

СЕКЦИЯ «ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И МАТЕМАТИЧЕСКИЕ МОДЕЛИ В ЭКОНОМИКЕ»

УДК 004.42:005.932.2

1. ПРОГРАММНОЕ СРЕДСТВО УПРАВЛЕНИЯ СИСТЕМОЙ МНОГОПРОДУКТОВЫХ ПОСТАВОК ПРЕДПРИЯТИЯ

Коренько Д.Д., студент гр. 073602

*Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь*

Сторожев Д.А. – ст. преподаватель каф. ЭИ

Аннотация. Актуальность разработки программного средства управления системой многопродуктовых поставок предприятия обусловлена не только повышением сложности и объема поставок, но и необходимостью оперативного реагирования на изменения внешней среды и рыночных условий. Программное средство данного типа представляет собой интегрированную систему, способную эффективно управлять всеми этапами поставочной цепи, начиная от планирования и закупок и заканчивая доставкой готовой продукции потребителю.

Ключевые слова. Управление поставками, программное средство, поставки предприятия, бизнес-процесс, база данных, серверная разработка, Bootstrap, Html, технологии Spring Boot, язык программирования Java, архитектурные решения.

В современной динамичной бизнес-среде, где логистика стала неотъемлемой частью успешного функционирования предприятий, существует необходимость в эффективном управлении процессом поставок. Несмотря на значительные достижения в автоматизации, многие организации продолжают сталкиваться с вызовами, связанными с ручным управлением многопродуктовыми поставками.

Системы управления поставками в наше время должны быть не только автоматизированными, но и гибкими, способными адаптироваться к быстро меняющимся условиям рынка. Одним из основных вызовов, с которым сталкиваются предприятия, является сложность координации и оптимизации поставок различных продуктов в разнообразные места назначения. Этот процесс часто требует высокой степени внимания к деталям, что делает его подверженным человеческим ошибкам и задержкам.

Подчеркивается также, что в условиях растущей конкуренции предприятиям необходимо не только предоставлять клиентам высококачественные товары, но и обеспечивать их своевременной и эффективной доставкой. Это требует не только оптимизации внутренних логистических процессов, но и активного внедрения инновационных технологий, таких как системы отслеживания поставок и аналитические инструменты для прогнозирования потребительского спроса [1].

Таким образом, целью является проектирование и эксплуатация программного средства, предназначенного для улучшения управления системой многопродуктовых поставок предприятия, с целью повышения его конкурентоспособности и эффективности бизнес-процессов. Основной упор делается на повышение конкурентоспособности и оптимизацию эффективности бизнес-процессов. Данное программное средство представляет собой инновационный инструмент, направленный на совершенствование логистических операций, управление запасами, обработку заказов и мониторинг поставок.

Объектом исследования являются многопродуктовые поставки предприятия, охватывая не только сам процесс поставок, но и комплекс логистических процессов, тесно связанных с управлением запасами, обработкой заказов и системой мониторинга поставок. В контексте исследования рассматриваются все этапы жизненного цикла товаров от момента формирования заказа до его полной поставки конечному потребителю.

В рамках фреймворка для разработки web-приложения был выбран Spring.

Spring - представляет собой широко используемый фреймворк для разработки приложений на языке Java. Он обеспечивает обширный набор инструментов, сокращающих сложность создания разнообразных приложений, таких как веб-приложения, микросервисы, приложения для обработки данных и многие другие.

Навигация происходит путем нажатия на иконки, которые отображены на экране. Разработка пользовательского интерфейса потребовала также создание более 20 различных макетов. Таким образом, была проведена спецификация пользовательского интерфейса и определение основных назначений для навигации пользователя. Была сформирована дизайн-система и макеты интерфейса.

Для создание понятного и удобного пользовательского интерфейса использовались такие компоненты как Bootstrap и Thymeleaf.

Bootstrap - представляет собой широко используемый инструмент для разработки современных веб-интерфейсов и страниц. Этот фреймворк обеспечивает доступ к разнообразным готовым компонентам, стилям и JavaScript-плагином, что способствует созданию визуально привлекательных, отзывчивых и стильных веб-сайтов.

Основные преимущества этого фреймворка включают в себя адаптивность, кросс-браузерность, простоту использования, быстрое освоение, чистый и понятный код, а также единообразие стилей.

Thymeleaf представляет собой мощный инструмент для динамического создания веб-страниц и электронных писем на платформе Java. Этот инструмент позволяет интегрировать HTML-код с выражениями Thymeleaf, которые вычисляются во время выполнения приложения. Thymeleaf обладает богатым набором функциональных возможностей, включая поддержку международных сообщений и обработку форм. Он также может легко интегрироваться с различными фреймворками, включая Spring Framework и Spring Boot. Более того, Thymeleaf доступен для использования бесплатно и распространяется как открытое программное обеспечение.

В соответствии с дизайн-стандартами пользовательский интерфейс представляет собой набор средств, обеспечивающих взаимодействие пользователя с программой[5]. Ключевыми критериями для успешного пользовательского интерфейса являются:

- Адаптивность (способность легко внедрять новый функционал);
- Понятность (исключение использования сложных формулировок, особенно в меню);
- Интуитивность (способность предоставить пользователю понятный и легкий в использовании интерфейс).

Для системы управления многопродуктовыми поставками экран должен быть простым и лаконичными. Должна быть возможность войти в систему для дальнейшей работы и пройти регистрацию. Для этого кнопки входа и регистрации размещаются в правом верхнем углу. Главная страница представлена на рисунке 1.

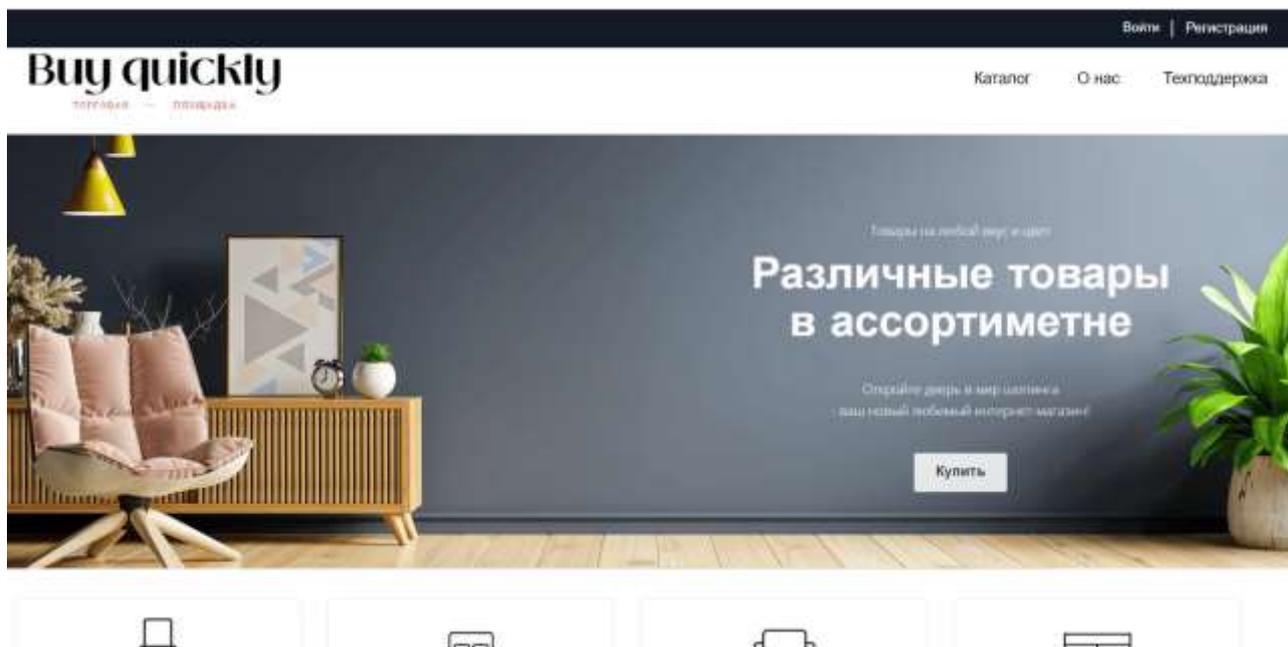


Рисунок 1 – Главная страница

Диаграмма компонентов в контексте разработки программного средства управления системой многопродуктовых поставок предприятия играет ключевую роль в определении архитектуры системы. Эта диаграмма помогает установить взаимосвязи и зависимости между различными программными компонентами, необходимыми для успешной реализации функциональности системы.

Основные компоненты, которые являются неотъемлемой частью разрабатываемого программного средства управления многопродуктовыми поставками на предприятии. Важно отметить, что каждый из этих компонентов выполняет определенную функцию и взаимодействует с другими компонентами для обеспечения работоспособности системы в целом. Диаграмма компонентов представлена на рисунке 2.

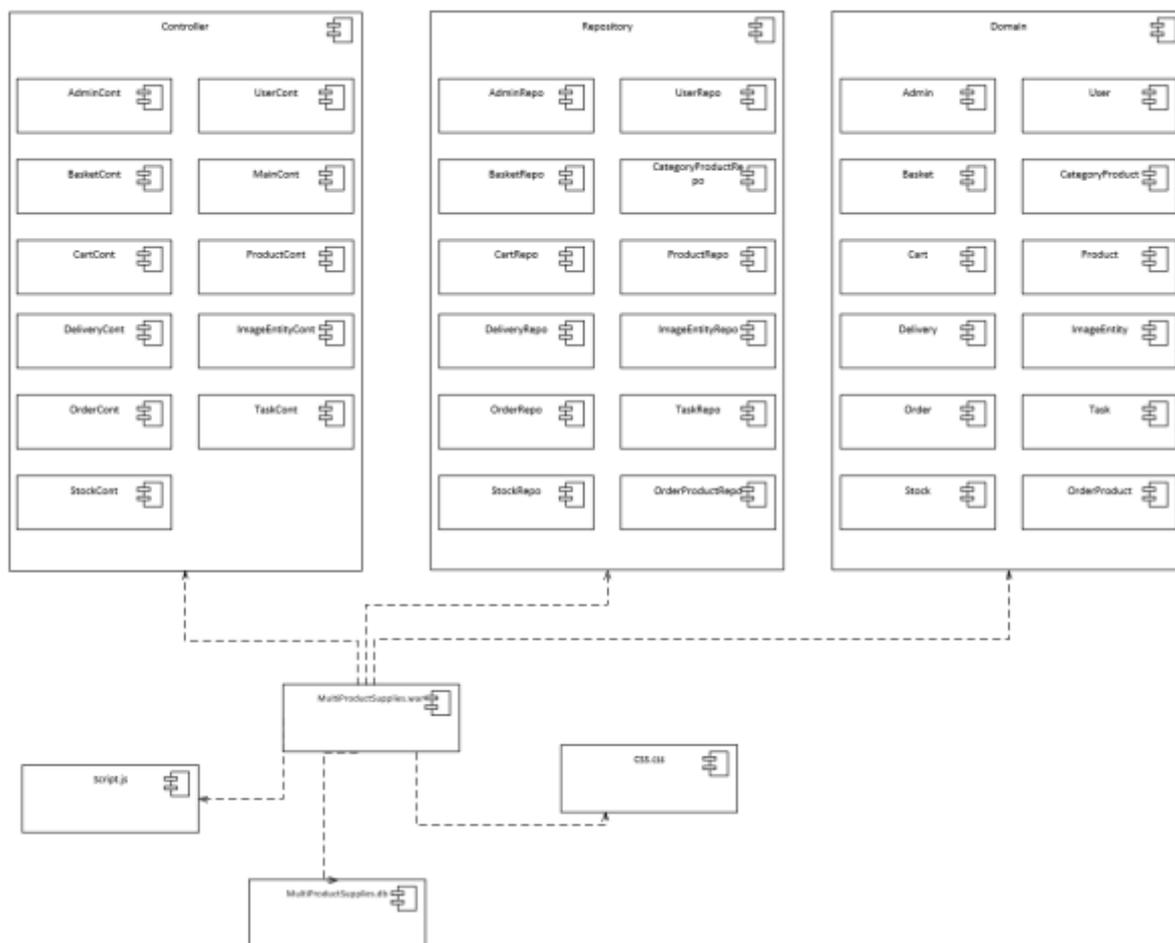


Рисунок 2 – Диаграмма компонентов

В процессе разработки особенно важно гарантировать безопасность данных. Фреймворк Spring Security предоставляет набор методов и инструментов для обеспечения безопасного доступа к ресурсам. Одним из ключевых компонентов является SecurityContextHolder, который служит для хранения информации о текущем контексте безопасности в приложении. В данном контексте содержится подробная информация о пользователе (принципале), взаимодействующем с приложением.

Spring Security использует объект Authentication для представления авторизованной сессии пользователя. Для обеспечения безопасности паролей рекомендуется применять BCryptPasswordEncoder для их хеширования перед сохранением в базе данных. BCryptPasswordEncoder гарантирует безопасное хеширование паролей, включая использование соли, что делает их устойчивыми к взлому.

Для определения правил доступа к различным URL-адресам и ресурсам приложения, полезны методы antMatchers() и hasRole(). Эти методы предоставляют возможность указать разрешения на доступ к определенным URL-адресам или ресурсам только для аутентифицированных пользователей или тех, у кого есть определенные роли.

Механизм авторизации пользователей в приложении можно реализовать с помощью аннотаций @PreAuthorize и @PostAuthorize. Они позволяют определить правила авторизации на уровне методов или классов сервисов и контроллеров. Например, с помощью аннотации @PreAuthorize("hasRole('ADMIN')") можно указать, что доступ к определенному методу или ресурсу имеют только пользователи с ролью "ADMIN". Для защиты от SQL-инъекций рекомендуется использовать параметризованные запросы или именованные параметры в SQL-запросах. Фреймворки JPA и Hibernate предоставляют механизмы автоматического экранирования пользовательских данных при выполнении SQL-запросов [2].

Регулярное создание резервных копий базы данных является неотъемлемой частью обеспечения безопасности данных. Для базы данных PostgreSQL можно использовать инструменты резервного копирования, такие как pg_dump, для создания резервных копий. Рекомендуется хранить резервные копии на отдельном сервере или в облачном хранилище.

Подводя итог, была реализована парадигма (поведенческая модель) работы приложения и определены стандарты, которым впоследствии будет соответствовать программный код приложения для оптимизации скорости разработки, стабильности, и производительности системы.

Ниже изображена диаграмма вариантов использования конечной системы. Диаграмма вариантов использования – диаграмма, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей [3].

На диаграммах использования представленные на рисунках 3, 4 были выбраны актеры в виде поставщика(администратора) и покупателя.

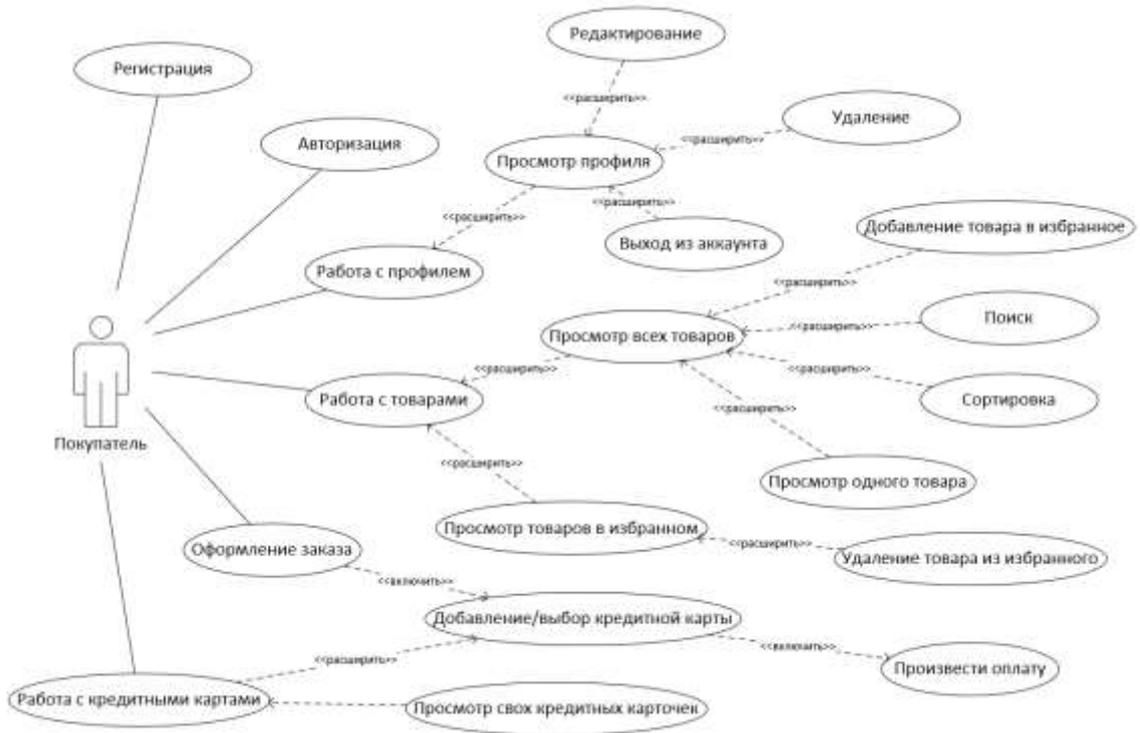


Рисунок 3 – Диаграмма вариантов использования для покупателя

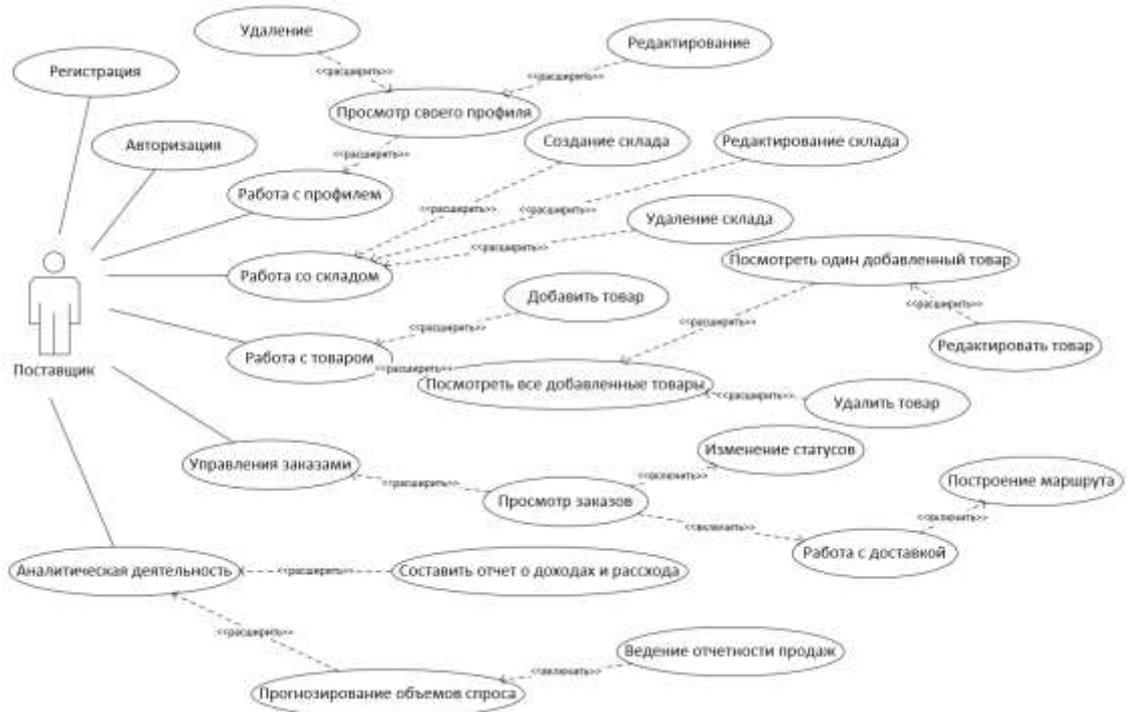


Рисунок 4 – Диаграмма вариантов использования для поставщика

Даталогическая модель базы данных формируется на основе инфологической модели, учитывая особенности конкретной системы управления базами данных (СУБД). Эта модель, также известная как схема данных в контексте реляционных баз данных, включает в себя определение структуры таблиц, указание типов данных и описания связей между таблицами. Даталогическая диаграмма представлена на рисунке 5.

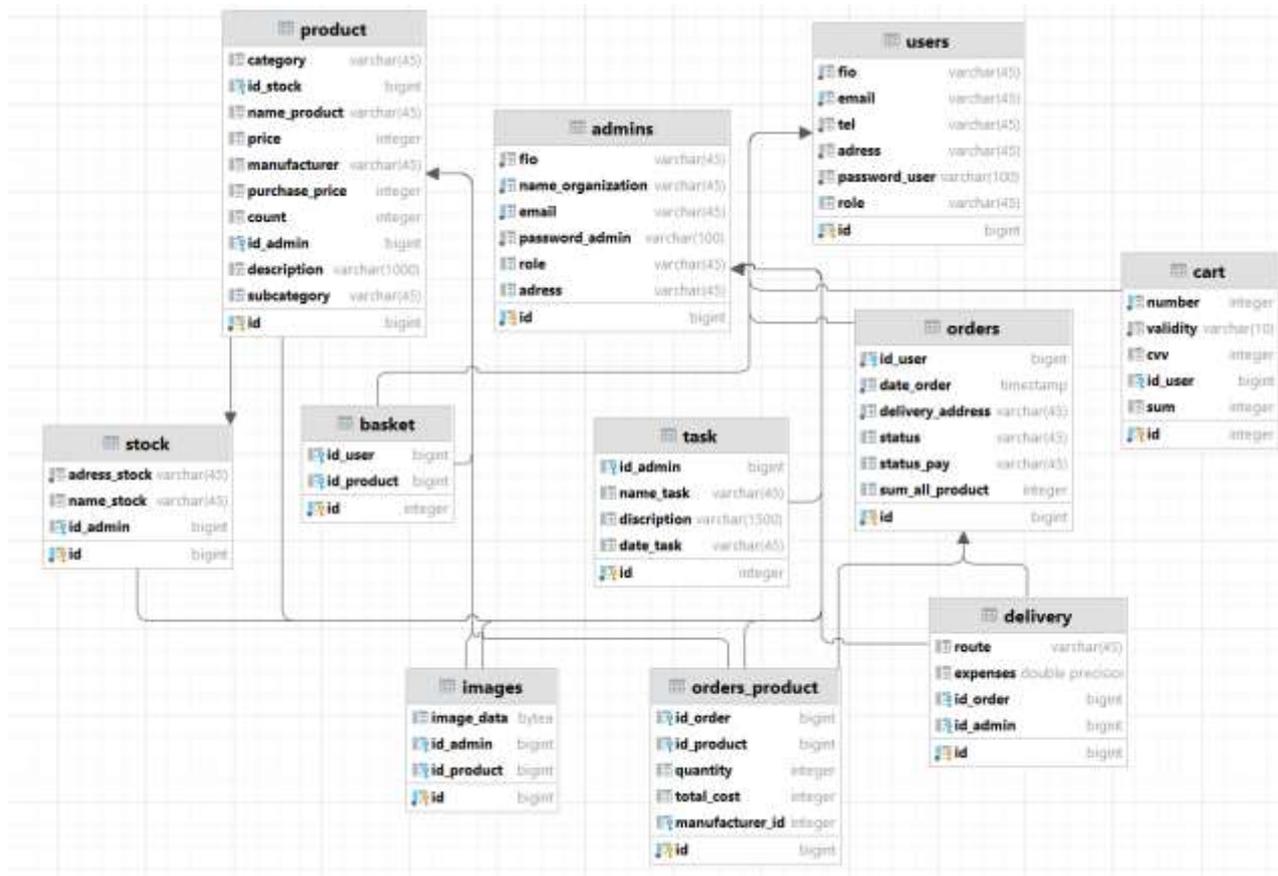


Рисунок 5 – Даталогическая модель базы данных

Каждая из этих сущностей содержит данные, например admins, хранит данные о поставщике и имеет роль "ADMIN", а таблица users содержит в себе данные о покупателях, так же покупатели имеют возможность привязывать кредитную карту, данные о картах хранятся в сущности cart, собственно так же добавленные товары в избранное хранятся в таблице basket.

Сущности product, images, stock, task напрямую связаны с поставщиком, они реализованы для хранения данных о продуктах, и их изображения, складах, а также запланированных задачах.

Сущности delivery, orders, orders_product, хранят данные, которые помогают организовывать отправку и оформление заказа, такие как, информация о доставке, данные о заказе, и данные о товарах, которые входят в заказ покупателя.

Подводя итог, была реализована парадигма (поведенческая модель) работы приложения и определены стандарты, которым впоследствии будет соответствовать программный код приложения для оптимизации скорости разработки, стабильности и производительности системы.

На рисунке 6 представлена диаграмма последовательности процесса оформления и контроля хода заказа. В процессе участвуют 3 объекта: приложение, система обработки и база данных, а также 2 актёра: покупатель и поставщик.

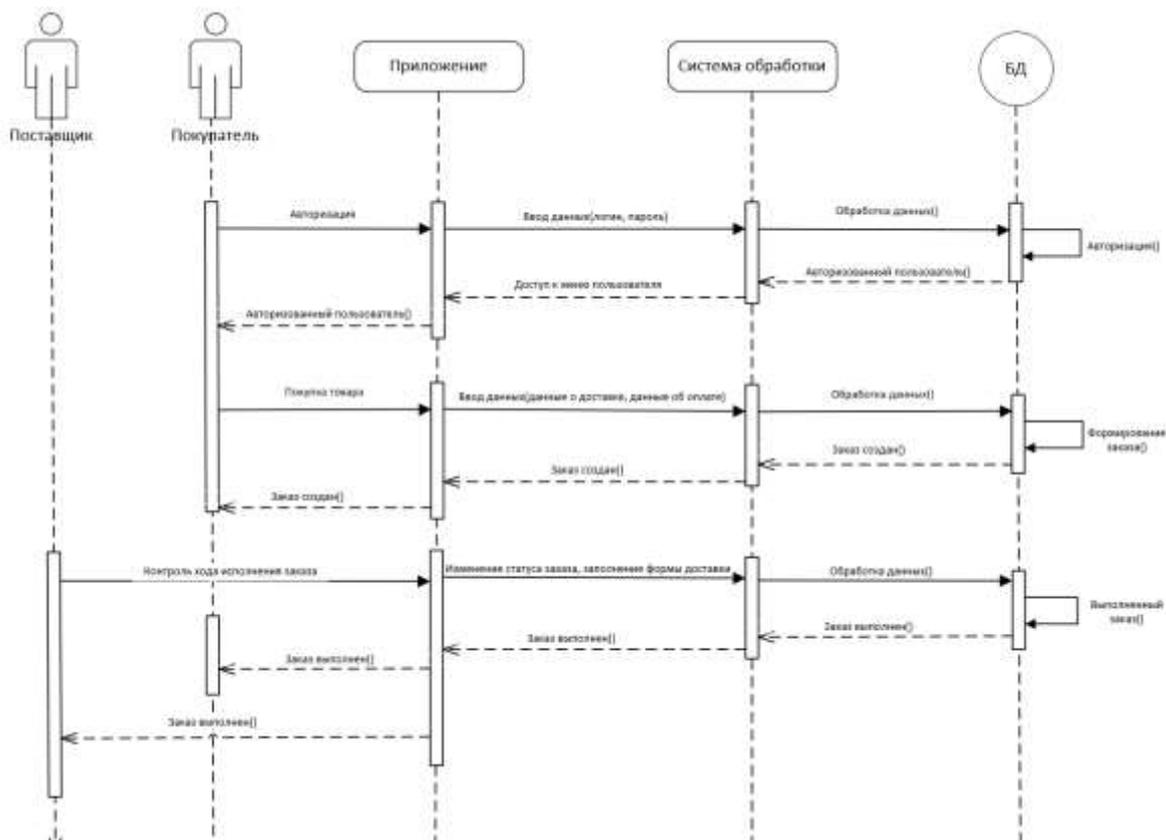


Рисунок 6 – Диаграмма последовательности модуля оплаты

Для начала покупатель заполняет свои личные данные (логин и пароль), затем эти данные передаются серверу. На следующем этапе сервер отправляет запрос на проверку в базу данных, если в базе данных такой пользователь находится, то происходит авторизация. На следующем этапе пользователь приобретает товар, он вводит персональные данные, и оплачивает товары, после чего, информация отправляется на сервер, выполняется запрос на оформление заказа, и запись о заказе с соответствующими данными добавляется в базу данных. База данных отправляет серверу результат операции добавления, а пользователь получает сообщение об успешном добавлении объявления. Дальше же подключается поставщик, который, собственно, и контролирует весь ход выполнения заказа, он заполняет определенные для формы доставки, а также изменяет статус заказа, после чего, информация отправляется на сервер, выполняется запрос на отправку заказа, и запись о заказе с измененными данными обновляется в базе данных.

Проведение тестирования ключевых компонентов приложения осуществлялось как автоматизированным, так и ручным способом с целью обеспечения надлежащего функционирования и соответствия всем требованиям. В рамках этого процесса был осуществлен полный охват реализованного функционала, включая его соответствие как функциональным, техническим и бизнес-требованиям, так и аспектам пользовательского интерфейса приложения. Автоматизированное тестирование позволило провести масштабное тестирование приложения на различных уровнях, включая функциональные и нагрузочные тесты, что позволило выявить потенциальные проблемы и дефекты в работе приложения. Ручное тестирование же позволило проверить интерактивность приложения, а также удостовериться в его удобстве использования для конечных пользователей. Такой подход обеспечил высокий уровень качества приложения и его соответствие всем требованиям, что является важным аспектом успешного внедрения и эксплуатации продукта.

Список использованных источников:

1. *Инновационные технологии в логистике и управлении цепями поставок [Электронный ресурс] – Режим доступа: <https://publications.hse.ru/pubs/share/direct/437257024.pdf>, свободный – Дата доступа: 23.03.2024.*
2. *Spring Framework [Электронный ресурс]. – Режим доступа: <https://spring.io/projects/spring-framework#learn>, свободный – Дата доступа: 18.03.2024.*
3. *Основы диаграммы вариантов использования [Электронный ресурс]. – Режим доступа: <https://uml.gitbook.io/uml-guide-book/use-case-diagrams>, свободный – Дата доступа: 18.03.2024.*
4. *Логистика: интеграция и оптимизация логистических бизнес-процессов в цепях поставок [Электронный ресурс] – Режим доступа: <https://publications.hse.ru/books/81055127/>, свободный – Дата доступа: 27.03.2024.*