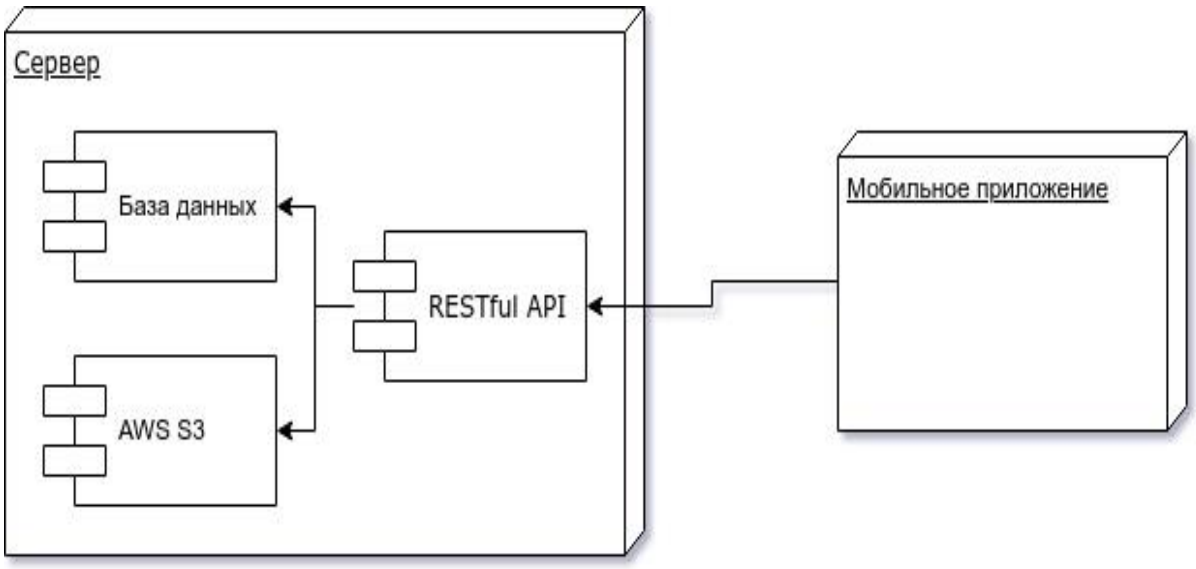

Рис. 9. Диаграмма компонентов

3.3. Реализация компонентов системы

Реализация некоторых основных функций приложения представлена в листингах.

В листинге 1 представлен скрипт для запуска сервера. Создаётся экземпляр класса Web и запускается сервер. Настраивается обработчик прерываний – сигнал. Когда сигнал будет передан процессу, процесс остановится операционной системой.

В листинге 2 показаны обработчики входящих запросов. При их реализации выдерживался паттерн проектирования «цепочка обязанностей». Согласно данному паттерну входящий запрос проходит через все функции-обработчики по цепочке. Каждая такая функция может как передать управление следующей функции, так и завершить цепочку, отправив ответ или бросив исключение [15].

Листинг 1. Запуск сервера

```
import "../global";
import "./_logger";
import {debug, Sequelize, Web} from "src";
Sequelize.instantiate();
Web.instantiate();
Web.instance.listen().then((address) => {
debug(`Web listening on %o`, address);
});
async function handle(signal: string): Promise<void> {
await Web.instance.close();
await Sequelize.instance.close();
debug(`Web was stopped by ${signal} signal`);
}
process.on("SIGINT", handle);
process.on("SIGTERM", handle)
```

Листинг 2. Обработчики запроса

```
import Router from "express-promise-router";
import {Const} from "src";
import api from "./api";
const router = Router();
export default router;
router.use(Const.API_MOUNT_POINT, api);
```

Использование библиотеки для сетевого взаимодействия retrofit помогает упростить процесс создания собственного клиента и осуществления запросов к API. Реализация их генерируется в рантайме. Однако есть необходимость описания интерфейса, которое представлено на листинге 3.

Листинг 3. Интерфейс для генерирования REST-клиента

```
interface SovmZakupApi {
        @FormUrlEncoded
        @POST("/api/v0/signin")
        Call<SignInResponse> signIn(@Field("phone") String phone,
@Field("password") String password);
        @GET("/api/v0/user")
        Call<DataResponse<User>> currentUser(@Header("Authorization")
String token);
        @GET("/api/v0/orders")
```

```
            Call<DataResponse<List<Order>>>                          allOr-
ders(@Header("Authorization") String token, @Query("userId") String userId,
@Query("offset") long offset, @Query("limit") long limit);
            @GET("/api/v0/orders/{id}")
            Call<DataResponse<Order>>      order(@Header("Authorization")
String token, @Path("id") String id);
            @FormUrlEncoded
            @PATCH("/api/v0/orders/{id}")
            Call<DataResponse<Order>>                            commentOr-
der(@Header("Authorization")   String   token,   @Path("id")   String   id,
@Field("comment") String comment);
            @PUT("/api/v0/orders/{id}/image")
            Call<DataResponse<Order>>                                up-
loadImage(@Header("Authorization")  String  token,  @Path("id")  String  id,
@Body RequestBody body);
    }
```

Для работы с базой данных было решено применить технологию программирования ORM (англ. Object-Relational Mapping) и библиотеку Sequelize. Она обеспечивает взаимодействие с моделями и SQL-запросами. Пример реализации запроса к базе данных для создания нового заказа приведён на листинге 4.

Листинг 4. Запрос для создания нового заказа

```
const order = await Order.create({
      advance: req.body.advance,
      amount: req.body.amount,
      delivery: req.body.delivery,
      markup: req.body.markup,
      productId: req.body.productId,
      purchaseId: req.body.purchaseId || null,
      userId: req.body.userId,
    });
```

Вывод по третьей главе

Описаны инструменты, используемые при реализации. Приведены диаграмма взаимодействия компонентов и листинги с примерами реализации основных функций.

4. ТЕСТИРОВАНИЕ

Для тестирования системы было применено функциональное тестирование.

Функциональное тестирование – это тестирование программного обеспечения в целях проверки реализуемости функциональных требований, то есть способности программного обеспечения в определенных условиях решать задачи, нужные пользователям.

Функциональные требования определяют, что именно делает программное обеспечение, какие задачи оно решает.

Ниже в таблице представлены примеры тестирование системы: название теста, шаги, необходимые для проверки, ожидаемый результат и его подтверждение или опровержение.

В результате проведения тестирования программа показала корректную работу. Все задачи решены верно.

Табл. Тестирование

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
1	Запуск приложения	Запустить мобильное приложение на устройстве с ОС Android версии 5.0 или выше	Открытие формы для авторизации при первичном запуске	Да
2	Аутентификация	Ввести логин и пароль, созданные администратором в десктопном приложении	Открытие формы со списком заказа пользователя	Да
№	Название теста	Шаги	Ожидаемый результат	Тест пройден?

Продолжение табл.

3	Просмотр информации о заказе	Нажать на заказ в форме «Мои заказы»	Открытие формы, содержащей информацию о заказе: Цену, Дату и Место доставки и закупки, Товар, Магазин. Сверху отображается изображение товара и его наименование.	Да
4	Добавление комментария	Открыть завершённый заказ. Внизу заполнить поле «Комментарий». Нажать на кнопку «Сохранить комментарий»	Сохранение комментария у завершённого заказа	Да
5	Добавление фотографии	Под полем «Комментарий» в завершённом заказе нажать на кнопку «Прикрепить фото». Выбрать готовую фотографию из галереи или сделать новую с помощью камеры на мобильном устройстве. Сохранить.	Добавление фотографии к завершённому заказу	Да

Вывод по четвертой главе

Произведено функциональное тестирование мобильного приложения для клиента сервиса совместных закупок.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы было реализовано мобильное приложение для клиента сервиса совместных закупок для ОС Android. Объём кода составил: физических срок – 4343, строк исходного кода – 3622, строк с комментариями –182. Скриншоты примеров окон реализованного мобильного приложения, запущенного в эмуляторе Android Studio, приведены в приложении.

Были решены следующие задачи.

1. Изучена предметная область.
2. Проанализированы аналоги.
3. Проведено проектирование интерфейса приложения.
4. Реализовано мобильное приложение.
5. Проведено тестирование приложения.

ЛИТЕРАТУРА

1. Cloud Application Platform | Heroku. [Электронный ресурс] URL: https://www.heroku.com (дата обращения: 12.04.2019).

2. PostgreSQL. [Электронный ресурс] URL: https://www.postgresql.org (дата обращения: 12.04.2019).

3. Rozentals N. Mastering TypeScript - Build enterprise-ready, industrial strength web applications using TypeScript and leading JavaScript Frameworks. – Packt Publishing, 2015. – 364 p.

4. Sequelize. [Электронный ресурс] URL: http://docs.sequelizejs.com (дата обращения: 12.04.2019).

5. Statcounter. [Электронный ресурс] URL: http://gs.statcounter.com/os-market-share/mobile/worldwide (дата обращения: 06.02.2019).

6. StudRef. [Электронный ресурс] URL: https://studref.com/463511/informatika/tipy_mobilnyh_prilozheniy (дата обращения: 05.02.2019).

7. Архитектура мобильного клиент-серверного приложения. [Электронный ресурс] URL: https://habr.com/post/246877/ (дата обращения: 07.03.2019).

8. Берд. Java для чайников. Барри Берд. – М.: Диалектика / Вильямс, 2013. – 521 с.

9. Варакин М.В. Разработка мобильных приложений под Android. М.: УЦ «Специалист», 2012. – 128 с.

10. Дэрси Лорен , Кондер Шейн Android за 24 часа. Программирование приложений под операционную систему Google. М.: Рид Групп, 2011. – 464 с.

11. Кантелон М. Node.js в действии. / М. Кантелон. – М.: Питер, 2015. – 810 .

12. Когаловский М. Энциклопедия технологий баз данных. – М.: Финансы и статистика, 2002. – 800 с.

13. Обзор магазинов мобильных приложений. [Электронный ресурс] URL: https://yamobi.ru/posts/obzor_magazinov_mobilnyih_prilojeniy.html (дата обращения: 01.02.2019).

14. Общие сведения о платформе Android. [Электронный ресурс] URL: https://developer.android.com/guide/index.html (дата обращения: 29.03.2019).

15. Паттерн Chain of Responsibility (цепочка обязанностей). [Электронный ресурс] URL: http://cpp-reference.ru/patterns/behavioral-patterns/chain-of-responsibility (дата обращения: 16.04.2019).

ПРИЛОЖЕНИЕ

На рисунках 1-3 представлены скриншоты нескольких окон реализованного мобильного приложения для клиента сервиса совместных закупок.

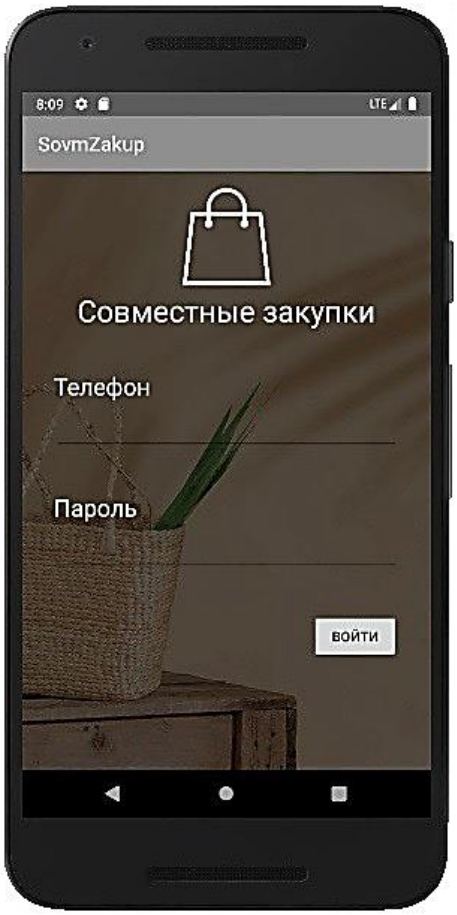

Рис. 1. Окно аутентификации

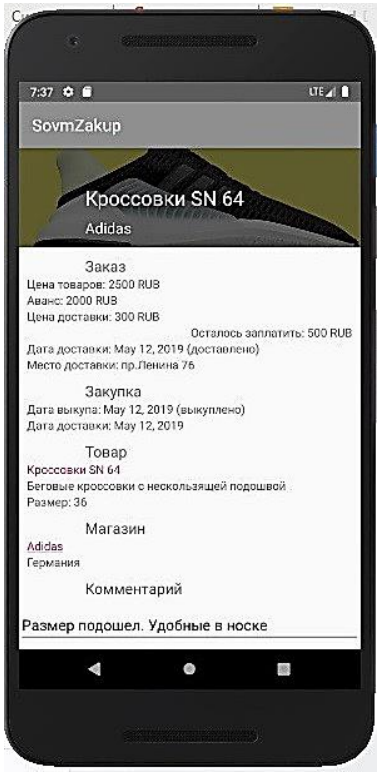

Рис. 2. Окно просмотра информации о заказе

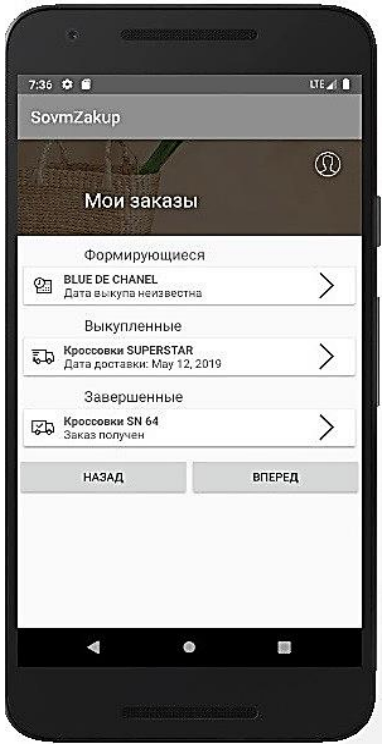

Рис. 3. Окно «Мои заказы»