Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет инженерно-экономический

Кафедра экономической информатики

Дисциплина «Программирование сетевых приложений»

|  |  |
| --- | --- |
|  | «К ЗАЩИТЕ ДОПУСТИТЬ» |
|  | Руководитель курсового проекта  старший преподаватель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | \_\_\_.\_\_\_\_.2022 |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту

на тему:

**«РАЗРАБОТКА СИСТЕМЫ ОЦЕНКИ УРОВНЯ КАЧЕСТВА ПРОДУКЦИИ»**

БГУИР КП 1-40 01 02-08 013 ПЗ

|  |  |
| --- | --- |
|  | Выполнил студент группы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |
|  | Курсовой проект представлен на проверку 09.12.2022  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |

Минск 2022

**РЕФЕРАТ**

БГУИР КР 1-40 01 02-08 013 ПЗ

**Круглей, А.А.** Программное средство системы оценки уровня качества продукции / А.А. Круглей. – Минск: БГУИР, 2022. – 60 с.

Пояснительная записка 60 с., 41 рис., 0 табл., 8 источников, 3 приложения

информационная система, ОЦЕНКА КАЧЕСТВА, автоматизация рабочего месТА, ПРОДУКЦИЯ, модели *UML*, *IDEF*0, *Java*, база данных, система

*Цель* *курсового проекта*: проектирование и разработка программного средства оценки уровня качества продукции с возможностью генерации предложений для его повышения.

*Методология проведения работы*: в процессе решения поставленных задач использованы принципы системного подхода, аналитические методы, методы компьютерной обработки экспериментальных данных и компьютерного моделирования.

*Результаты работы*: выполнена постановка задачи и определены основные методы ее решения; в ходе объектного моделирования системы построен ряд *UML*-диаграмм; разработана информационная модель системы, представленная в виде схемы базы данных; описаны основные алгоритмы работы веб-сервиса; разработано руководство пользователя; выполнено тестирование системы, показавшее ее соответствие функциональным требованиям, поставленным в задании на разработку.

Программный продукт разработан на языке *Java* и СУБД *MySQL Server*.

*Область применения результатов*: в результате выполнения данного курсового проекта было разработано программное приложение, позволяющие оптимизировать работу организаций. В нем предусмотрена система администрирования, а также проверка учетных данных при авторизации пользователей.

Приложение удовлетворяет основным характеристикам, которые были заявлены. Оно удобно в эксплуатации, целостно, конкретизировано в рамках заданной предметной области.

СОДЕРЖАНИЕ

[Введение 9](#_Toc121483144)

[1 Анализ и моделирование предметной области программного средства 10](#_Toc121483145)

[1.1 Описание предметной области 10](#_Toc121483146)

[1.2 Разработка функциональной модели предметной области 13](#_Toc121483147)

[1.3 Анализ требований к разрабатываемому программному средству. Спецификация функциональных требований 15](#_Toc121483148)

[1.4 Разработка информационной модели предметной области 16](#_Toc121483149)

[1.5 Модели представления программного средства и их описание 18](#_Toc121483150)

[2 Проектирование и конструирование программного средства 21](#_Toc121483151)

[2.1 Постановка задачи 21](#_Toc121483152)

[2.2 Архитектурные решения 21](#_Toc121483153)

[2.3 Описание алгоритмов, реализующих бизнес-логику разрабатываемого программного средства 24](#_Toc121483154)

[2.4 Проектирование пользовательского интерфейса 27](#_Toc121483155)

[2.5 Обоснование выбора компонентов и технологий для реализации программного средства 29](#_Toc121483156)

[3 Тестирование и проверка работоспособности программного средства 31](#_Toc121483157)

[4 Руководство по развертыванию и использованию программного средства 34](#_Toc121483158)

[4.1 Запуск приложения 34](#_Toc121483159)

[4.2 Описание формы авторизации 34](#_Toc121483160)

[4.3 Меню пользователя 36](#_Toc121483161)

[4.4 Меню администратора 45](#_Toc121483162)

[Заключение 55](#_Toc121483163)

[Список использованных источников 56](#_Toc121483164)

[Приложение А](#_Toc121483165)[(обязательное) 57](#_Toc121483166)

[Приложение Б](#_Toc121483168) [(обязательное) 58](#_Toc121483169)

[Приложение В](#_Toc121483171)[(обязательное) 59](#_Toc121483172)

[Ведомость документов курсового проекта 60](#_Toc121483174)

# Перечень условных обозначений, символов и терминов

|  |  |
| --- | --- |
| БД (база данных) | – представленная в объективной форме совокупность самостоятельных материалов, систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины |
| Информационная система | – система, предназначенная для хранения, поиска и обработки информации, и соответствующие организационные ресурсы (человеческие, технические, финансовые), которые обеспечивают и распространяют информацию |
| Среда выполнения | – вычислительное окружение, необходимое для выполнения компьютерной программы и доступное во время выполнения компьютерной программы |
| СУБД (система управления базами данных) | – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных |
| *API* (*application programming interface*) | – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой |
| *HTTP* (*HyperText Transfer Protocol*) | – протокол прикладного уровня передачи данных изначально – в виде гипертекстовых документов в формате *html*, в настоящий момент используется для передачи произвольных данных. |
| *IDE* (*Integrated development environment*) | – комплекс программных средств, используемый программистами для разработки программного обеспечения |
| *IDEF* | – методология функционального моделирования (англ. *function modeling*) и графическая нотация, предназначенная для формализации и описания бизнес-процессов |
| *Java* | – строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией *Sun* *Microsystems* |
| *MySQL Server* | – свободная реляционная система управления базами данных |
| *Sybase ASE* (*Adaptive Server Enterprise*) | – реляционная система управления базами данных компании *SAP*, одна из СУБД, использующая в качестве основного процедурного *SQL*-расширения язык *Transact-SQL* |
| *UML* (*Unified Modeling Language*) | – язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур |
|  |  |

# Введение

Главным фактором прогресса является непрерывное улучшение. Улучшение качества – важное условие повышение уровня жизни человека, что является целью многих стратегических программ на мировом уровне.

Каждый день в мире выпускаются много новых продуктов в самых различных сферах: от сельскохозяйственной продукции, до мобильных приложений и компьютерных игр. Сейчас практически любой товар или услуга находятся в условиях жесткой конкуренции. Если зайти в современный гипермаркет или открыть страницу популярного интернет-магазина, можно часами изучать витрины или огромные каталоги разнообразной продукции.

В таких условиях на выбор потребителя влияют даже самые незначительные факторы. А при одинаковых главных характеристиках они являются решающими. Современная система ГОСТов и опыт предыдущих проектов позволяет производителям предполагать будущие результаты. Однако именно потребители оценивают уровень качества продукции. Сбор информации о мнении клиентов и ее анализ является важнейшим шагом для улучшения своего продукта производителем. Современные технологии позволяют сильно облегчить и ускорить этот трудоемкий процесс.

Актуальность создания программы для оценки уровня качества продукции обусловлена сегодня ростом количества продуктов на рынке и необходимостью быстрой реакции производителя на мнение потребителя.

Целью данной курсового проекта является проектирование и разработка программного средства оценки уровня качества продукции с возможностью генерации предложений для его повышения.

Поставленная цель потребовала решение следующих задач:

– рассмотреть предметную область и разработать модель, отражающую бизнес-процесс на предприятии.

– провести конкурентный анализ;

– найти и проверить на целостность наборы данных уже зарекомендовавших себя методик или существующих приложений;

– описать постановку задачи на разработку программного средства;

– разработать и описать алгоритм работы программы;

– выполнить реализацию программного средства;

– разработать и описать руководство пользователя. Объектом исследования является процесс учета оценки качества продукции.

Предметом исследования является опрос для учета оценки качества продукции.

Ключевые слова: ПРОГРАММНОЕ СРЕДСТВО, АЛГОРИТМЫ, ПОЛЬЗОВАТЕЛЬ, ОЦЕНКА КАЧЕСТВА, КЛИЕНТ-СЕРВЕР, ПРОДУКЦИЯ.

Курсовой проект выполнен самостоятельно, проверен в системе «Антиплагиат». Процент оригинальности составляет 94,11%. Скриншот приведен в приложении А.

# Анализ и моделирование предметной области программного средства

## 1.1 Описание предметной области

Современная рыночная экономика предъявляет все больше требований к качеству выпускаемой продукции. Это связано с тем, что сейчас положение фирмы на рынке зависит от уровня конкурентоспособности.

Конкурентоспособность связана с десятком факторов, главные из которых: уровень цен и качество продукции.

Качество – это совокупность свойств и характеристик продукции, которые придают ей способность удовлетворять предполагаемые потребности. Обеспечение сохранения качества товаров происходит за счет соблюдения установленных мер. К подобным мерам относятся правильное производство, а также условия транспортировки. Качество продукции зависит также и от :

– качества сырья и материалов;

– уровня технического оборудования;

– организации производства и труда;

– квалификации сотрудников;

–интенсивность и продолжительность используемых производственных процессов.

В настоящее время управление качеством исходит из того, что деятельность по управлению качеством не может быть полезной после того, как продукция произведена. Эта деятельность осуществляется в ходе производства.

Для любой продукции характерны собственные свойства, которые в полной мере могут описать качество данной продукции. Для управления качеством выпускаемой продукции необходимо правильно и объективно е оценивать. В стандартах представлены общие критерии оценки качества. В них описаны наиболее эффективные показатели качества любых видов изделий.

Оценка качества товара – это определение соответствия товара общественным потребностям[3].

Контроль качества представляет собой проверку соответствия показателей качества конкретного товара требованиям, установленными стандартами, техническими условиями.

Оценка уровня качества – это комплекс мероприятий, которые включают выбор номенклатуры показателей качества продукции, установление значений указанных показателей и сравнение их с базовыми значениями, находящимися в стандартах.

Оценка качества любой продукции выполняется по следующей схеме:

– точное определение цели оценки;

– установление необходимых показателей;

– определение методов оценки;

– определение норм и требований к показателям;

– сопоставление реальных показателей качества с нормами.

Для оценки качества продукции используется система показателей, которая разделяется на следующие группы: обобщающие, комплексные и единичные. Обобщающими называются показатели, которые характеризуют общий уровень качества продукции: сорт, экономический эффект, затраты, связанные с улучшением качества. Единичными называются показатели, которые характеризуют одно из свойств продукции(назначение, надежность, эффективность и т.д.). Комплексными называются показатели, которые характеризуют несколько свойств продукции( вкл. Затраты на разработку, эксплуатацию).

На данный момент существует множество возможностей для оценки уровня качества. Одна из возможностей – опрос клиентов. Он используется в 90% исследований.

Опрос – это метод сбора первичной маркетинговой информации, предусматривающий письменное или устное обращение исследователей к определенной совокупности людей, с вопросами, содержание которые отражает изучаемую проблему. Опрос является одним из самых распространенных методов получения информации о респондентах.

Цель опроса состоит в том, чтобы выяснить субъективные мнения людей, их убеждения, предпочтения[2].

Опрос заключается в задавании людям специальных вопросов, ответы на которые позволяют исследователю получить необходимые сведения в зависимости от задач исследования. К особенностям опроса можно причислить его массовость, что вызвано спецификой задач, которые им решаются.

Преимущества опросов:

– простота проведения;

– ответы достоверны, так как количество ответов ограничено;

– использование вопросов множественного выбора позволяет устранить расхождение в результатах;

– кодирование данных довольно простое.

Недостатки опросов:

–респонденты не всегда хотят(не могут) дать необходимую информацию;

– респонденты не всегда готовы отвечать о личных мотивах, чувствах;

– необходимо научиться правильно формулировать вопросы.

Методы опроса делятся на следующие две группы: управляемые интервьюером и управляемые респондентом. К первым из них относятся опросы, которые проходят лично( на улице, на дому и т.д.), и телефонные опросы. К второй группе относятся почтовые опросы, самозаполнение анкет, интернет-опросы, опросы с помощью СМС и терминалов.

В зависимости от типа респондентов различают массовые и специализированные опросы. Массовыми называются опросы, в которых основным источником информации выступают люди, чья профессиональная деятельность не связана с предметом анализа. Специализированными называются опросы, в которых главным источником информации являются компетентные лица, чья профессиональная деятельность напрямую, или косвенно связана с предметом анализа[1].

В зависимости от частоты различают разовые и повторные опросы. Повторные опросы помогают выявить основные тенденции изменения запросов и потребностей клиентов с течением времени для принятия решений о деятельности на рынке.

В то же время опрос успешно применяется рядом наук для решения исследовательских задач. Так, социологи давно и успешно используют опрос для сбора данных о структуре населения, трудовых ресурсах, бюджетах потребления, структуре семей и о многих других сферах жизнедеятельности общества.

Опросы также разделяют на стандартизированные, частично стандартизированные и нестандартизированные. Стандартизированные опросы можно рассматривать как строгие опросы, дающие прежде всего общее представление об исследуемой проблеме. Не стандартизированные опросы менее строгие в сравнении со стандартизированными, в них отсутствуют жёсткие рамки. Они позволяют варьировать поведение исследователя в зависимости от реакции респондентов на вопросы. Частично стандартизированные опросы осуществляются при помощи «путеводителя» с перечнем строго необходимых вопросов, так и возможных вопросов. Основные вопросы должны быть заданы каждому опрашиваемому, дополнительные вопросы задаются в зависимости от ответов опрашиваемого на основные вопросы.

В данной программе предусмотрено создание стандартизированных опросов.

Правильный анализ собранных данных позволит компаниям увеличить свои доходы, поэтому стало необходимостью создание программного приложения с удобным интерфейсом, способного упростить получение мнений клиентов о продукте.

Программы-аналоги, которые уже представлены на рынке:

1. Rate&Goods. Доступно для всех платформ. Является бесплатной. Требует авторизации, но это можно сделать в один клик, если использовать аккаунт Facebook. Для получения отзывов о товаре необходимо навести камеру смартфона на штрих-код любого товара.
2. Cackle Reviews – система сбора отзывов с авторизацией через социальные сети. Интегрируется с Яндекс.Маркет. Для того, чтобы оставить какую-либо оценку продукции, необходимо пройти опрос с небольшим количеством вопросов и поставить оценкку.
3. Mneniya.Pro – система сбора отзывов. Сервис позволяет автоматически собирать отзывы от клиентов с помощью сообщений на Email, которые высылаются автоматически. Сервис также собирает мнения клиентов со сторонних сайтов.

У всех этих программ есть значительные недостатки: довольно завышенная стоимость, необходимость подключения к Интернету, интерфейс, с которым довольно сложно разобраться интуитивно. И все эти недостатки отсутствуют в разработанной в данном курсовом проекте программе.

## 1.2 Разработка функциональной модели предметной области

IDEF0 — методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов. В IDEF0 рассматриваются логические отношения между работами. Описание выглядит как «чёрный ящик» с входами, выходами, управлением и механизмом, который постепенно детализируется до необходимого уровня[5].

На рисунке 1.1 представлена контекстная диаграмма верхнего уровня с функциональной моделью «Оценка качества продукции», а также определены потоки входных и выходных данных, механизмы ограничения и управления данными.

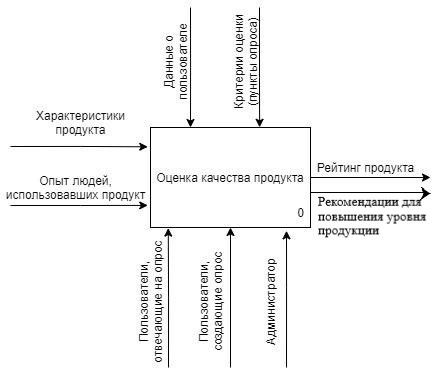


Рисунок 1.1 – Контекстная диаграмма «Оценка качества продукции»

Входной поток включает в себя характеристику продукта и опыт людей, использовавших продукт. После соответствующей обработки в выводном потоке имеем обработанные записи, содержащие информацию о товаре. В роли управляющих воздействий выступают критерии, по которым можно оценить продукт, и личные данные о пользователях, проводящих оценку. Механизмами являются создающие опросы пользователи, пользователи, проходящие опросы и администрация.

На рисунке 1.2 отображена декомпозиция контекстной диаграммы, состоящая из трех блоков: «Добавление опроса», «Прохождение опроса пользователями», «Получение анализа результатов опроса».

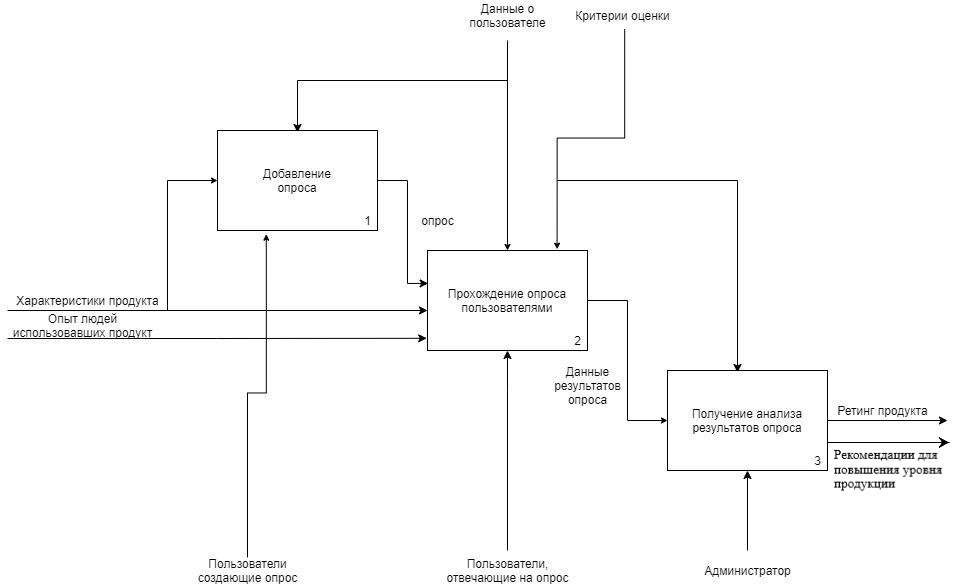


Рисунок 1.2 – Декомпозиция контекстной диаграммы «Оценка качества продукции»

На рисунке 1.3 отображена декомпозиция работы «Получение анализа результатов опроса», состоящая из трех блоков: «Подсчет оценок (ответов) опроса», «Анализ результатов опроса», «Выведение анализа результатов опроса».



Рисунок 1.3 – Декомпозиция работы «Получение анализа результатов опроса»

## 1.3 Анализ требований к разрабатываемому программному средству. Спецификация функциональных требований

Данное программное приложение представляет собой учет продукции в сфере услуг. Оно включает в себя два способа пользования данными: от имени администратора и от имени пользователя. Варианты использования программы у них различаются.

Пользователь может войти или зарегистрироваться. Пользователь может создать товар, описав его и добавив его фотографию. Ему можно будет оценить другие товары двумя разными способами: обычная оценка(краткое описание и сама оценка), а также опрос о товаре ( в виде да/нет). Разница между ними в том, какие ответы будут предлагаться к вводу. В первом случае это либо да, либо нет. Во втором варианте ответом будет считаться оценка от 1 до 5. Но редактировать и удалять опросы пользователь не может.

В случае необходимости у него есть возможность отправки сообщения администратору. Пользователь может вывести аналитику его опросов. Также пользователь может просмотреть отсортированные и/или отфильтрованные товаров или пользователей.

Теперь разберем возможности администратора. Зарегистрироваться администратором конечно же нельзя. Он не может создавать или проходить опросы. Вся его задача состоит в редактировании и удалении опросов и аккаунтов пользователя. Переложение функций редактирования и удаления опросов на администратора устраняет фальсификацию опроса недобросовестным пользователем. Вход в аккаунт администратора осуществляется таким же образом, что и в пользователь. Администратор может посмотреть отсортированные и/или отфильтрованные опросы и пользователей, как и пользователь.

Администратор может читать и удалять сообщения обыкновенных юзеров. Может заблокировать пользователя или задать ему статус. Также он может “войти” в любой аккаунт пользователя, однако там будут только функции редактирования и удаления опросов. Также администратор может выводить статистику пользователей, где отображается количество пройденных и созданных ими опросов.

Ниже приведена диаграмма вариантов использования (Рисунок 1.4)



Рисунок 1.4 – Диаграмма вариантов использования

## 1.4 Разработка информационной модели предметной области

Для выделения сущностей, атрибутов и первичных ключей происходит построение информационной модели предметной области. Общепринятым представлением графического изображения реляционной модели данных является ER-диаграмма. ER-диаграмма показывает связь данных между собой и какие данные хранит система.

При проектировании системы было принято решение использовать следующие сущности:

* *good*;
* *usr*;
* *user\_role*;
* *review;*
* *message*;
* *hibernate\_sequence.*

Графическое отображение информационной модели приведено на диаграмме рисунке 1.5.

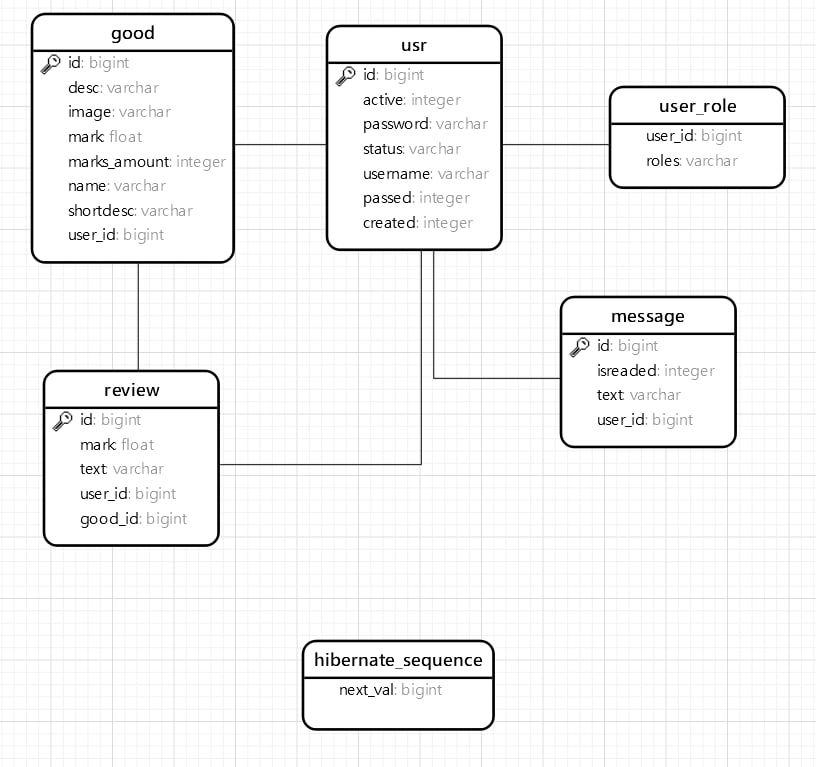


Рисунок 1.6 – Информационная модель системы

Ниже представлено подробное описание всех сущностей, входящих модель.

Сущность *good* содержит в себе следующие атрибуты:

* *id* – хранит идентификационный номер авторизованного продукции;
* *desc* – отвечает за хранение описания продукции;
* *image* – хранит изображения продукции;
* *mark* – хранит информацию о выставленных оценках;
* *marks\_amount* – необходимо для хранения количества выставленных оценок;
* *name* – хранит информацию о названии продукции;
* *shortdesk* – хранит краткое описание продукции;
* *user\_id* – хранит информацию идентификационного номера пользователя, добавившего продукцию.

Сущность *usr* содержит следующие атрибуты:

* *id*– хранит идентификационный номер пользователя;
* *active* – хранит данные об активности пользователя;
* *password – хранит пароль пользователя;*
* *status – хранит информацию о статусе пользователя;*
* *username – хранит информацию о имени пользователя в системе;*
* *passed – хранит информацию о количестве;*
* *created – хранит данные о дате создания пользователя.*

Сущность *user\_role* содержит в себе следующие атрибуты:

* *user\_id* – хранит идентификационный номер авторизованного пользователя;
* *roles* – отвечает за хранение роли пользователя.

Сущность *review* содержит в себе следующие атрибуты:

* *id*– хранит идентификационный номер авторизованного пользователя;
* *mark* – отвечает за хранение выставленных оценок ;
* *text* – хранит информацию об оставленных комментариях;
* *user\_id* – хранит идентификационный номер пользователя;
* *good­\_id* – хранит идентификационный номер продукции.

Сущность *message* содержит в себе следующие атрибуты:

* *id* – хранит идентификационный номер авторизованного пользователя;
* *isreaded* – отвечает за хранение данных о прочтении сообщения;
* *text* – хранит отправленный тект;
* *user\_id* – необходимо для хранения идентификационного номера пользователя, который отправил сообщение.

Сущность *hibernate\_sequence* содержит в себе следующие атрибуты:

* *next\_val* – хранит информацию о режиме «спячки».  
  *SQL*-скрипт для генерации базы данных приведен в приложении В.

## 1.5 Модели представления программного средства и их описание

Диаграмма состояний по существу является графом специального вида, который представляет некоторый автомат. Вершинами этого графа являются состояния и некоторые другие типы элементов автомата (псевдосостояния), которые изображаются соответствующими графическими символами. Дуги графа служат для обозначения переходов из состояния в состояние[6].

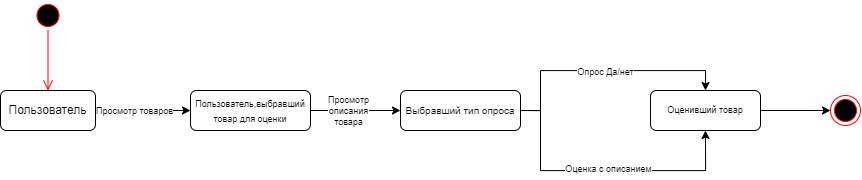


Рисунок 1.6 – Диаграмма состояний

Диаграмма последовательности, которая показывает авторизации пользователя представлена на рисунке 1.7.

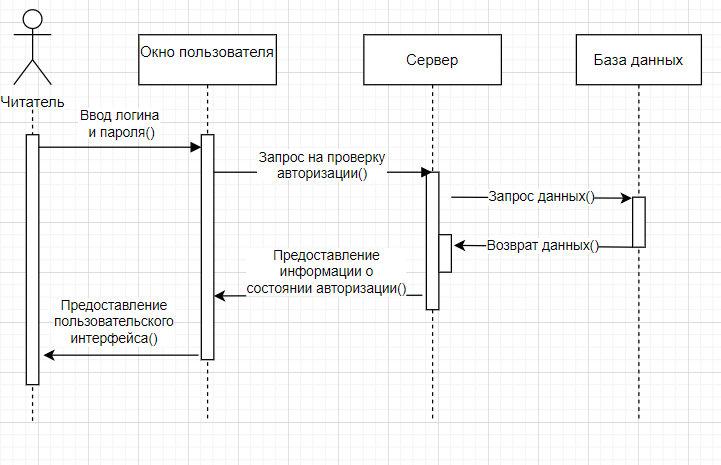


Рисунок 1.7 – Диаграмма последовательности

Диаграмма развёртывания. С помощью диаграммы развертывания отобразим элементы и компоненты системы, существующие на этапе ее исполнения.

Диаграмму развертывания системы приведена на рисунке 1.8[7].

На диаграмме отражено, что для ПК пользователя обязательно наличие среды выполнения *jdk* 1.8+, также, как и для ПК, который будет выступать в качестве сервера.

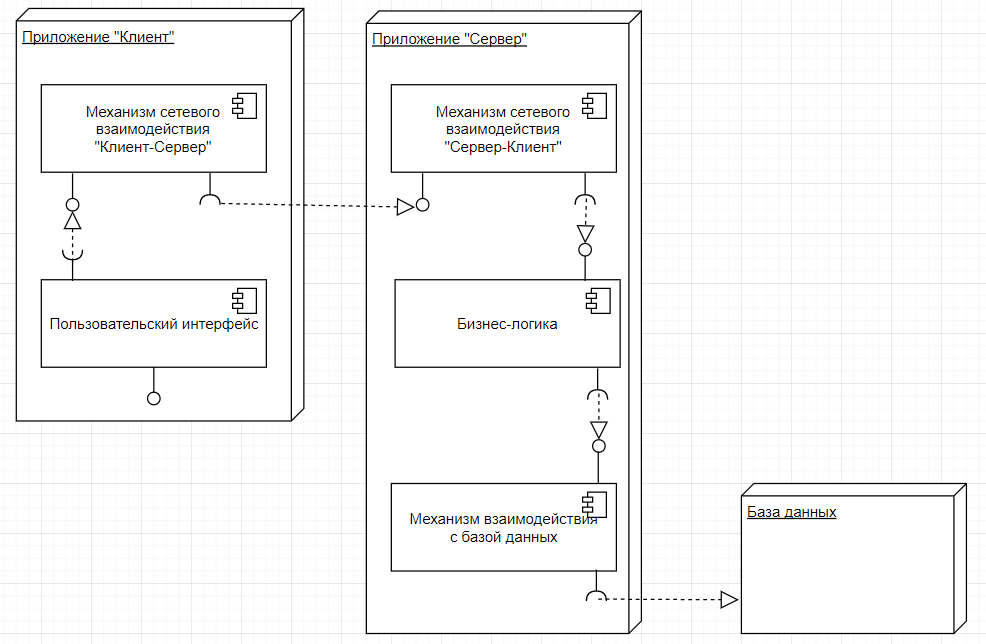


Рисунок 1.8 – Диаграмма развертывания

Таким образом, в данной главе курсового проекта представлены моделирование предметной области и разработка требований к программному средству, анализ и формализация бизнес-процессов службы управления персоналом, разработаны спецификация функциональных требований, информационная модель программного средства, а также модели представления программного средства на основе языка *UML*.

# Проектирование и конструирование программного средства

## 2.1 Постановка задачи

Чтобы разработанное программное средство было качественным, необходимо разобраться в самом процессе оценке уровня качества.

Так как у всей продукции есть документы с оценками, то необходимо спроектировать базу данных, которая осуществляла бы их хранение электронном виде. Это также упростит поиск информации, ее редактирование, добавление, удаление и фильтрацию.

Для работы сотрудников необходимо организовать удобный интерфейс со списком продукции, который позже можно будет преобразовать в мобильную версию. Для сотрудников будет доступна вся информация о продуктах, все поставленные оценки и написанные комментарии.

Добавление новой продукции будет осуществлять администратор. В его обязанности будет входить запись всех данных о продукции, а также осуществление добавления критериев для оценки качества. По итогу записи должен формироваться документ.

Для того, чтобы в дальнейшем провести анализ качества, будет вестись отчетность выставления оценок тому, или иному продукту за день, за неделю, за месяц. Это позволит увидеть все точки зрения пользователей, что может быть эффективно использовано для улучшения качества.

## 2.2 Архитектурные решения

Диаграммы классов используются при моделировании программных средств наиболее часто. На диаграммах классов показываются классы, интерфейсы и отношения между ними.

Диаграммы классов показаны на рисунках 2.1 – 2.2.



Рисунок 2.1 – Диаграмма классов(общая)

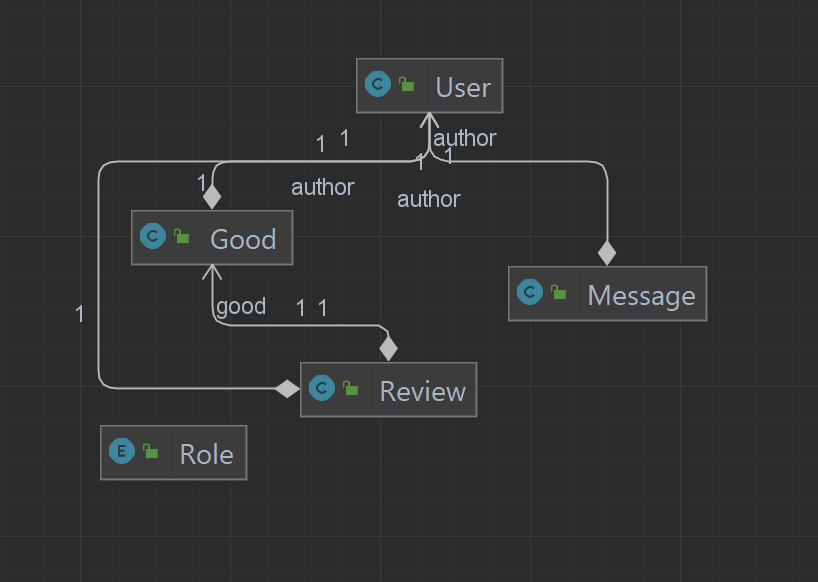


Рисунок 2.2 – Диаграмма классов пакета models

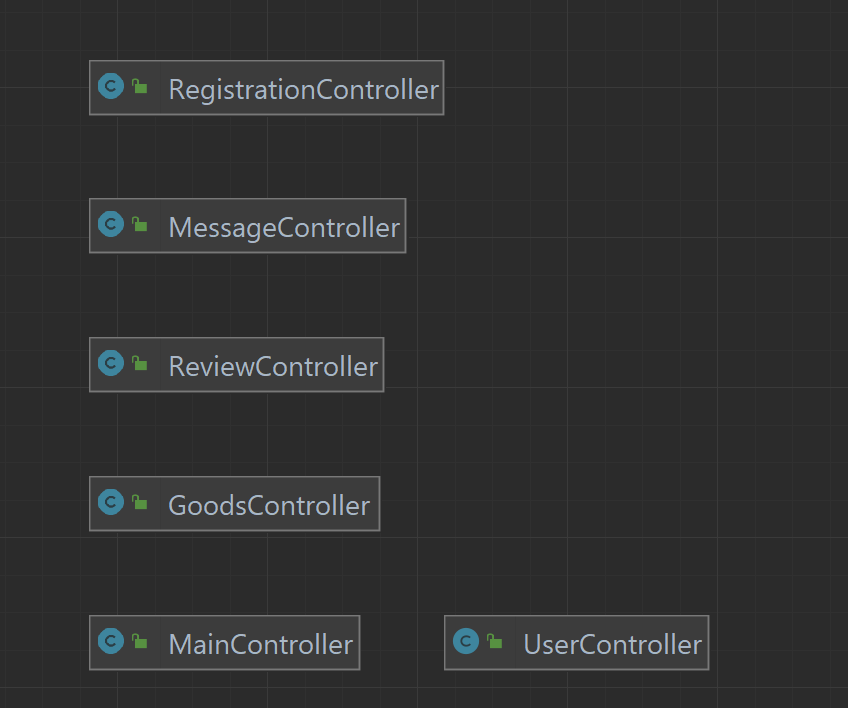


Рисунок 2.3 – Диаграмма классов пакета controllers

Экземпляры клиентских приложений размещаются на машинах сотрудников организации, на которых предварительно должен быть установлен JDK 1.8 и выше. Данные, необходимые для установки соединения с сервером, хранятся в конфигурационном файле клиентского приложения и доступны для настройки пользователем. Настройки, используемые всеми приложениями системы, хранятся в одной из таблиц базы данных. Это позволяет производить изменения, которые распространяются сразу на все экземпляры приложений, с другой стороны, доступ к настройкам системы закрыт для рядовых пользователей.

Система предусматривает один способ ввода информации – вручную через пользовательский интерфейс приложения и один способ вывода – на экран машины сотрудника.

Экземпляры приложений могут взаимодействовать между собой для пересылки уведомлений сотрудникам. Для этого используются протоколы стека TCP/IP. Сетевое взаимодействие обеспечивается средствами архитектуры java.net, сетевые настройки хранятся в базе данных.

Диаграмма компонентов описывает особенности физического представления системы. Она позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный и исполняемый код. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними. Диаграмму можно увидеть на рисунке 2.4.

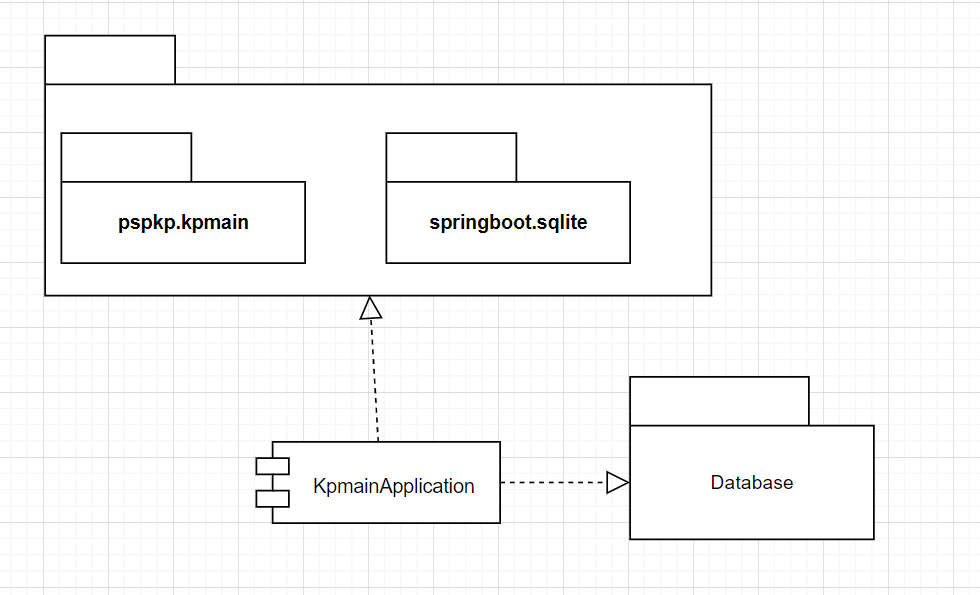


Рисунок 2.4 – Диаграмма компонентов

## 2.3 Описание алгоритмов, реализующих бизнес-логику разрабатываемого программного средства

Рассмотрим алгоритмы реализации ряда методов серверной части, соответствующих вариантам использования.

При запуске открывается главное окно, в котором пользователь должен пройти авторизацию. При успешной авторизации в зависимости от уровня доступа, пользователь получает разный набор возможностей для работы с программой. Если выполнен вход под администратором, то пользователь имеет доступ к базам данных добавлять/удалять/редактировать данные в них. Если же была выполнена авторизация в качестве пользователя, то ему предоставляется вывод продукции, выставление оценки, индивидуальное задание(см. рисунок 2.5).

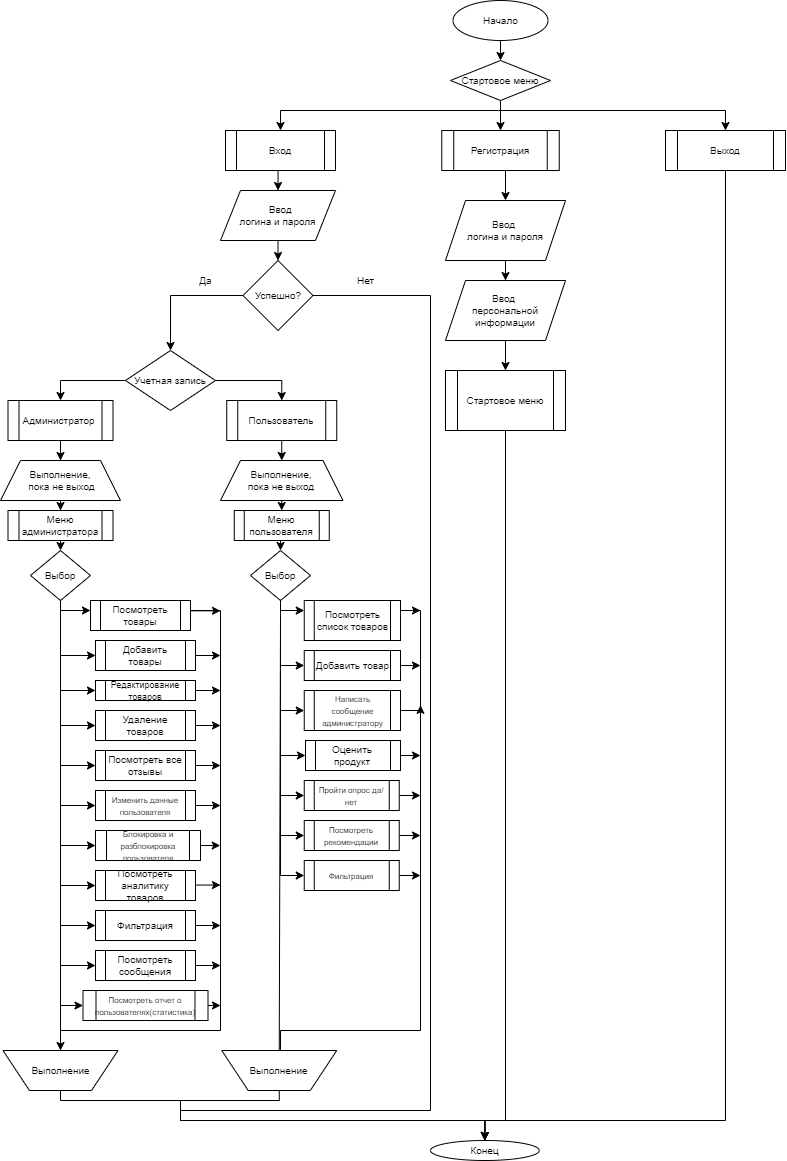


Рисунок 2.5 – Схема алгоритма работы программы

В начале работы каждому пользователю необходимо произвести авторизацию для того, чтобы можно было определить роль в программе и открыть нужный функционал. Для того, чтобы авторизоваться, пользователю необходимо ввести логин и пароль и после этого происходит соединение с БД и на сервер отправляется запрос выборки данных о пользователях. Далее в цикле проверяется совпадение введенных данных с поступающими данными из базы данных. В случае несовпадения вызывается окно ошибки и пользователю предлагается ввести данные повторно.

Алгоритм данной функции представлен на рисунке 2.6.

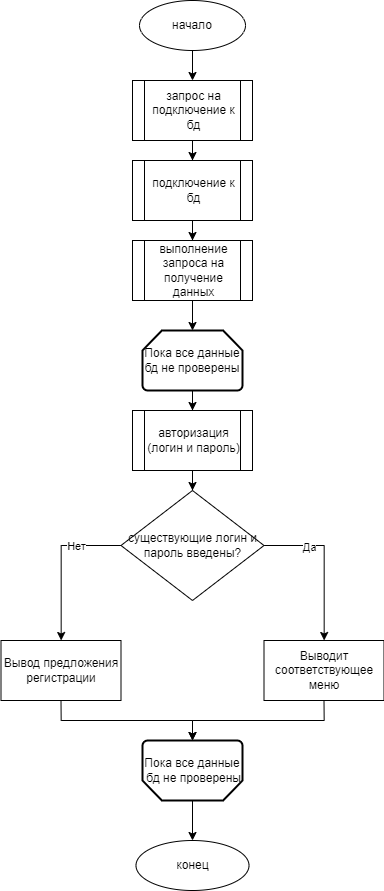


Рисунок 2.6 – Схема алгоритма работы функции авторизации

Схема алгоритма клиент-серверного взаимодействия. В приложении реализована архитектура клиент-сервер. Поэтому, все запросы клиента отправляются на сервер, который обрабатывает их и отсылает ответ. Данная схема приведена на рисунке 2.7.

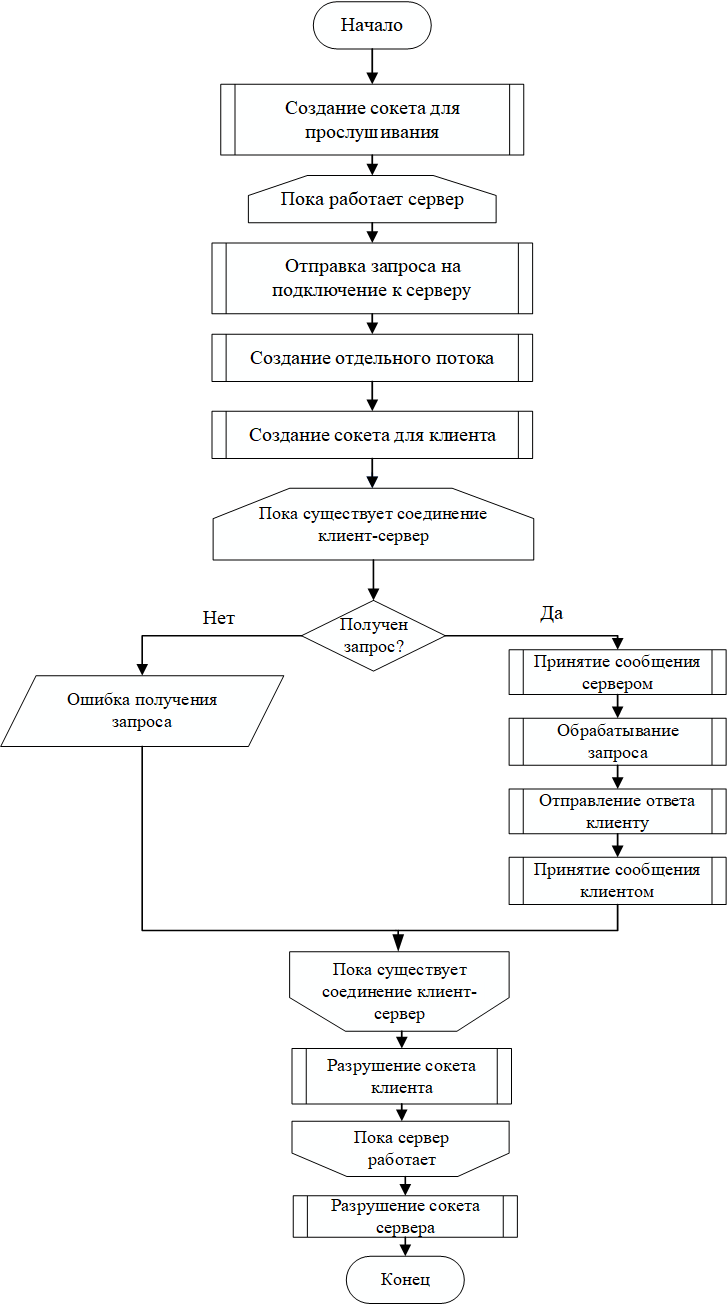


Рисунок 2.7 – Схема алгоритма клиент-серверного взаимодействия

## 2.4 Проектирование пользовательского интерфейса

Пользовательский интерфейс – это интерфейс, с помощью которого человек может управлять программным средством. Пользовательский интерфейс должен быть удобным в использовании, чтобы взаимодействие с ним происходило на максимально интуитивном уровне.

Графический пользовательский интерфейс – это вид пользовательского интерфейса, в котором все элементы (кнопки, меню, пиктограммы, списки и другое) представленные пользователю на дисплее, выполнены в виде картинок, графики.

При запуске программы, пользователю выводится окно авторизации. Данное окно служит для предоставления ему определенных прав в работе программы(рисунок 2.8).

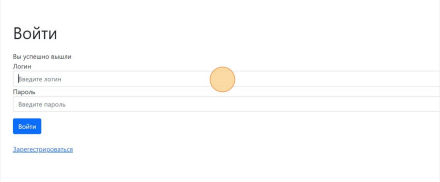


Рисунок 2.8 – Окно авторизации

При успешной авторизации администратора на экране появляется главное и основное меню администратора.

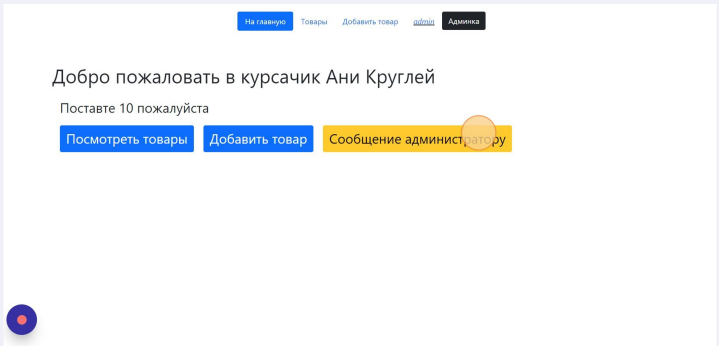


Рисунок 2.9 – Главное окно администратора

Все компоненты взаимодействуют друг с другом и связаны. Также могут работать по отдельности. Для каждой части разработана таблица в которой содержатся и хранятся данные.

При запуске программы появится окно аутентификации (проверка имени и пароля пользователя программы). Данное окно служит для идентификации пользователя программы в целях предоставления ему определенных прав в работе с программой.

## 2.5 Обоснование выбора компонентов и технологий для реализации программного средства

Данное программное приложение будет иметь архитектуру «клиент-сервер» и написано оно будет на языке программирования Java.

Сервер – логический процесс, который обеспечивает некоторый сервис по запросу от клиента. Обычно сервер не только выполняет запрос, но и управляет очередностью запросов, буферами обмена, извещает своих клиентов о выполнении запроса и т. д. Клиент – процесс, который запрашивает обслуживание от сервера. Процесс не является клиентом по каким-то параметрам своей структуры, он является клиентом только по отношению к серверу.

Достоинства архитектуры «клиент-сервер»:

– центральное хранилище файлов;

– простая управляемость при большом числе пользователей;

– сильная централизованная защита;

– нет дублирования кода.

Связь между сервером и клиентом в приложении осуществляется при помощи TCP/IP. Также в реализации системы используются следующие техники:

* разработка и использование собственной иерархии, расширение базовых классов, предоставляемых JDK;
* реализация двух паттернов проектирования;
* использование сокрытия данных (инкапсуляция), перегрузка методов, переопределение методов, абстрактные типы данных (интерфейсы, абстрактные классы), статические методы, обработка исключительных ситуаций.

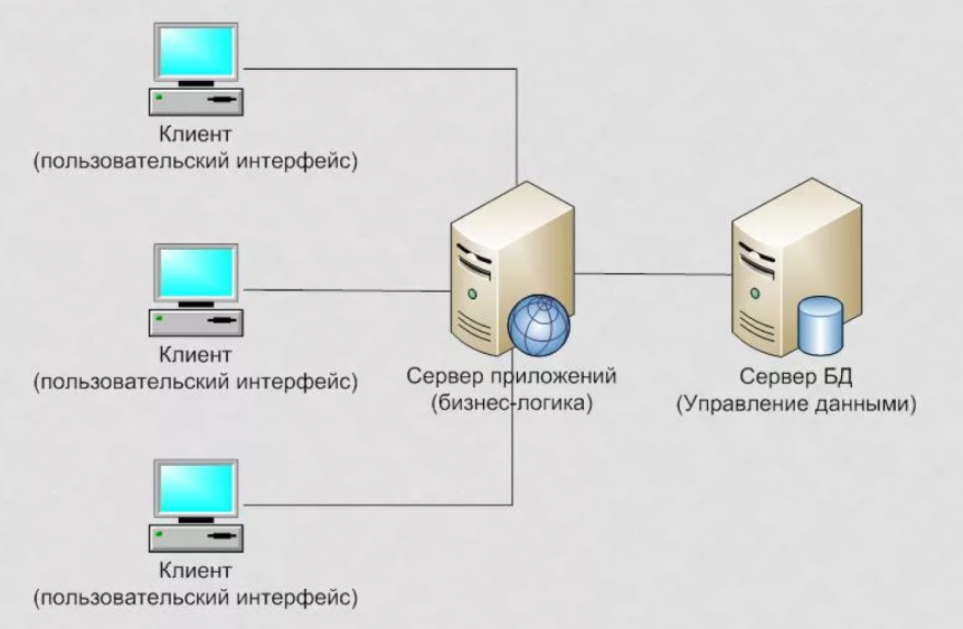


Рисунок 2.10 – Архитектура «клиент-сервер»

Для более удобного хранения данных необходимо спроектировать базу данных, которая может осуществить хранение данных, и дает возможность поиска информации, ее редактирование, добавление, удаление и фильтрацию. В качестве основной среды разработки в проекте используется Intellij IDEA. Для создания UML-диаграмм в проекте используется средство *Drawio*, которое позволяет осуществлять создание диаграмм вариантов использования, состояний, последовательности с помощью просто моделирования. Для выполнения UML-моделей в стандарте IDEF0 используется CASE-средство CA AllFusion Process Modeler r7 (BPwin). Для информационного моделирования применяется средство CA AllFusion ERwin Data Modeler r7 (ERwin). Подключение к нему будет осуществляться при авторизации и в зависимости от полученных прав, пользователь будет получать определённый набор возможностей для выполнения своей работы.

# Тестирование и проверка работоспособности программного средства

Тестирование – это процесс проверки функционала программы с целью подтверждения того, что она работает в соответствии с определёнными требованиями. Юнит-тест (unit test), или модульный тест, — это программа, которая проверяет работу небольшой части кода. Разработчики регулярно обновляют сайты и приложения, добавляют фичи, рефакторят код и вносят правки, а затем проверяют, как всё работает. Проверять всё это надо.

Junit — библиотека для модульного тестирования программного обеспечения на языке java[8].

Для тестирования процесса авторизации на сервере и проверки правильности выбранной роли для входа в систему был написан класс KpmainApplicationTests.

В ходе тестирования функций получения роли авторизации и выборки из таблицы выяснилось, что функция работает корректно и проходит все тесты. Результат тестирования функции представлен на рисунке 3.1.

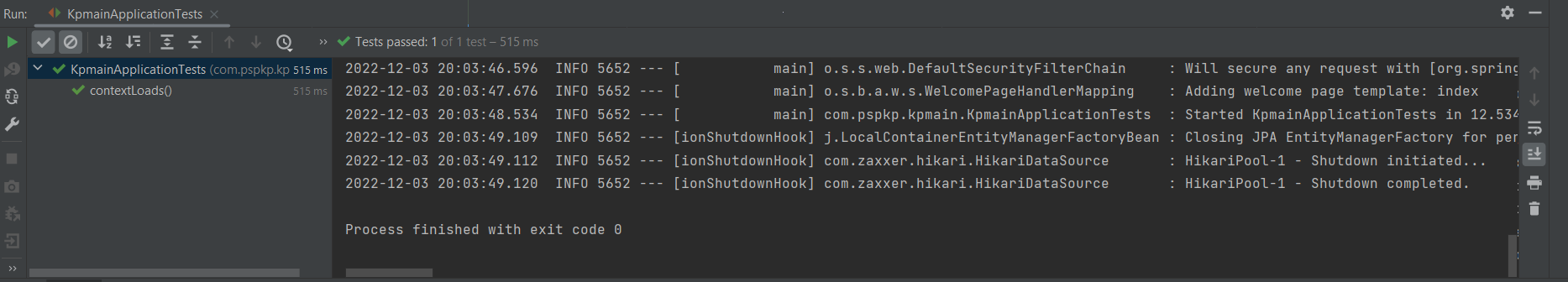


Рисунок 3.1 – Результат тестирования процесса авторизации

Также был написан тест, который проверял добавление объекта Good в модель:

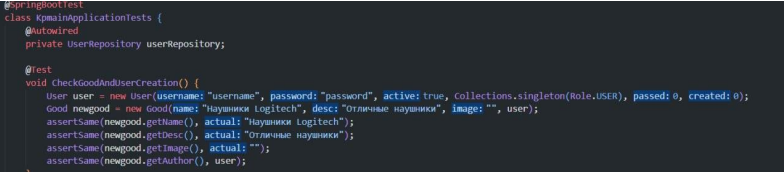


Рисунок 3.2 – Тест на добавление объекта

Для тестирования блокировки пользователя был написан метод BanUser(рисунок 3.3).

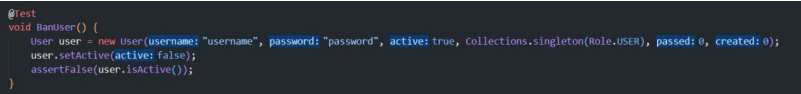


Рисунок 3.3 – Тест на блокировку пользователя

В ходе тестирования метода, который возвращает короткое описание товара(рисунок 3.4).

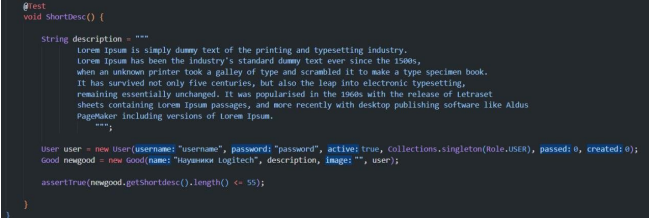


Рисунок 3.4 – Тест метода, который возвращает короткое описание товара

Провальный тест на поиск несуществующего пользователя (рисунок 3.5):



Рисунок 3.5 – Провальный тест

Так как один тест мы намеренно сделали провальным – увидим такой результат(рисунок 3.6):

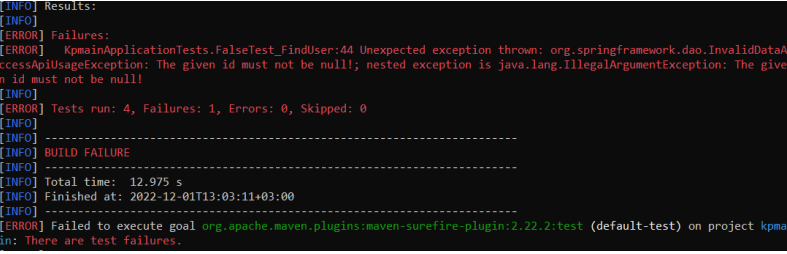


Рисунок 3.6 – Результат тестирования

Плагин surfare сгенерировал отчёт:

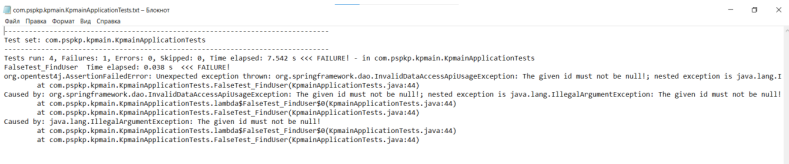


Рисунок 3.7 – Отчет о результатах тестирования

Тестирование необходимо потому, что все мы совершаем ошибки. Некоторые из них могут быть незначительными, в то время как другие – иметь самые разрушительные последствия. Все, что производится человеком, может содержать ошибки. Именно поэтому любой продукт нуждается в проверке – тестирование, прежде чем его можно будет эффективно и безопасно использовать.

# Руководство по развертыванию и использованию программного средства

* 1. Запуск приложения

Для запуска серверной части необходимо установить плагины Maven. Необходимо открыть cmd, перейти в директорию с pom.xml файлом(рисунок 4.1).

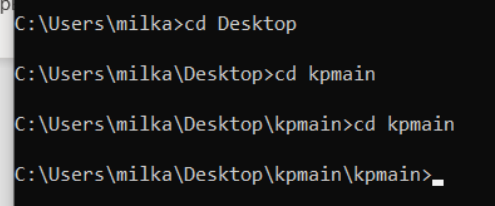


Рисунок 4.1 – Переход в необходимую директорию

При условии удачно установленного Maven – ввести команду *mvn spring-boot:run (*рисунок 4.2*).*



Рисунок 4.2 – Команда для запуска проекта

Результат запуска приложения(рисунок 4.3):

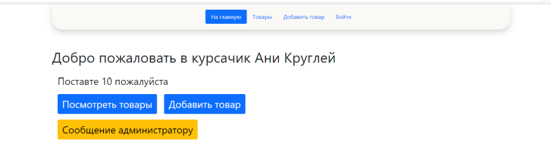


Рисунок 4.3 – Результат удачного выполнения команд

* 1. Описание формы авторизации

При запуске приложения требуется ввести логин и пароль и нажать на кнопку «Войти». Также, если входит незарегистрированный в системе студент, то он может нажать на соответствующую кнопку «Зарегистрироваться». Для входа не нужно выбирать роли, так как сервер, отправляя данные авторизации в БД, сам определяет, кто входит в систему и открывает соответствующее меню (рисунок 4.4).

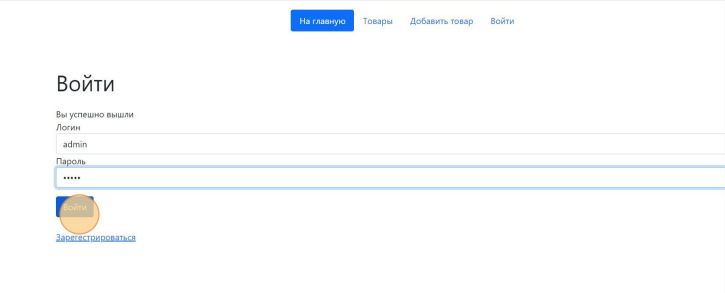


Рисунок 4.4 – Авторизация

Если у пользователя аккаунта еще нет – его можно зарегистрировать(рисунок 4.5).

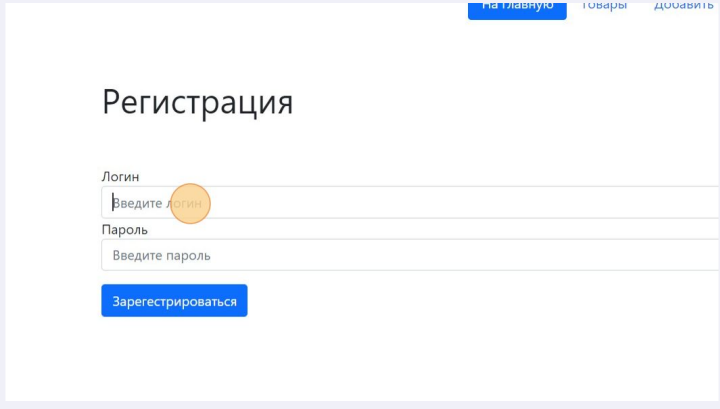


Рисунок 4.5 – Регистрация пользователя

После регистрации пользователю необходимо ввести ранее зарегистрированные данные, для входа в систему. Если введенные данные верны, то пользователю будет предоставлено меню, согласно его роли.

* 1. Меню пользователя

После удачной авторизации, у пользователя на экране появляется главное меню(рисунок 4.6).

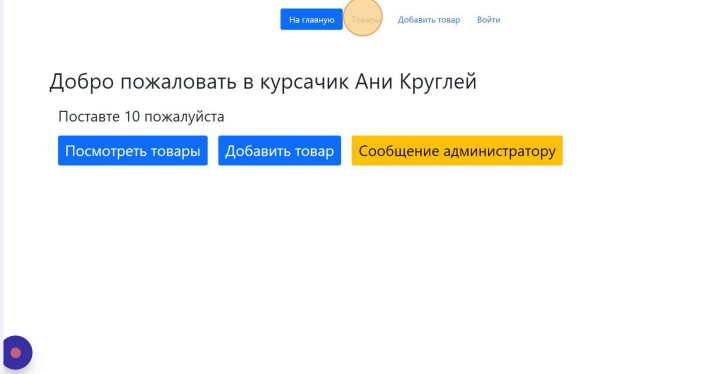


Рисунок 4.6 – Главное меню пользователя

При переходе на страницу «Товары» – пользователь может посмотреть все созданные товары(рисунок 4.7).

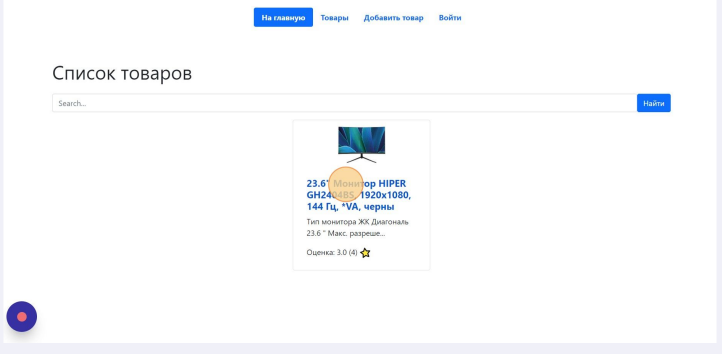


Рисунок 4.7 – Страница «Товары»

При переходе на страницу товара пользователь может посмотреть все отзывы на товар , его полное и краткое описание и оценки( рисунок 4.8).

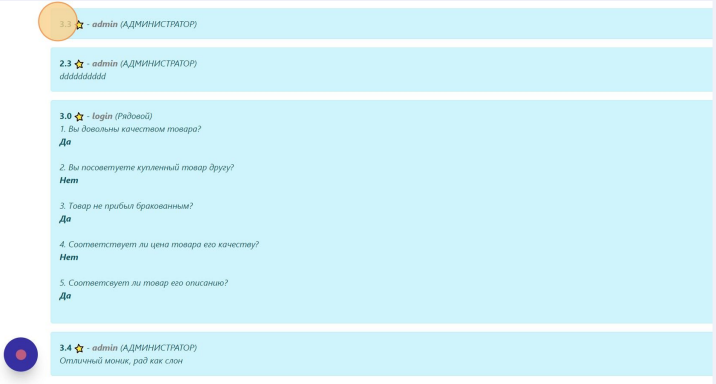


Рисунок 4.8 – Страница выбранного товара

При переходе на страницу «Добавить товар» пользователю предоставляется возможность описать свой товар, добавить полное его описание и картинку(рисунки 4.9 – 4.12).

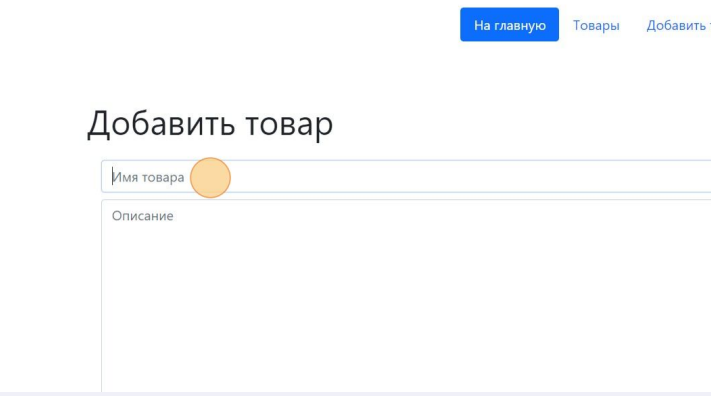


Рисунок 4.9 – Добавление товара

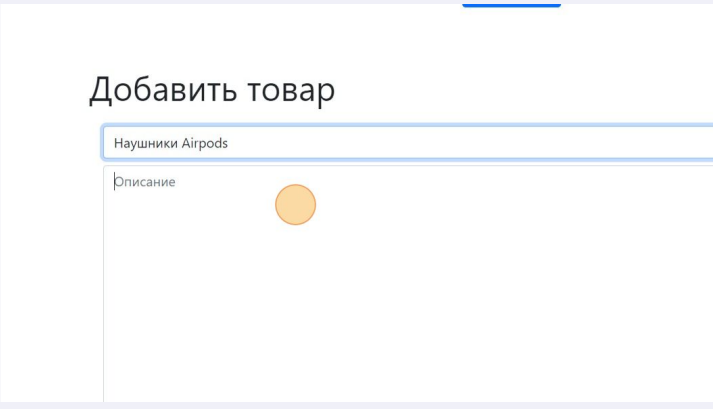


Рисунок 4.10 – Ввод названия товара

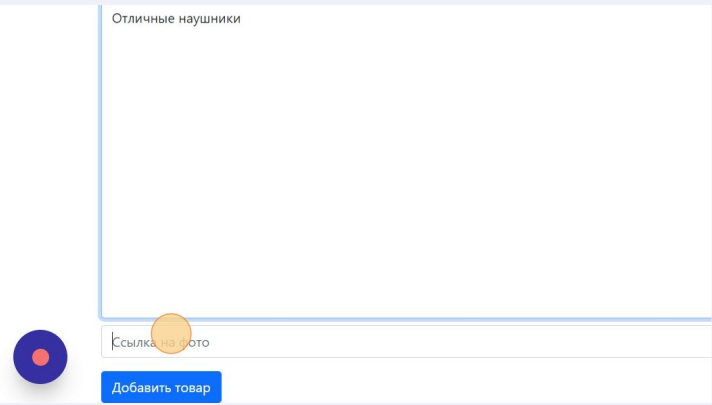


Рисунок 4.11 – Добавление описания товара

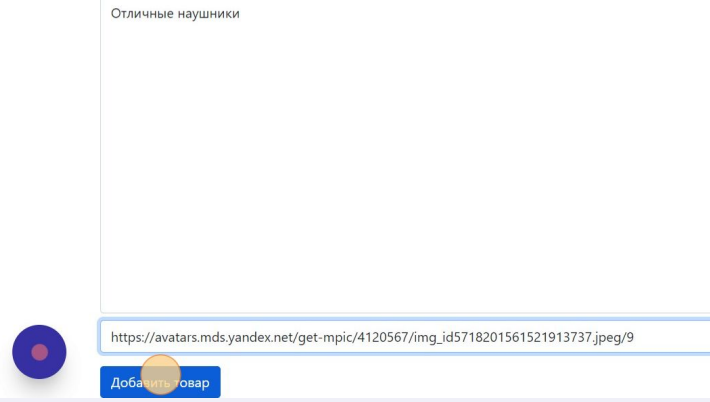


Рисунок 4.12 – Добавление ссылки на фотографию

После заполнения всех полей в список товаров добавится новый (рисунок 4.13).

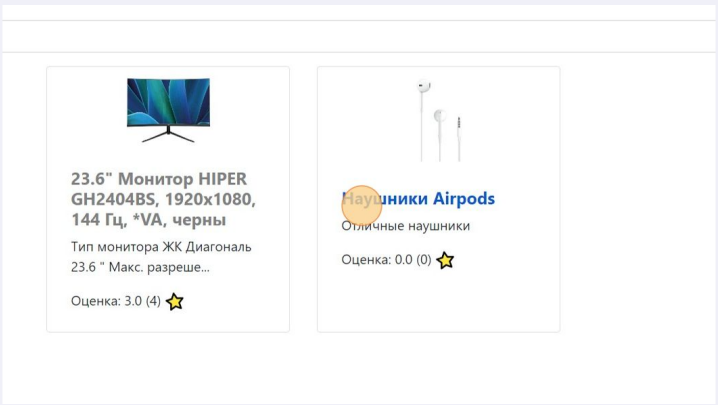


Рисунок 4.13 – Обновленный список товаров

Также пользователь может выбрать товар для оценки. Выставление оценок возможно двумя способами : отзыв с оценкой и опрос «Да\Нет»(рисунки 4.14-4.20). Также пользователь может просмотреть рекомендации по повышению качества продукции.

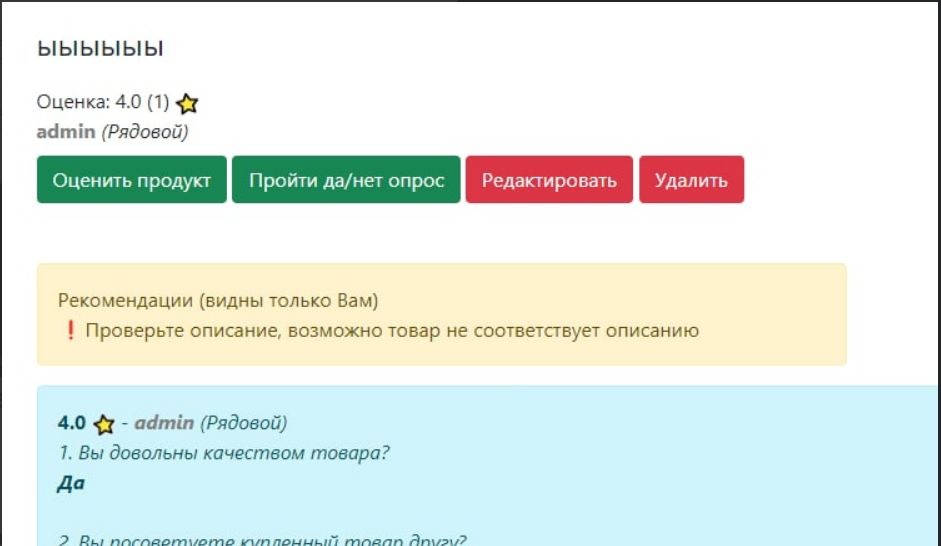


Рисунок 4.14 – Оценка выбранного продукта



Рисунок 4.15 – Написание отзыва

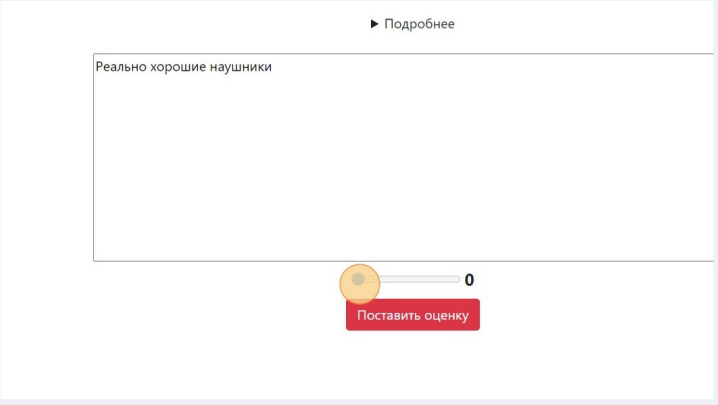


Рисунок 4.16 – Выставление оценки

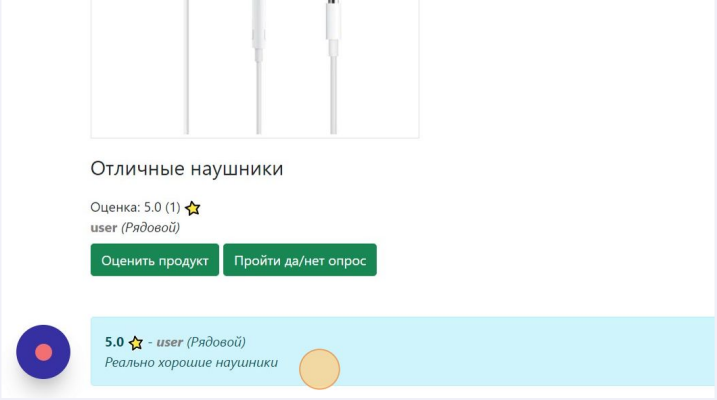


Рисунок 4.17 – Сохраненная оценка на странице товара

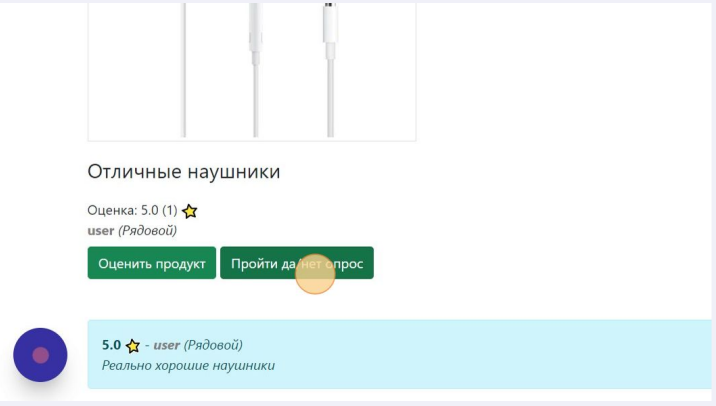


Рисунок 4.18 – Пройти «Да/Нет» опрос

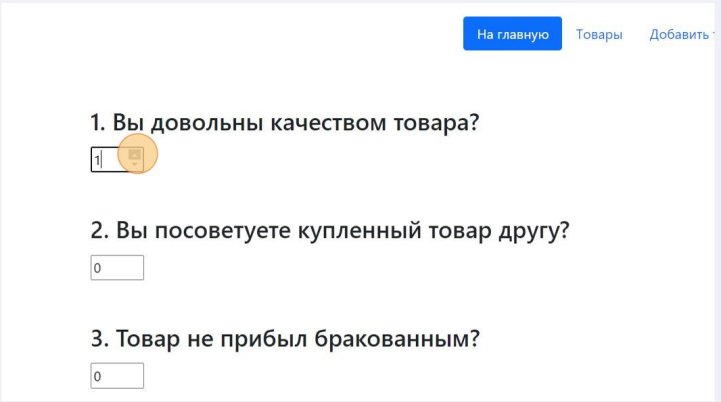


Рисунок 4.19 – Выбор необходимых ответов

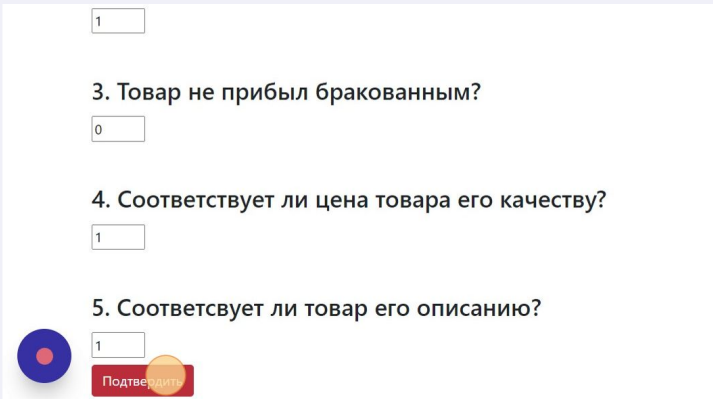


Рисунок 4.20 – Подтверждение выбранных ответов

Также пользователю доступен поиск товара по названию(рисунки 4.21 – 4.22).

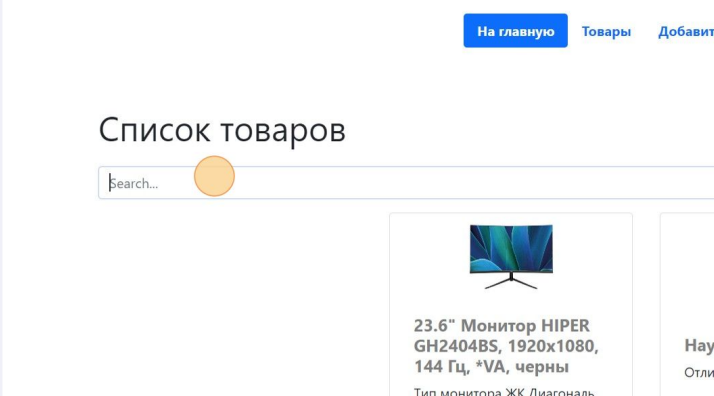


Рисунок 4.21 – Ввод названия для поиска

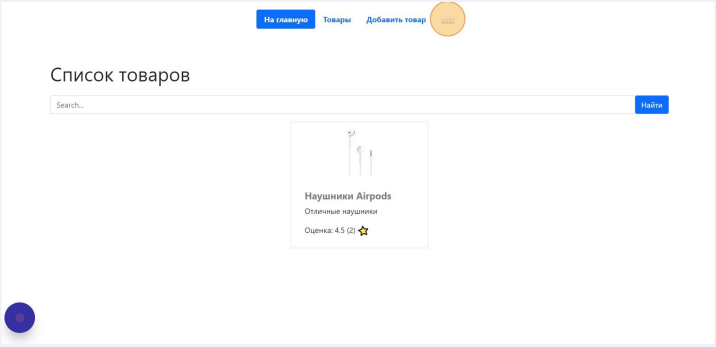


Рисунок 4.22 – Результат поиска

При необходимости, у пользователя есть возможность написать сообщение администратору(рисунки 4.23-4.24).

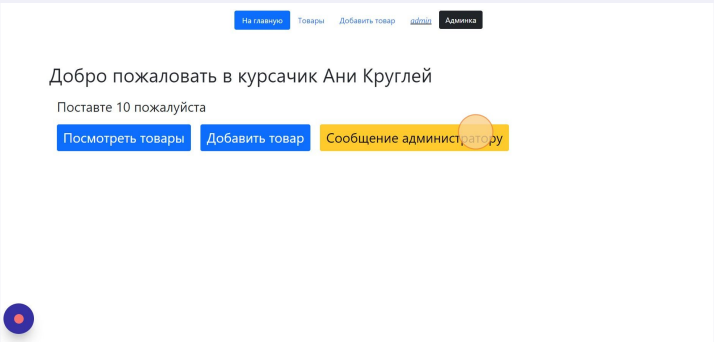


Рисунок 4.23 – Нажатие кнопки «Сообщение администратору»

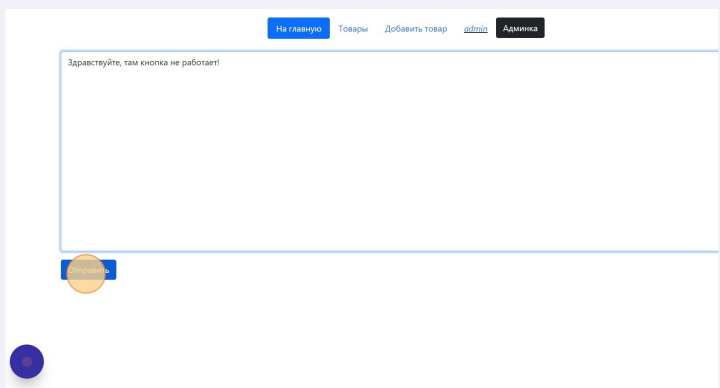


Рисунок 4.24 – Отправление сообщения администратору

При нажатии на свой логин(верхний правый угол), пользователь выходит из аккаунта(рисунок 4.25).

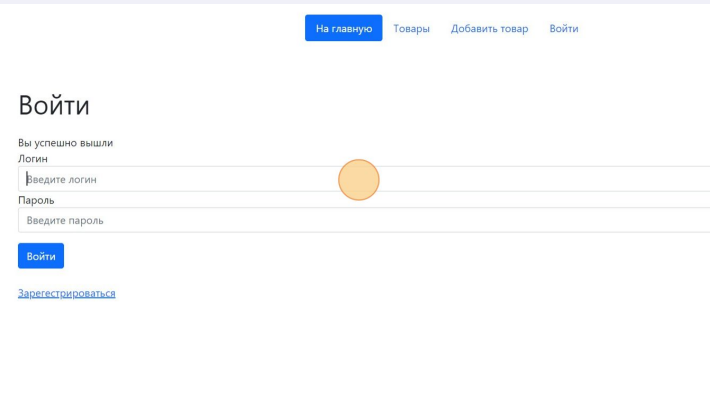


Рисунок 4.25 – Выход из аккаунта

* 1. Меню администратора

После удачной авторизации администратора, на экране появится главное меню администратора(рисунок 4.26).

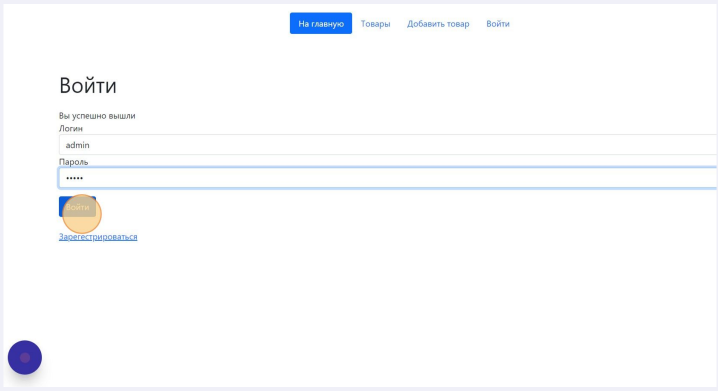


Рисунок 4.26 – Меню администратора

При переходе на страницу товаров, администратору предлагается возможность редактировать и удалять продукцию(4.27-4.31).



Рисунок 4.27 – Выбор товара для редактирования



Рисунок 4.28 – Изменение необходимой информации

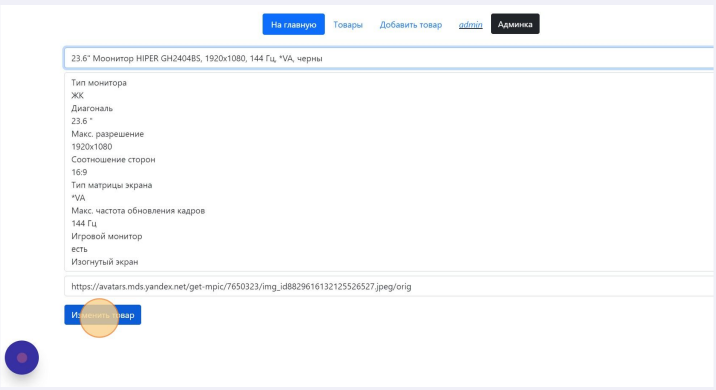


Рисунок 4.29 – Сохранение изменений

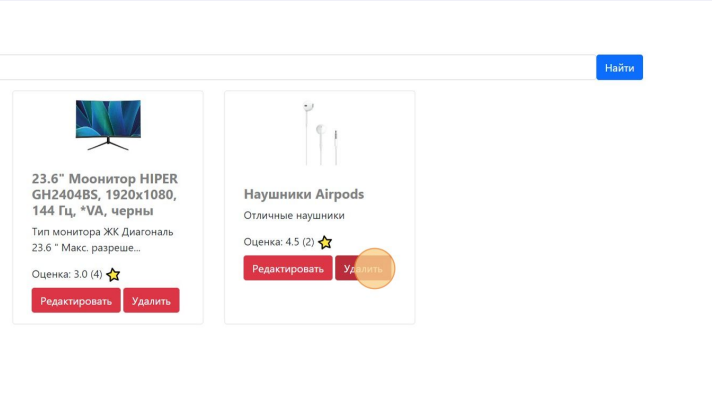


Рисунок 4.30 – Выбор продукции для удаления

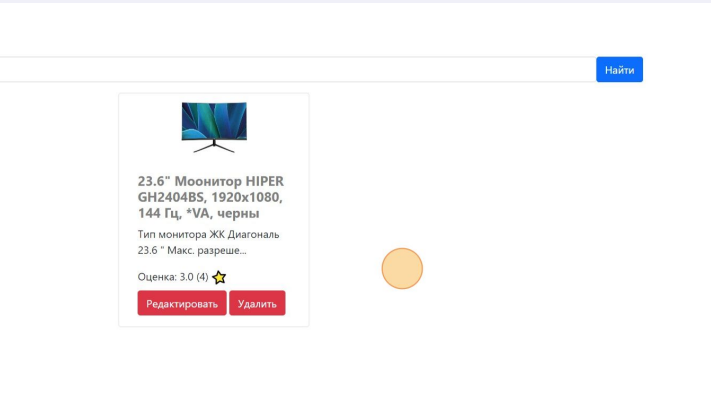


Рисунок 4.31 – Обновленный список товаров

Также для администратора есть отдельное меню «Админка»(рисунок 4.32).

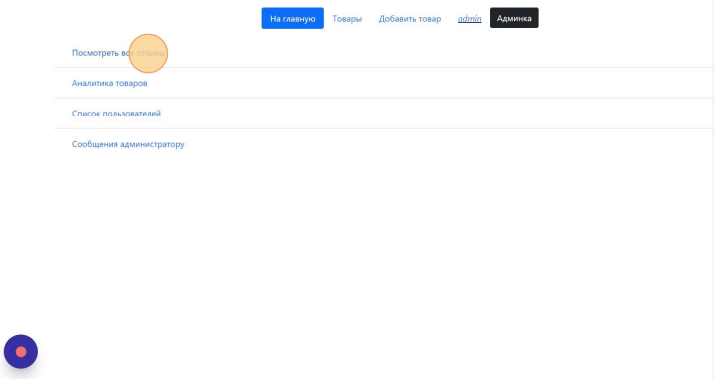


Рисунок 4.32 – Дополнительное меню администратора

После перехода на страницу «Посмотреть все отзывы, на экран выводится следующее (рисунок 4.33) :

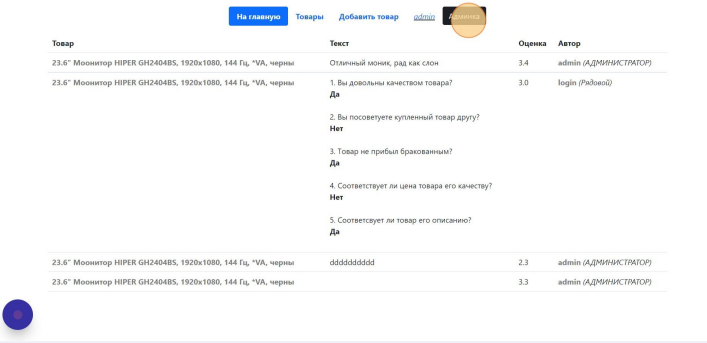


Рисунок 4.33 – Список всех отзывов

Если выполнить аналогичное действие со страницей «Аналитика товаров»(рисунок 4.34), то на эк ран выводится статистика , которая включает в себя среднюю оценку товаров, количество созданных товаров и пройденных опросов.

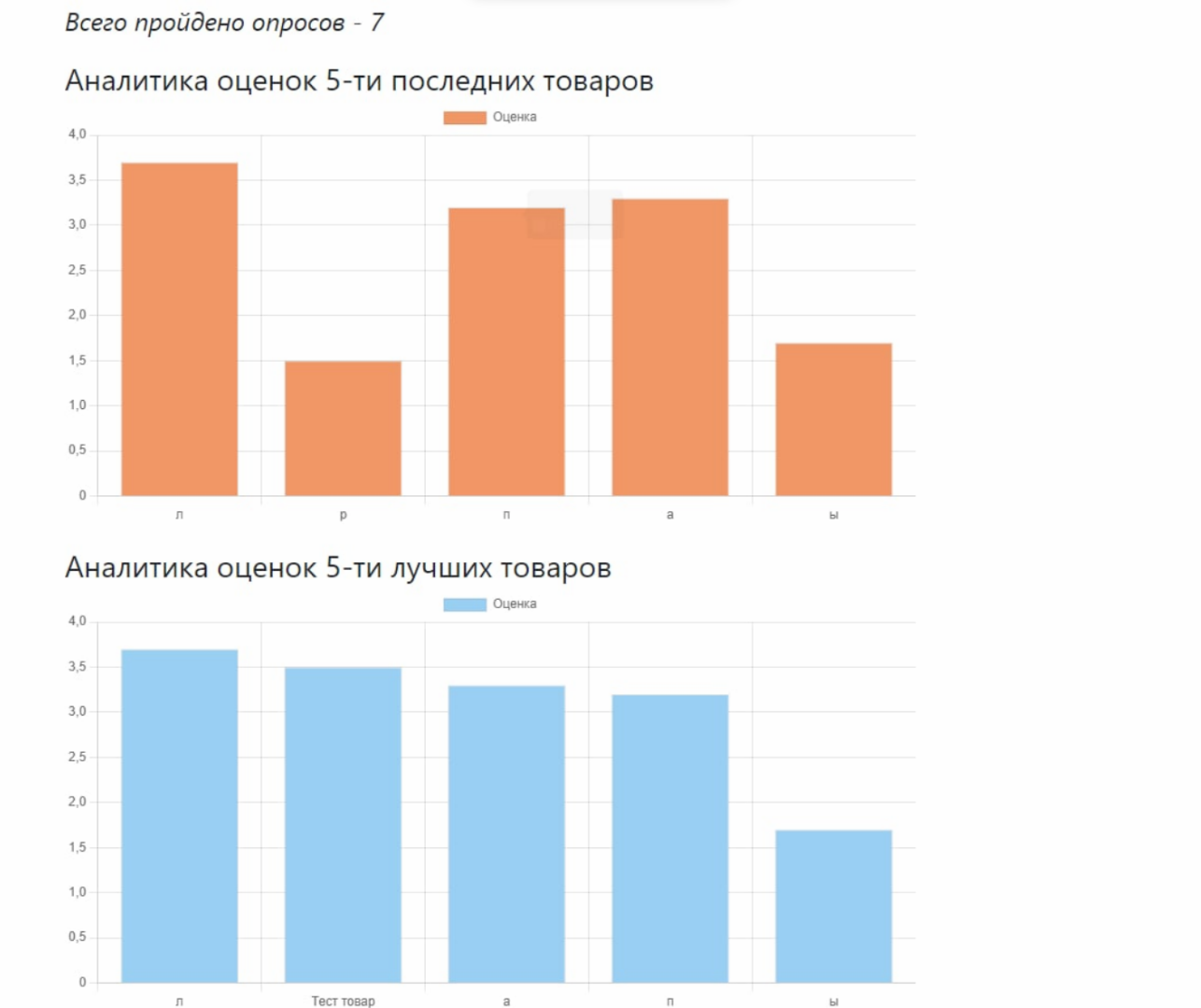


Рисунок 4.34 – Аналитика товаров

При переходе на страницу «Список пользователей», на экране будет выведен список всех пользователей, с возможностью редактирования, блокировки и разблокировки пользователя(рисунки 4.35 – 4.40).

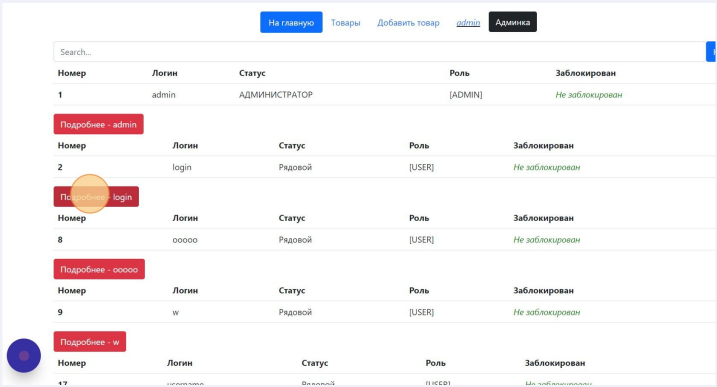


Рисунок 4.35 – Список пользователей

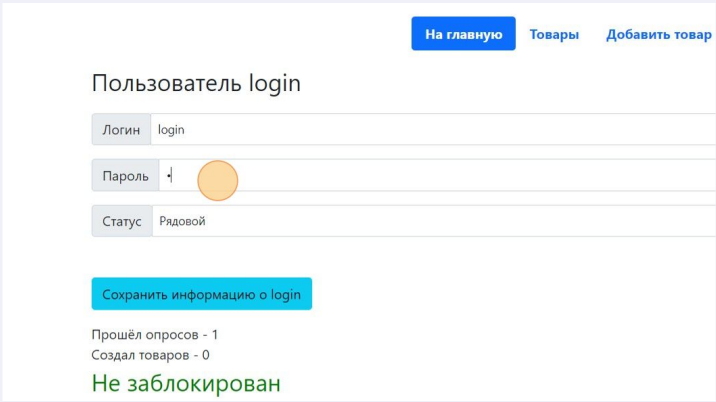


Рисунок 4.36 – Редактирование пользователя

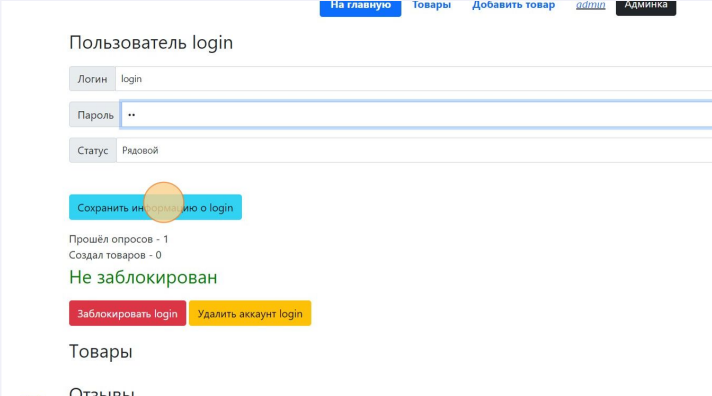


Рисунок 4.37 – Сохранение изменений пользователя

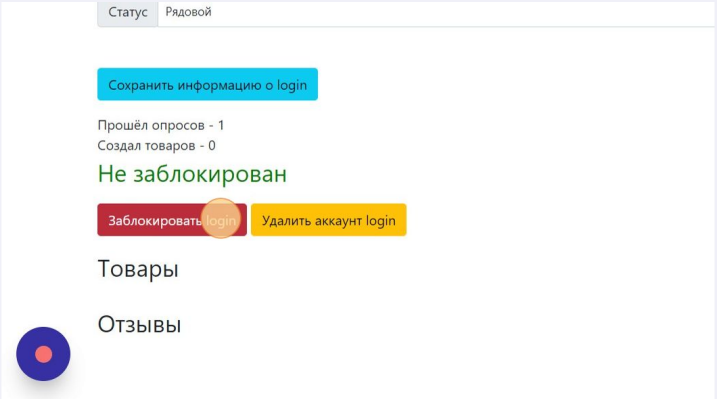


Рисунок 4.38 – Блокировка пользователя

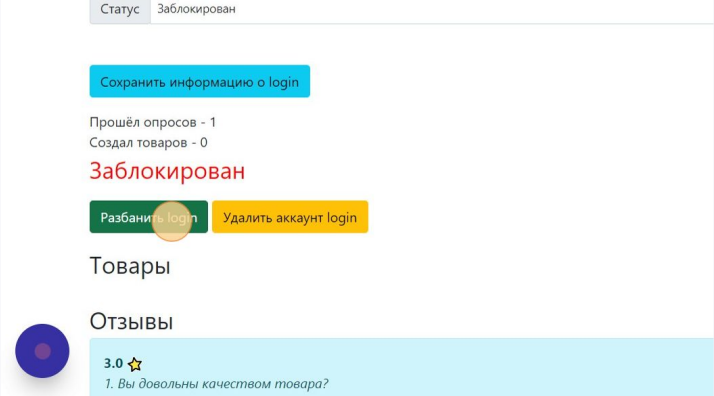


Рисунок 4.39 –Разблокировка пользователя

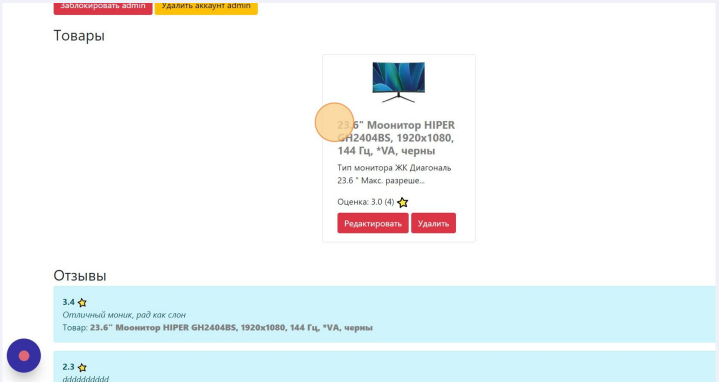


Рисунок 4.40 – Список всех отзывов и товаров пользователя

Также администратор может посмотреть все сообщения, которые были ему отправлены(серым отмечены сообщения, которые прочитаны) (рисунки 4.41 – 4.42).



Рисунок 4.41 – Список сообщений

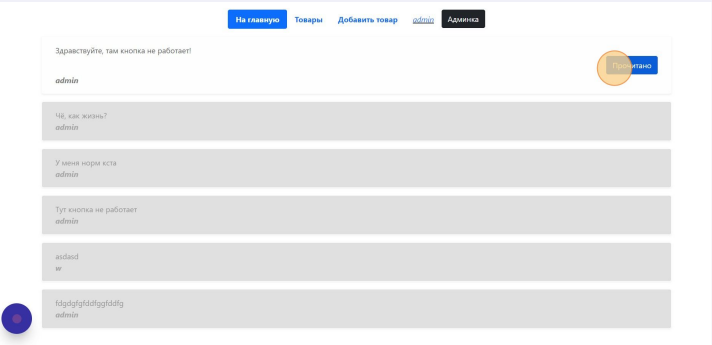


Рисунок 4.42 – Отметить сообщение как прочитанное

Также для администратора и для пользователя возможна фильтрация товаров по оценке(рисунок 4.43).

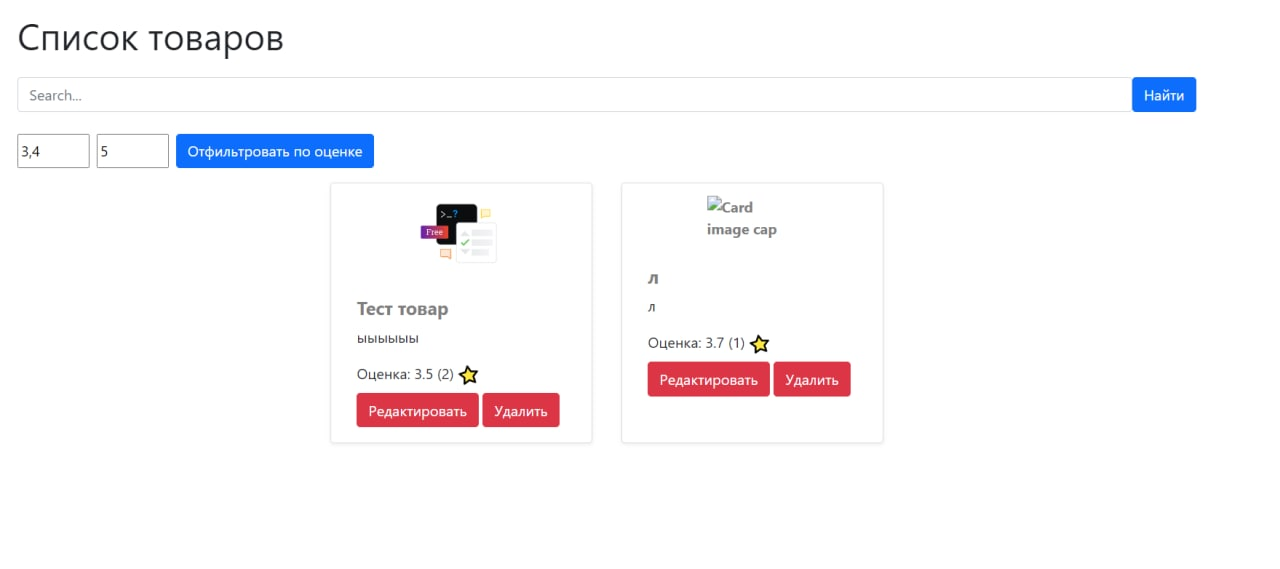


Рисунок 4.43 – Фильтрация товаров

И аналогичное действие возможно и с пользователями (только для администратора) (рисунок 4.44).

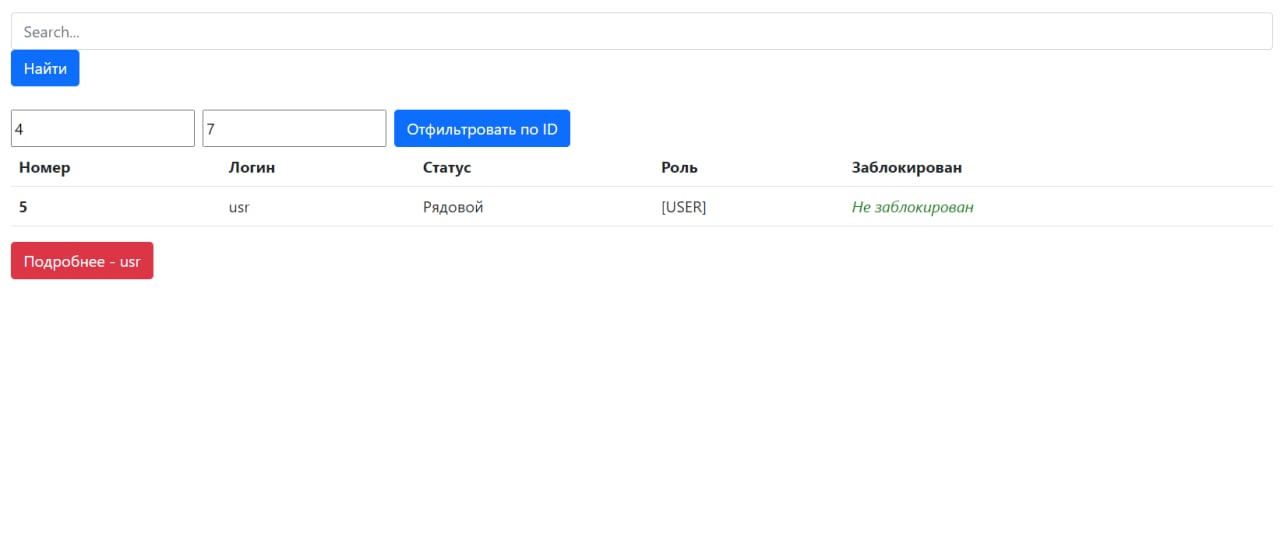


Рисунок 4.44 – Фильтрация пользователей по ID

# Заключение

В результате выполнения данного курсового проекта было разработано программное приложение, позволяющие оптимизировать работу организации. В нем предусмотрена система администрирования, а также проверка учетных данных при авторизации пользователей.

Приложение удовлетворяет основным характеристикам, которые были заявлены. Оно удобно в эксплуатации, целостно, конкретизировано в рамках заданной предметной области.

При разработке данного программного продукта была учтена логика пользователя, и интерфейс данной программы сделан удобным и понятным.

Использование данной программы в реальной организации сделает работу автоматизированной, более быстрой и упрощенной. В дальнейшем данная программа может редактироваться и совершенствоваться в соответствии с требованиями предметной области. Подводя итог, можно сказать, что цели и задачи, поставленные перед данной работой, успешно достигнуты и выполнены.

Выполнено проектирование и разработка программного средства оценки качества продукции: выполнена постановка задачи и определены основные методы ее решения; в ходе объектного моделирования системы построен ряд *UML*-диаграмм; разработана информационная модель системы, представленная в виде схемы базы данных; описаны основные алгоритмы работы программного средства; разработано руководство пользователя; выполнено тестирование системы, показавшее ее соответствие функциональным требованиям, поставленным в задании на разработку.

# Список использованных источников

1. Что такое показатель качества? Определение. [Электронный ресурс]. – Режим доступа: <https://www.calltouch.ru/blog/glossary/pokazatel-kachestva/?ysclid=lb7xzum2p048778413> .
2. Повышение качества продукции на предприятии: цели, методы и этапы [Электронный ресурс]. – Режим доступа: <https://www.kom-dir.ru/article/3820-povyshenie-kachestva-produktsii?ysclid=lb74vf0qti722556157> .
3. Ефимов, В. Улучшение качества продукции, процессов, ресурсов. - Москва : КноРус, 2010. - 240 с.
4. Клиент-сервер [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://www.mstu.edu.ru/study/materials/zelenkov/ch>
5. Нотация IDEF0 [Электронный ресурс]. – Электронный данные. – Режим доступа: <https://www.businessstudio.ru/wiki/docs/v4/doku.php/ru/csdesign/bpmodeling/idef0>
6. Фаулер, М. UML. Основы. – 3-е изд. – СПб. : Символ-плюс, 2006. –192 с.
7. Ларман, К. Применение UML 2.0 и шаблонов пректирования. Введение в объектно-ориентированный анализ и проектирование. – 3-е изд. – СПб. : Вильямс, 2012. – 736 с.
8. Руководство по JUnit [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://proselyte.net/tutorials/junit/introduction/> .

# Приложение А

# (обязательное)

## Отчет о проверке на заимствования в системе «Антиплагиат»

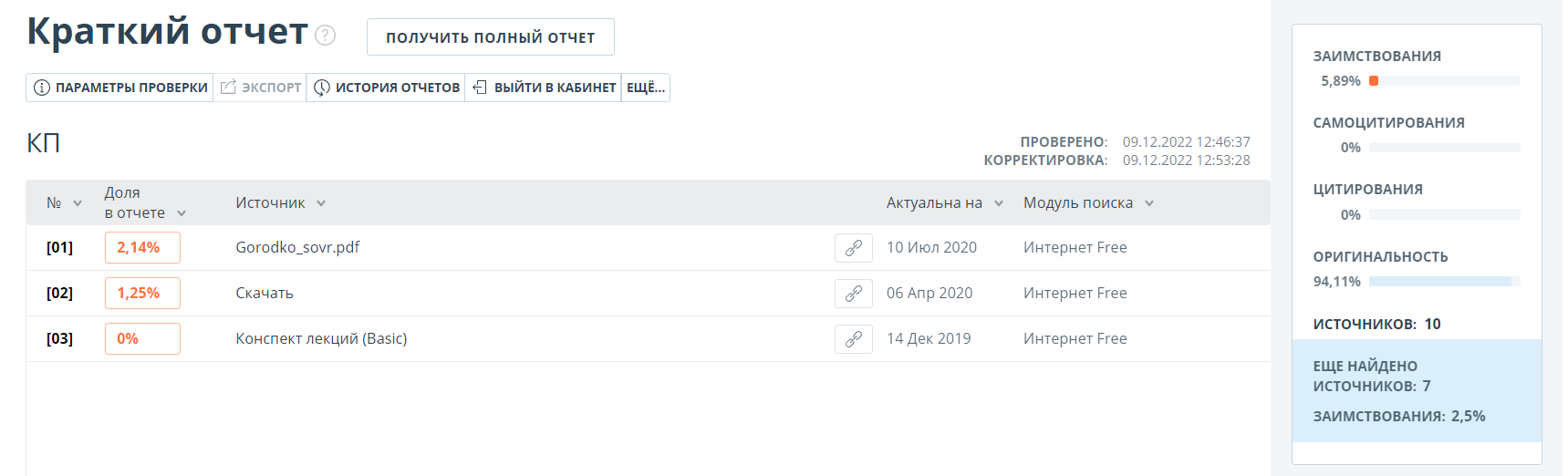
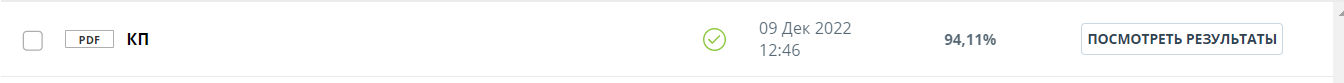


Рисунок А.1 – Отчет о проверке на заимствования в системе «Антиплагиат»

# Приложение Б

# (обязательное)

## Листинг кода алгоритмов, реализующих бизнес-логику

package com.pspkp.kpmain.controllers;

import java.util.NoSuchElementException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.core.annotation.AuthenticationPrincipal;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

import com.pspkp.kpmain.models.Good;

import com.pspkp.kpmain.models.Review;

import com.pspkp.kpmain.models.User;

import com.pspkp.kpmain.repo.GoodRepository;

import com.pspkp.kpmain.repo.ReviewRepository;

import com.pspkp.kpmain.repo.UserRepository;

@Controller

public class ReviewController {

@Autowired

private ReviewRepository reviewRepository;

@Autowired

private GoodRepository goodRepository;

@Autowired

private UserRepository userRepository;

@GetMapping("/good/rank/{id}")

public String rank\_good(@PathVariable(value = "id") Long id, Model model) {

model.addAttribute("id", id);

Good good = goodRepository.findById(id).orElseThrow(NoSuchElementException::new);

model.addAttribute("good", good);

return "rank-good";

}

@PostMapping("/good/rank/{id}")

public String post\_rank\_good(@PathVariable(value = "id") Long id, Model model, @RequestParam Float mark,

@RequestParam String text,

@AuthenticationPrincipal User user) {

if (!goodRepository.existsById(id)) {

return "Error";

}

Good good = goodRepository.findById(id).orElseThrow(NoSuchElementException::new);

int marksamount = good.getMarks\_amount();

float oldmark = good.getMark();

float totalmark = marksamount \* oldmark;

float newmark = (totalmark + mark) / (marksamount + 1);

good.setMarks\_amount(marksamount + 1);

good.setMark(newmark);

user.setPassed(user.getPassed() + 1);

userRepository.save(user);

Review review = new Review(good, mark, text, user);

reviewRepository.save(review);

goodRepository.save(good);

return "redirect:/goods";

}

@GetMapping("/checkreviews")

public String showreviews(Model model) {

Iterable<Review> reviews = reviewRepository.findAll();

model.addAttribute("reviews", reviews);

return "showreviews";

}

@GetMapping("/goods/YNrank/{id}")

public String YNRank(@PathVariable(value = "id") Long id, Model model) {

return "yn-rank-good";

}

@PostMapping("/goods/YNrank/{id}")

public String YNRank(@PathVariable(value = "id") Long id, @RequestParam float question1, float question2,

float question3, float question4, float question5, Model model, @AuthenticationPrincipal User user) {

String[] questions = {

"1. Р’С‹ РґРѕРІРѕР»СЊРЅС‹ РєР°С‡РµСЃС‚РІРѕРј С‚РѕРІР°СЂР°?",

"2. Р’С‹ РїРѕСЃРѕРІРµС‚СѓРµС‚Рµ РєСѓРїР»РµРЅРЅС‹Р№ С‚РѕРІР°СЂ РґСЂСѓРіСѓ?",

"3. РўРѕРІР°СЂ РЅРµ РїСЂРёР±С‹Р» Р±СЂР°РєРѕРІР°РЅРЅС‹Рј?",

"4. РЎРѕРѕС‚РІРµС‚СЃС‚РІСѓРµС‚ Р»Рё С†РµРЅР° С‚РѕРІР°СЂР° РµРіРѕ РєР°С‡РµСЃС‚РІСѓ?",

"5. РЎРѕРѕС‚РІРµС‚СЃРІСѓРµС‚ Р»Рё С‚РѕРІР°СЂ РµРіРѕ РѕРїРёСЃР°РЅРёСЋ?"

};

float[] answers = { question1, question2, question3, question4, question5 };

float mark = 0;

for (float answer : answers) {

mark += answer;

}

String text = "";

for (int i = 0; i < 5; i++) {

text += questions[i] + "<br>";

if (answers[i] == 1.0)

text += "<b>Р”Р°</b><br><br>";

else

text += "<b>РќРµС‚</b><br><br>";

}

Good good = goodRepository.findById(id).orElseThrow(NoSuchElementException::new);

int marksamount = good.getMarks\_amount();

float oldmark = good.getMark();

float totalmark = marksamount \* oldmark;

float newmark = (totalmark + mark) / (marksamount + 1);

good.setMarks\_amount(marksamount + 1);

good.setMark(newmark);

goodRepository.save(good);

user.setPassed(user.getPassed() + 1);

userRepository.save(user);

reviewRepository.save(new Review(good, mark, text, user));

return "redirect:/goods/" + id;

}

}

# Приложение В

# (обязательное)

## Листинг скрипта генерации базы данных

# Ведомость документов курсового проекта