Министерство транспорта Российской Федерации  
Федеральное агентство железнодорожного транспорта  
ФГБОУ ВО «Дальневосточный государственный университет путей   
сообщения»

Кафедра «Информационные технологии  
и системы»

Курсовой проект

по дисциплине «Методы проектирования информационных систем»  
Тема: «Интернет магазин цифровой и бытовой техники»

Выполнил: Ильченко М. А.  
студент гр.СО251КОБ

Проверил: Анисимов В.В.

Хабаровск

2020 г.

Оглавление

[1. Описание предметной области 3](#_Toc54695558)

[2. Модель вариантов использования 4](#_Toc54695559)

[2.1 Диаграммы вариантов использования 4](#_Toc54695560)

[2.2 Диаграммы автоматов 9](#_Toc54695561)

[3. Модель анализа 13](#_Toc54695562)

[3.1 Диаграмма классов анализа 13](#_Toc54695563)

[3.2 Диаграммы последовательности 17](#_Toc54695564)

[3.3 Диаграммы коммуникации 19](#_Toc54695565)

[4. Модель проектирования 23](#_Toc54695566)

[4.1 Диаграммы классов 23](#_Toc54695567)

[4.2 Диаграммы деятельности 29](#_Toc54695568)

[5. Модель реализации 33](#_Toc54695569)

[5.1 Диаграммы компонентов 34](#_Toc54695570)

[5.2 Диаграмма развертывания 37](#_Toc54695571)

[6. Сгенерированный программный код 38](#_Toc54695572)

[7. Руководство пользователя 41](#_Toc54695573)

[7.1 Авторизация 41](#_Toc54695574)

[7.2 Главная страница 41](#_Toc54695575)

[7.3 Процесс работы в системе пользователя 44](#_Toc54695576)

[7.4 Процесс работы контент-мейкера 46](#_Toc54695577)

[7.5 Процесс работы администратора 47](#_Toc54695578)

[Заключение 48](#_Toc54695579)

[Список используемых источников 49](#_Toc54695580)

# 1. Описание предметной области

Целью данного курсового проекта является разработка информационной системы для магазина цифровой и бытовой техники.

В свою очередь основной целью создания системы является: автоматизация человеческой деятельности и продажа товаров и услуг.

Требования заказчика к информационной системе:

* Пользователи должны иметь доступ к приложению средствами браузера (Google Chrome, Mozilla Firefox, Microsoft Edge и другие.)
* Кроссплатформенность (macOS, Windows, Linux);
* Простота в развертывании и администрировании системы;
* Отдельные учётные записи и права доступа для каждого пользователя;

На основе требований заказчика платформой для информационной системы было выбрано веб-приложение с использованием архитектуры MVC. Model-View-Controller (MVC, «Модель-Представление-Контроллер», «Модель-Вид-Контроллер») — схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо.

# 2. Модель вариантов использования

Модель вариантов использования — это модель, описывающая взаимодействие пользователей и системы между собой, для решения поставленных задач. Данная модель описывает цели пользователей, поведение системы, а также взаимодействие пользователей с системой или между собой.

Главной целью при разработке этой модели является достижение максимального взаимопонимания между разработчиками и заказчиками по вопросам назначения, возможностям и технологиям использования ИС.

Достижение этой цели, в первую очередь, достигается за счет разработки диаграмм UML, которые являются основными факторами технологического процесса «Формирование требований».

Унифицированный язык моделирования (UML) в настоящий момент является стандартом де-факто при описании (документирования) результатов проектирования и разработки объектно-ориентированных систем.

Диаграммы, которые включает в себя модель вариантов использования:

* диаграмма вариантов использования;
* диаграмма автоматов.

## 2.1 Диаграммы вариантов использования

Диаграмма вариантов использования (сценариев поведения, прецедентов) является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Диаграмма вариантов использования включает в себя три основных элемента: актеров, варианты использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Актером — это любой объект, субъект или система, взаимодействующая с моделируемой системой извне.

Вариант использования — это спецификация сервисов (функций), которые система предоставляет актеру.

В процессе анализа проектируемой информационной системы было определено четыре актера:

1. Пользователь — лицо, которое производит покупки в интернет-магазине.
2. Контент-мейкер – работник компании, который занимается наполнением сайта (добавляет новые товары и редактирует информацию об уже имеющихся, занимается модерацией комментариев пользователей).
3. Администратор — работник компании, который следит за работоспособностью системы, управлением техническими настройками, занимается непосредственной работой с информационной системой, вносит изменения в работу системы.

Так же определены следующие варианты использования:

1. Регистрация пользователя;
2. Поиск товаров;
3. Добавление товара в корзину;
4. Оплата товаров;
5. Авторизация;
6. Написание отзыва о товаре;
7. Управление учетной записью;
8. Корректировка работы информационной системы;
9. Управление базой данных;
10. Управление настройкой безопасности аккаунтов;
11. Регистрация контент-мейкера;

На основе перечисленных данных была построена контекстная диаграмма, описывающая общую схему взаимодействия актеров в пределах ИС (рис. 2.1.1).



Рисунок 2.1.1. Контекстная диаграмма вариантов использования

Основываясь на контекстную диаграмму, были построены три диаграммы декомпозиций.

Обычно в центре диаграммы декомпозиции располагается декомпозируемый вариант использования, а вокруг – входящие в него обязательные (««include»») или расширяющие (««extend»») составные части.

Рассмотрим декомпозицию варианта использования «Регистрация пользователя» (рис. 2.1.2), на которой в качестве «центрального» актера выступает «Пользователь». В данном разделе пользователь может выполнять следующие действия:

1. Заполнение логина;
2. Заполнение пароля;
   1. Генерация случайного пароля;
3. Заполнение ФИО;
4. Заполнение номера телефона;
5. Заполнение адреса электронной почты;
6. Заполнение даты рождения;
   1. Выбор даты из календаря.

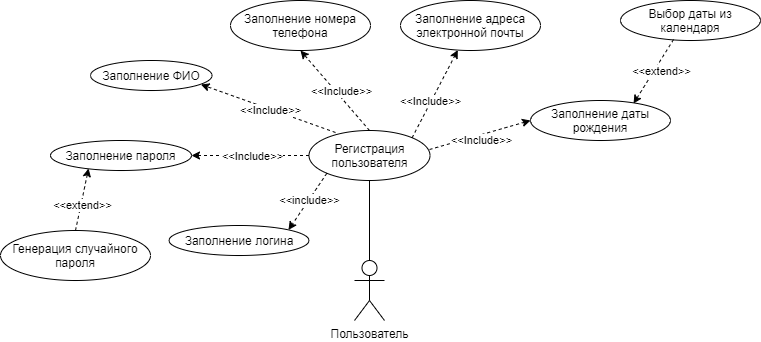


Рисунок 2.1.2. Диаграмма декомпозиции "Регистрация пользователя"

На диаграмме декомпозиции варианта использования «Написание отзыва о товаре» (рис. 2.1.3) показаны основные возможности пользователя в данном разделе:

1. Выбор товара;
2. Указание статуса рекомендации;
3. Заполнение текста отзыва;
   1. Ответ на типовые вопросы;
   2. Написание текста отзыва самостоятельно.

Для выполнения каких-либо манипуляций в рамках данной диаграммы пользователю необходимо успешно авторизоваться в информационной системе.

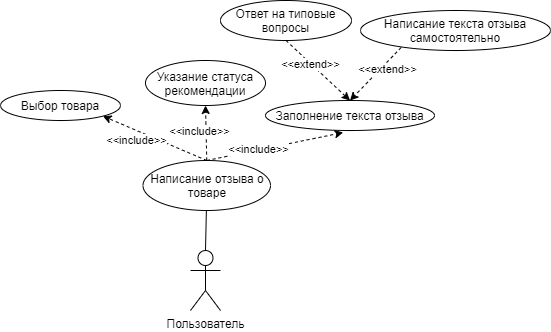


Рисунок 2.1.3. Диаграмма декомпозиции "Написание отзыва о товаре"

На диаграмме декомпозиции варианта использования «Создание товара» (рис. 2.1.4) показаны основные возможности контент-мейкера в данном разделе:

1. Заполнение названия
2. Заполнение категории товара:
   1. Выбор категории из списка доступных;
   2. Добавление новой категории;
3. Заполнение характеристик;
4. Заполнение цены;
5. Заполнение компании производителя;
   1. Выбор компании из списка доступных;
   2. Добавление новой компании.

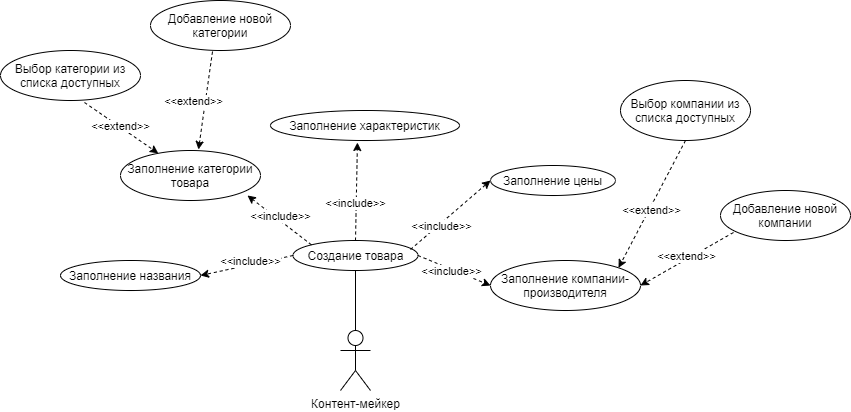


Рисунок 2.1.4. Диаграмма декомпозиции "Создание товара"

## 2.2 Диаграммы автоматов

Диаграммы автоматов (англ. state machine) используются для описания поведения, реализуемого в рамках варианта использования, или поведения экземпляра сущности (класса, объекта, компонента, узла или системы в целом). Поведение моделируется через описание возможных состояний экземпляра сущности и переходов между ними на протяжении его жизненного цикла, начиная от создания и заканчивая уничтожением. Диаграмма автоматов представляет собой связный ориентированный граф, вершинами которого являются состояния, а дуги служат для обозначения переходов из состояния в состояние.

Под состоянием (англ. state) понимается ситуация в ходе жизни экземпляра сущности, когда эта ситуация удовлетворяет некоторому условию, экземпляр выполняет некоторые операции или ждет наступления некоторого события.

Дуги графа служат для обозначения переходов из состояния в состояние. Диаграммы автоматов могут быть вло­жены друг в друга, образуя вложенные диаграммы более детального пред­ставления отдельных элементов модели.

В связи с тем, что к данной системе будут иметь доступ разные пользователи, которые соответственно обладают разными правами, то у нас появляется необходимость для каждого типа пользователей организовать индивидуальный интерфейс с определёнными правами именно для данного типа. Эта ситуация отражена на контекстной диаграмме автоматов (рис. 2.2.1).

В данном случае, мы видим состояния системы, при ее использовании пользователями, имеющими разные роли, и как следствие, типы доступа к компонентам системы.

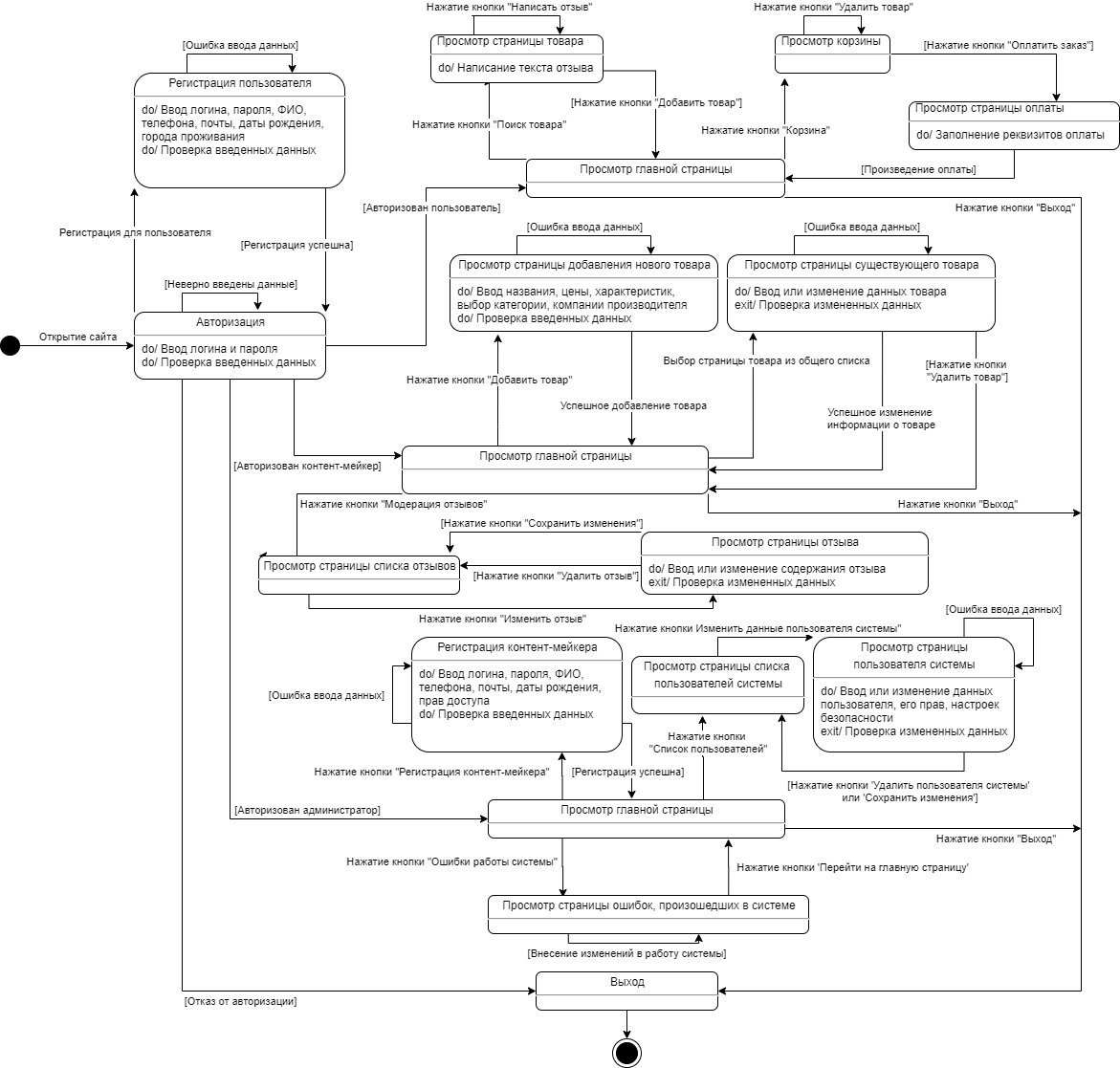


Рисунок 2.2.1. Контекстная диаграмма автоматов.

На диаграмме автоматов подсистемы «Регистрация пользователя» (рис. 2.2.2) отображены состояния системы при управлении системой пользователем. Диаграмма более подробно детализирует вариант использования «Регистрация пользователя». Сценарий «Регистрация пользователя» следующий: пользователь переходит на страницу «Регистрация пользователя», где при ее загрузке, происходит инициализация формы для заполнения данных для регистрации. Далее пользователь должен заполнить поля формы. Такие поля, как логин и пароль проходят обязательные проверки (логин – на уникальность, пароль – на соответствие требованиям политики безопасности). В случае неудачного прохождения проверки пользователь получает сообщение об этом и должен ввести эти данные снова. После успешного заполнения всех полей необходимо нажать кнопку «Зарегистрировать пользователя». Далее происходит обращение к серверу и добавление записи в базу данных.

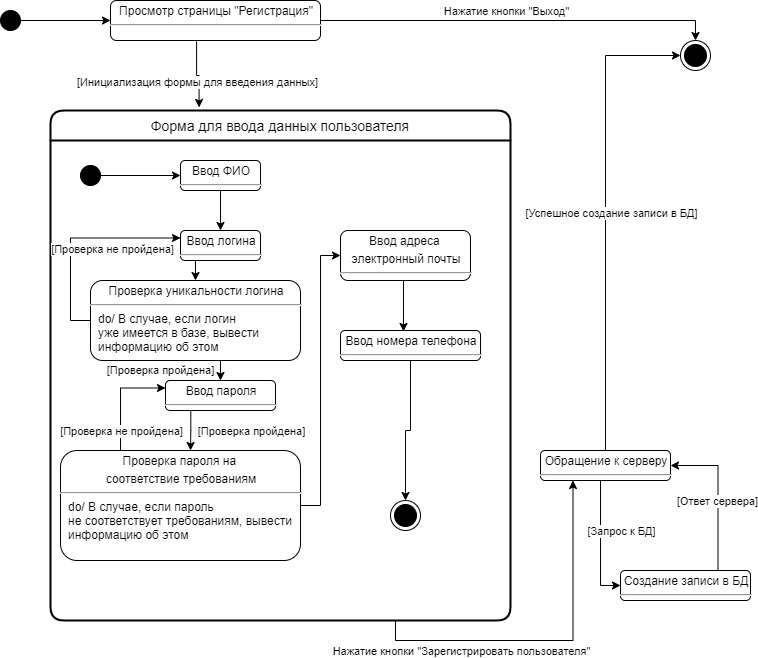


Рисунок 2.2.2 Диаграмма автоматов «Регистрация пользователя»

На диаграмме автоматов подсистемы «Добавление товара» (рис. 2.2.3) отображены состояния системы при управлении системой контент-мейкером. Диаграмма более подробно детализирует вариант использования «Добавление товара». Сценарий «Добавление товара» следующий: контент-мейкер переходит на страницу «Добавление товара», где при ее загрузке, происходит инициализация формы для заполнения данных для добавления нового товара. Далее контент-мейкер должен заполнить поля формы. Данные в поле “Название” проходят проверку на уникальность. Если проверка не была пройдена, то выводится информация об этом, это поле нужно заполнить снова. После успешного заполнения всех полей необходимо нажать кнопку «Добавить товар». Далее происходит обращение к серверу и добавление записи в базу данных.

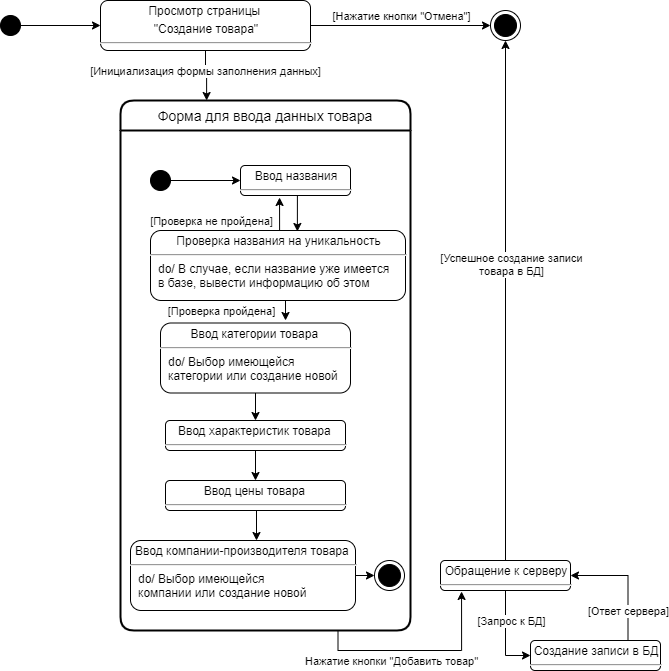


Рисунок 2.2.3 Диаграмма автоматов "Добавление товара"

# 3. Модель анализа

Главная цель построения модели анализа заключается в уточнении вариантов использования с учетом внутренней архитектуры проектируемой системы.

Построение этой модели необходимо:

* для того чтобы выявить внутренние архитектуры, то есть определить основные классы и подсистемы;
* для поиска альтернативных вариантов реализации системы (подсистем) и выбора основного;
* для уточнения всех требований (функциональных и нефункциональных).

При разработке модели анализа строятся следующие диаграммы:

* классов анализа;
* последовательности;
* коммуникации.

## 3.1 Диаграмма классов анализа

Класс анализа – это укрупненная абстракция, которая на концептуальном уровне (без точного определения атрибутов и операций) описывает некоторый фрагмент системы.

Существует три вида классов анализа:

* граничный;
* управляющий;
* сущности.

Диаграмма классов анализа является прообразом классической диаграммы классов. Элементами, отображаемыми на диаграмме, являются классы и отношения между ними.

Назначение классов анализа:

* граничный класс – используется для моделирования взаимодействия между системой и актерами (пользователями, внешними системами или устройствами);
* управляющий класс – отвечает за координацию, взаимодействие и управление другими объектами, выполняет сложные вычисления, управляет безопасностью, транзакциями и т. п.;
* класс сущности – используется для моделирования долгоживущей, нередко сохраняемой информации. Классы сущности являются абстракциями основных понятий предметной области – людей, объектов, документов и т. д., как правило, хранимых в табличном или ином виде.

Связи между классами анализа отображаются с использованием отношений пяти видов:

* ассоциаций – показывает, что объекты одного класса содержат информацию о существовании (наличии в памяти) объектов другого класса и между ними имеется некоторая логическая или семантическая связь;
* агрегаций – указывает на отношение «часть–целое» и отображается сплошной линией с не закрашенным ромбиком со стороны «целого»;
* композиций – аналогично агрегации, в которой «части» не могут существовать отдельно от «целого»;
* обобщения – является обычным таксонометрическим отношением между более общим (абстрактным) классом (родителем или предком) и его частным случаем (дочерним классом или потомком);
* зависимостей – означает, что в спецификации или теле методов объектов одного класса (зависимого) выполняется обращение к атрибутам, методам или непосредственно к объектам другого класса.

Диаграмма классов анализа изображена на рисунке 3.1.1.

На диаграмме в виде классов отображено клиент-серверное взаимодействие между клиентским приложением и сервером. Соединение обеспечивает некоторый управляющий класс, который позволяет всем граничным классам приложения получать информацию, которая содержится на сервере, и наоборот, отправлять информацию на сервер для дальнейшей ее обработки. Все граничные классы представляют собой View-компоненты, которые отображаются на странице. В свою очередь все сущности представляют собой таблицы в базе данных.

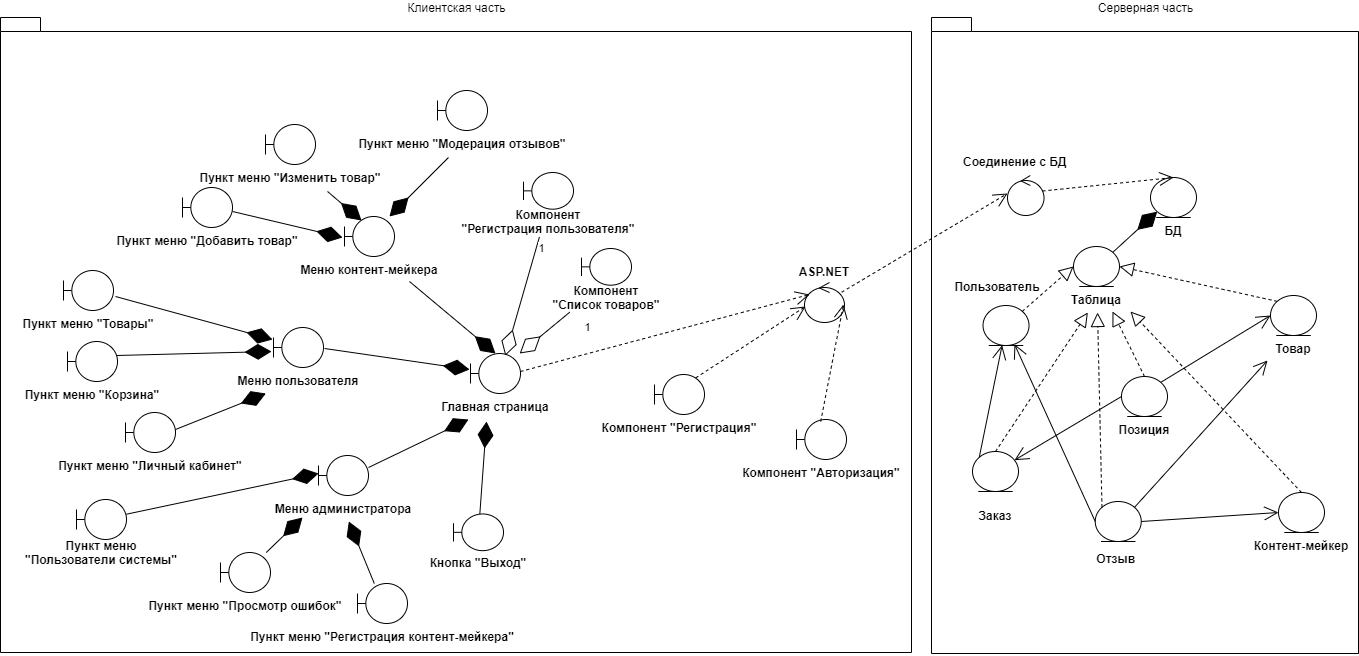


Рисунок 3.1.1 Диаграмма классов анализа

## 3.2 Диаграммы последовательности

Диаграмма последовательности – это одна из разновидностей диаграмм взаимодействия, её назначение заключается в моделирование взаимодействий различных объектов системы во времени, а также в обмене сообщениями между этими объектами.

На диаграмме последовательности изображаются объекты, которые в основном представляют экземпляры класса или сущности, обладающие поведением. Объектами на диаграмме последовательности могут быть пользователи, классы, программные компоненты, а иногда и системы в целом.

Диаграмма последовательности наглядно отображает временной аспект взаимодействия. Она имеет два измерения. Одно измерение (слева-направо) указывает на порядок вовлечения экземпляров сущностей во взаимодействие. Крайним слева на диаграмме отображается экземпляр актера или объект, который является инициатором взаимодействия. Правее отображается другой экземпляр сущности, который непосредственно взаимодействует с первым и т.д. Второе измерение (сверху-вниз) указывает на порядок обмена сообщениями. Начальному моменту времени соответствует самая верхняя часть диаграммы. Масштаб на оси времени не указывается, поскольку диаграмма отображает лишь временную упорядоченность взаимодействия типа «раньше-позже».

Для проектируемой информационной системы были построены две диаграммы последовательностей.

На рисунке 3.2.1 представлена диаграмма последовательности «Добавление товара». Для того чтобы попасть на страницу «Добавление товара» контент-мейкеру необходимо выбрать советующий пункт. После чего произойдет получение записей из БД, формирование страницы на сервере и отправка её клиенту.

Далее контент-мейкер должен заполнить все поля. Поля “Название” и “Категория” проходят проверку. Далее необходимо заполнить все остальные поля (“Характеристики”, “Цена”, “Пользователь”). После этого по нажатию кнопки происходит добавление записи в базу данных.

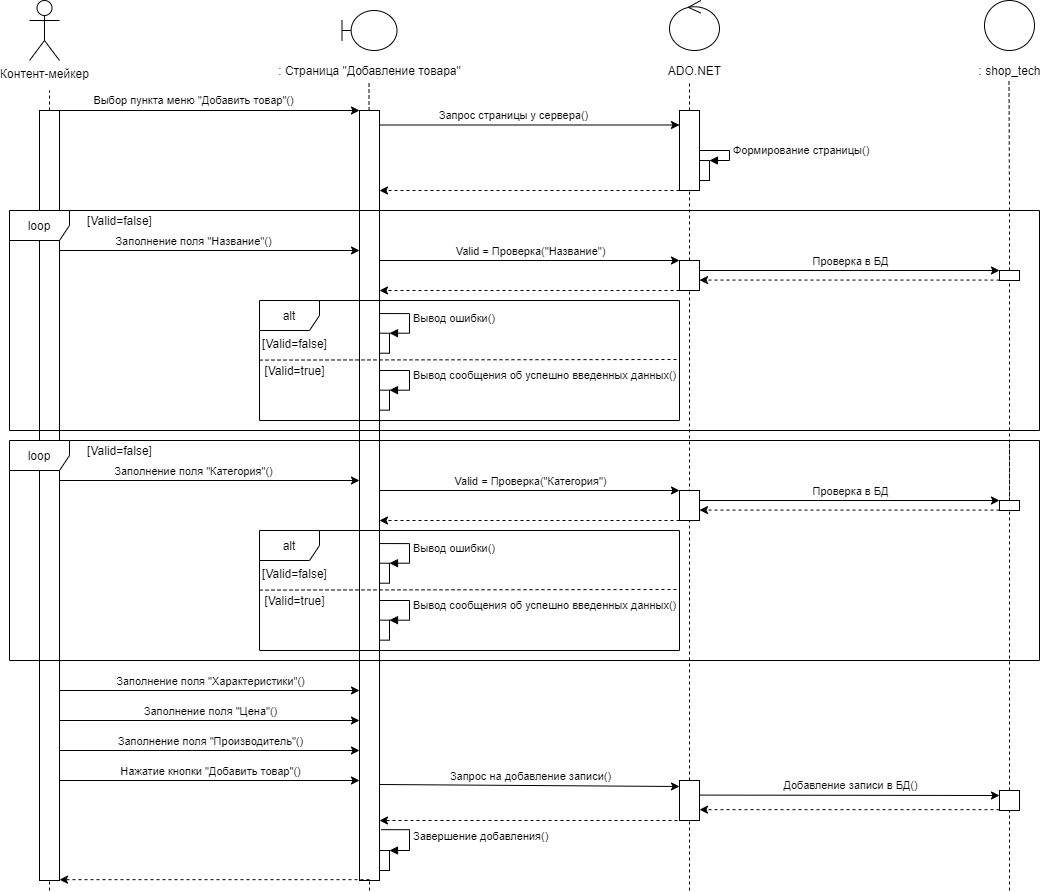


Рисунок 3.2.1. Диаграмма последовательности «Добавление товара»

На рисунке 3.2.2 представлена диаграмма последовательности «Регистрация пользователя». Для того, чтобы зарегистрироваться пользователю необходимо выбрать соответствующий пункт меню. По нажатию кнопки происходит загрузка формы. После этого необходимо последовательно заполнить все поля. Данные в поле “Логин” проходят проверку на уникальность в базе данных. Данные в поле “Пароль” проходят проверку на соответствие требованиям политики безопасности. После того, как все поля будут заполнены, по нажатию кнопки “Зарегистрировать пользователя” отправляется запрос на добавление записи в базу данных.

.

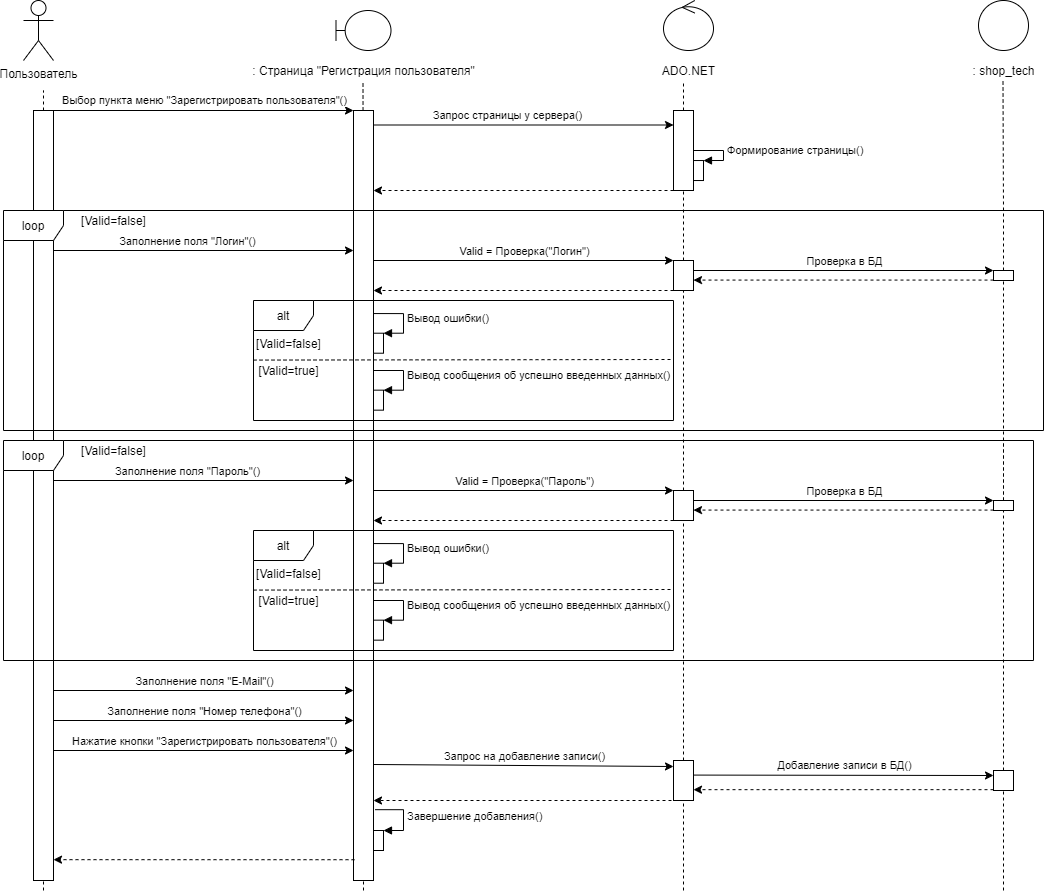


Рисунок 3.2.2. Диаграмма последовательности "Регистрация пользователя"

## 3.3 Диаграммы коммуникации

В отличие от диаграммы последовательности на диаграмме коммуникации основное внимание уделяется структуре взаимодействия. Помимо общих элементов (экземпляров актеров, объектов и сообщений) между участниками взаимодействия отображаются ненаправленные ассоциации, над которыми указываются передаваемые ими сообщениями. Другой отличительной особенностью является использование в спецификации сообщений нумерации, отражающей порядок их выполнения.

Проектировщикам диаграмма коммуникации может дать богатый материал о распределении обязанностей между объектами. Так, например, если диаграмма напоминает форму звезды, то можно сделать вывод, что система сильно зависит от центрального объекта. В этом случае стоит подумать о более равномерном распределении обязанностей между участниками взаимодействия. Или, наоборот, если в системе хранится и обрабатывается конфиденциальная информация, то большинство сообщений должно проходить через ядро безопасности – классы, отвечающие за идентификацию, аутентификацию и, возможно, шифрование / расшифрование данных.

Таким образом, цель самой коммуникации состоит в том, чтобы специфицировать особенности реализации отдельных наиболее значимых операций в системе. Коммуникация определяет структуру поведения системы.

На рисунке 3.3.1 показана диаграмма коммуникации «Добавление товара». Данная диаграмма сгенерирована автоматически на основе диаграммы последовательности, изображенной на рисунке 3.2.1.

На рисунке 3.3.2 показана диаграмма коммуникации «Регистрация пользователя». Данная диаграмма сгенерирована автоматически на основе диаграммы последовательности, изображенной на рисунке 3.2.2.

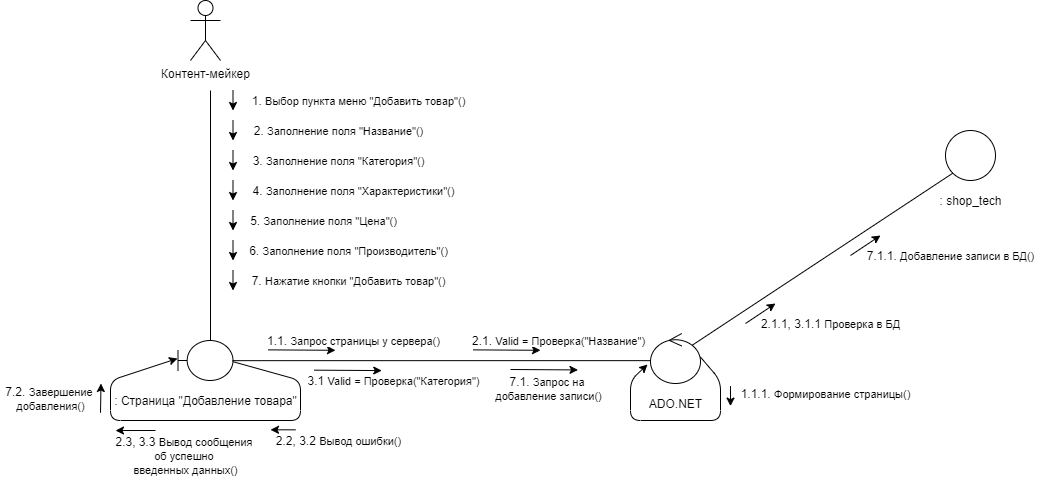


Рисунок 3.3.1. Диаграмма коммуникации «Добавление товара»

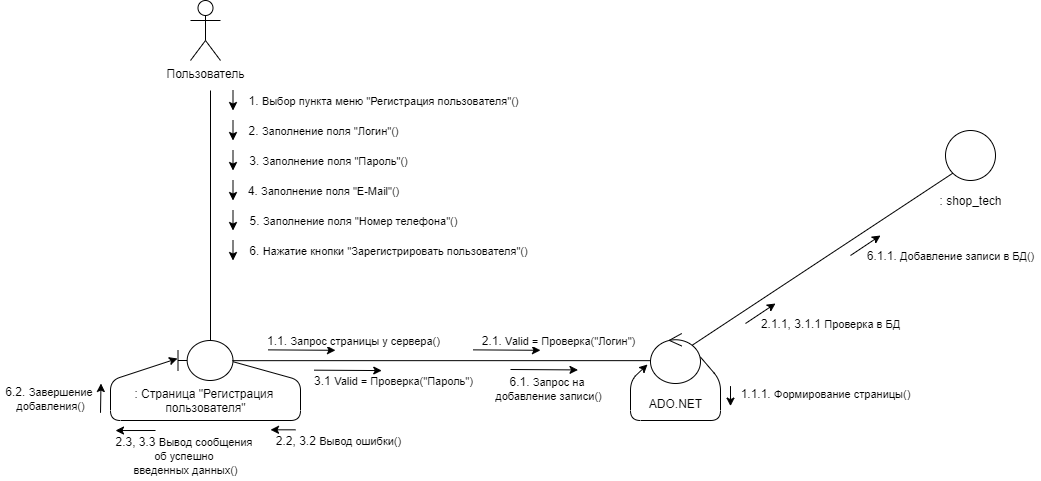


Рисунок 3.3.2. Диаграмма коммуникации «Регистрация пользователя»

# 4. Модель проектирования

В процессе проектирования создается архитектура системы, которая позволит реализовать и затем поддерживать все функции информационной системы.

Назначение модели проектирования заключается в создании полного детализированного описания внутренней архитектуры и алгоритмов работы системы.

Рекомендуется разрабатывать данную модель без привязки к конкретным языкам программирования, с помощью которых будет создаваться программный продукт, т. е. разрабатывать логическую модель.

Стоит оговориться, что создать модель без оглядки на используемые языки программирования невозможно, но, по крайней мере, необходимо стремиться к этому.

Построение модели проектирования необходимо:

* для уточнения внутренней архитектуры и вариантов использования системы;
* уточнения требований;
* определения детализированных алгоритмов работы системы в целом и ее отдельных элементов.

Модель проектирования представляется диаграммами классов и диаграммами деятельности.

## 4.1 Диаграммы классов

Диаграммы классов используются при моделировании информационных систем наиболее часто. Они являются одной из форм статического описания системы с точки зрения ее проектирования, показывая ее структуру. Диаграмма классов не отображает динамическое поведение объектов, изображенных на ней классов.

На диаграммах классов показываются классы, интерфейсы и отношения между ними.

Диаграмма классов представляет собой граф, вершинами которого являются элементы типа «классификатор», связанные различными типами структурных отношений.

Существуют разные точки зрения на построение диаграмм классов в зависимости от целей их применения:

1. концептуальная точка зрения – диаграмма классов описывает модель предметной области, в ней присутствуют только классы прикладных объектов;
2. точка зрения спецификации – диаграмма классов применяется при проектировании информационных систем;
3. точка зрения реализации – диаграмма классов содержит классы, используемые непосредственно в программном коде.

Классы могут иметь логическую и физическую реализации. Логические диаграммы классов в отличие от физических, строятся без привязки к языкам программирования.

В ходе выполнения курсового проекта было разработано два типа диаграмм классов: для клиентского приложения и для серверного приложения.

Прежде чем перейти к разработке этих двух моделей, опишем структуру таблиц базы данных:

Таблица 1. User (пользователь)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер пользователя | int | Уникальный идентификатор пользователя |
| Логин | varchar(15) | Уникальное название аккунта пользователя |
| Пароль | varchar(15) | Комбинация символов, которую знает только пользователь |
| Фамилия | varchar(15) | Фамилия пользователя |
| Имя | varchar(15) | Имя пользователя |
| Отчество | varchar(15) | Отчество пользователя |
| Адрес электронной почты | varchar(15) | Адрес электронной почты пользователя |
| Номер | varchar(15) | Номер телефона пользователя |

Таблица 2. Заказ (Order)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер заказа | int | Уникальный идентификатор заказа |
| Номер корзины | int | Корзина, по которой производится оплата |
| Сумма | money | Общая сумма заказа |
| Статус оплаты | bit | Была ли произведена оплата |
| Номер пользователя | int | Номер пользователя, который оформил заказ |
| Реквизиты оплаты | varchar(15) | Реквизиты для оплаты заказа |
| Дата заказа | datetime | Дата и время оформления заказа |

Таблица 3. Позиция (Position)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер позиции | int | Уникальный идентификатор позиции |
| Номер товара | int | Номер товара, добавленного в корзину |
| Номер пользователя | int | Номер пользователя, который собирал корзину |
| Количество | int | Количество единиц товара в позиции |

Таблица 4. Товар (Product)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер товара | int | Уникальный идентификатор товара |
| Название товара | varchar(15) | Название товара в каталоге |
| Категория | int | Номер группы в справочнике, к которой относится товар |
| Характеристики | text | Все характеристики, особенности товара |
| Цена | money | Стоимость товара |
| Производитель | int | Номер в справочнике компании, которая произвела торвар |

Таблица 5. Отзыв (Review)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер отзыва | int | Уникальный идентификатор отзыва |
| Номер товара | int | Номер товара, на который написан отзыв |
| Номер пользователя | int | Номер пользователя, написавшего обзор |
| Дата написания | datetime | Дата и время написания отзыва |
| Текст | text | Текст написанного отзыва |
| Статус рекомендации | bit | Показатель отношения пользователя к товару (рекомендую/не рекомендую) |
| Дата последнего изменения | datetime | Дата и время последнего внесенного изменения отзыва контент-мейкером |
| Номер контент-мейкера | int | Номер контент-мейкера, внесшего последние изменения |

Таблица 6. Контент-мейкер (Content maker)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер контент-мейкера | int | Уникальный идентификатор контент-мейкера |
| Логин | varchar(15) | Уникальное название аккунта контент-мейкера |
| Пароль | varchar(15) | Комбинация символов, которую знает только контент-мейкер |
| Фамилия | varchar(15) | Фамилия контент-мейкера |
| Имя | varchar(15) | Имя контент-мейкера |
| Отчество | varchar(15) | Отчество контент-мейкера |
| Адрес электронной почты | varchar(15) | Адрес электронной почты контент-мейкера |
| Товары | bit | Права доступа контент-мейкера для работы с товарами |
| Отзывы | bit | Права доступа контент-мейкера для работы с отзывами |

Таблица 7. Категория (Category)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер категории | int | Уникальный идентификатор категории |
| Название | varchar(15) | Название категории |
| Описание | text | Основные параметры товаров, входящих в категорию, их назначение, применение |

Таблица 8. Производитель (Manufacturer)

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| Номер производителя | int | Уникальный идентификатор производителя |
| Название | varchar(15) | Название категории |
| Примечание | text | Важные сведения о производителе |

Далее приведены диаграммы классов для БД (рис. 4.1.1 и рис. 4.1.2) и диаграммы классов для приложения (рис. 4.1.3 и рисунок 4.1.4). Каждый вид диаграмм представлен в двух экземплярах: логическом (на русском языке) и физическом (с учетом языка программирования).

Целевой СУБД для описания диаграмм классов БД, является MS SQL Server.

Целевым языком программирования для описания диаграмм классов приложения является С# (ADO.NET).

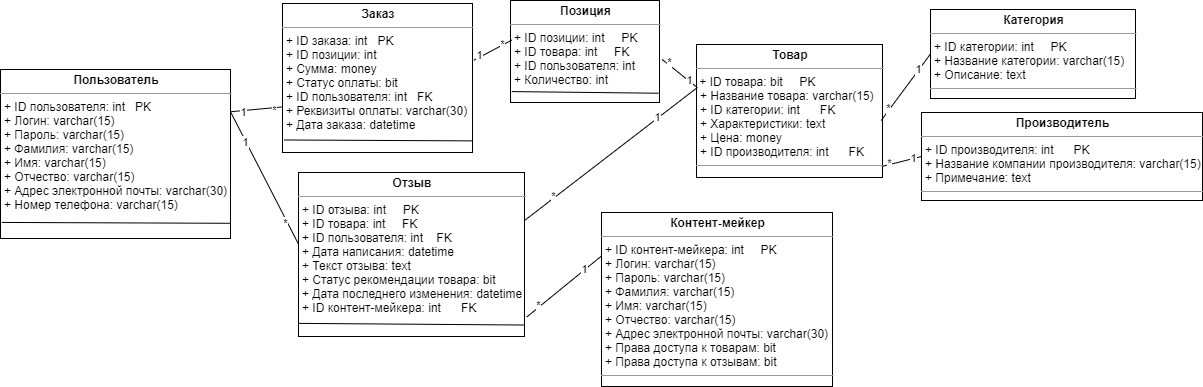


Рисунок 4.1.1. Логическая диаграмма классов базы данных

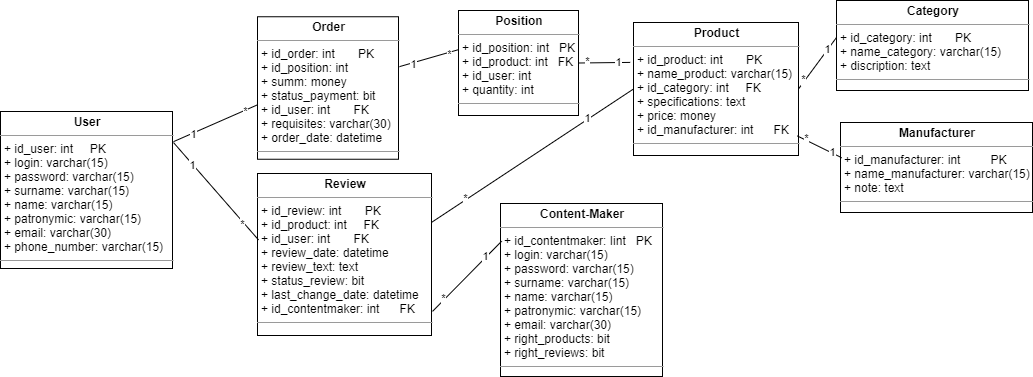


Рисунок 4.1.2. Физическая диаграмма классов базы данных

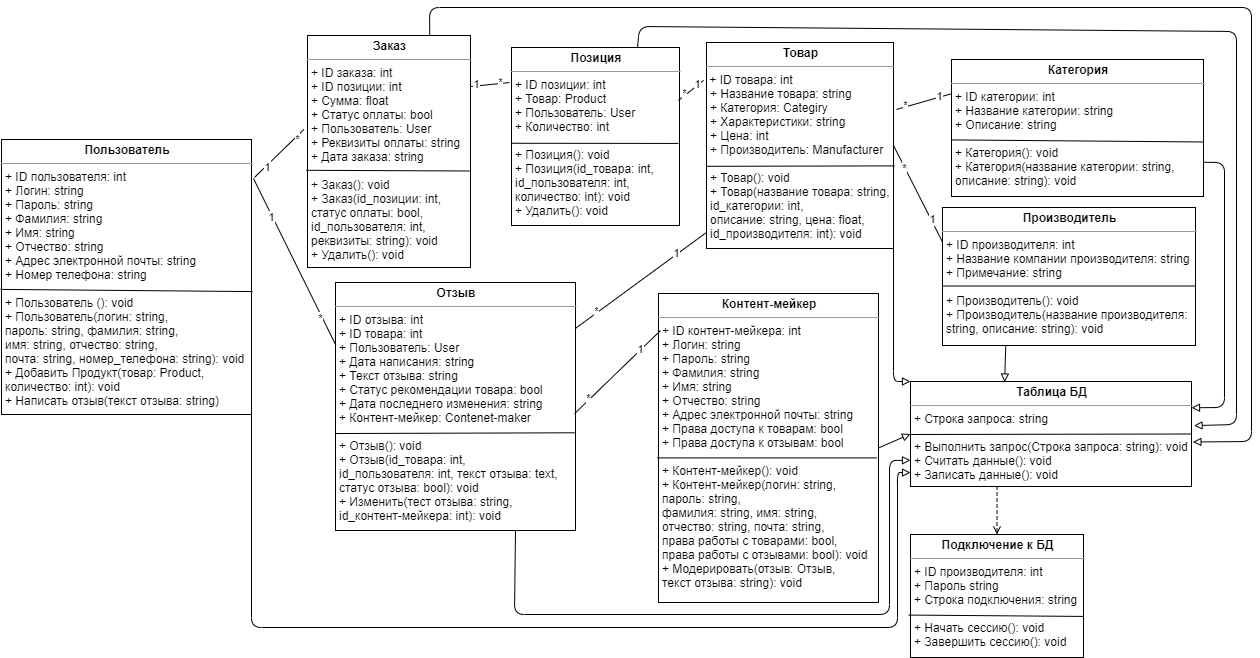


Рисунок 4.1.3. Логическая диаграмма классов приложения

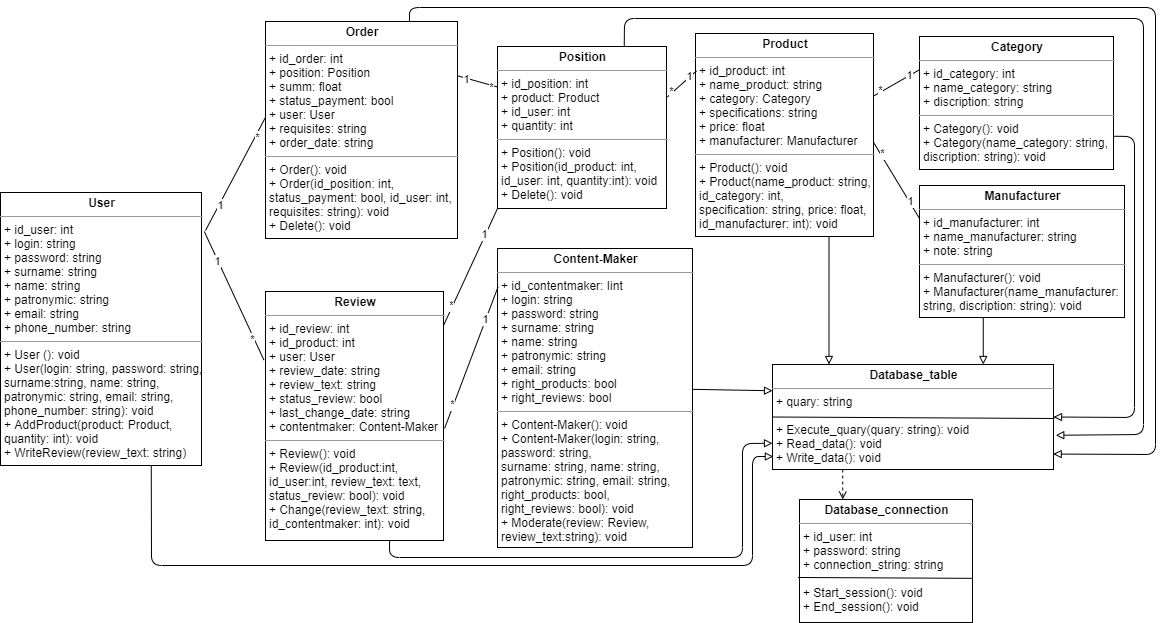


Рисунок 4.1.4. Физическая диаграмма классов приложения

## 4.2 Диаграммы деятельности

При моделировании поведения системы возникает необходимость не только представить процесс изменения ее состояний, но и детализировать особенности алгоритмической и логической реализации выполняемых системой операций.

Для описания поведения системы и ее отдельных элементов (поведенческих моделей) в UML предусмотрено четыре вида диаграмм:

* диаграммы автоматов;
* диаграммы последовательности;
* диаграммы коммуникации;
* диаграммы деятельности.

Несмотря на то, что первые три вида диаграмм, так или иначе, отображают динамические аспекты системы, они недостаточно формальны для детального описания алгоритмов работы. В структурном подходе для этого применяются блок-схемы, диаграммы EPC и BPMN. В UML аналогом блок-схем являются диаграммы деятельности (активности), схожие с ними по своей семантике и выразительным средствам (набору элементов).

Каждая диаграмма деятельности акцентирует внимание на последовательности выполнения определенных действий, которые в совокупности приводят к получению желаемого результата. Они могут быть построены для отдельного варианта использования, кооперации, метода и т. д. Диаграммы деятельности являются разновидностью диаграмм автоматов, но если на второй основное внимание уделяется статическим состояниям, то на первой – действиям.

Графически диаграмма деятельности, как и диаграмма автоматов, представляется в виде ориентированного графа, вершинами которого являются действия или деятельности, а дугами – переходы между ними. При этом в UML действие – это атомарная операция, выполнение которой не может быть прервано, а деятельность – составная операция, с возможностью ее прерывания. Переход к следующему действию или деятельности срабатывает сразу по их завершении.

Основными элементами диаграммы являются:

* исполняемые узлы – к исполняемым узлам (англ. executable nodes) относятся действия (англ. action) и деятельности (англ. activity);
* объекты – к объектам относятся непосредственно объекты (англ. object) в традиционном понимании UML, отправка сигнала (англ. send signal), прием сигнала (англ. accept signal) и событие времени (англ. time event);
* переходы – переход (англ. transition или activity edge), как и на диаграмме автоматов, отображается ассоциацией. На диаграммах деятельности различают следующие виды переходов:
  + поток управление;
  + объектный поток;
  + поток прерывания;
  + поток исключения.
* управляющие узлы – управляющим узлам (англ. control nodes) на диаграмме деятельности соответствуют псевдосостояния на диаграмме автоматов;
* коннекторы – коннекторы (англ. connectors) выступают в качестве соединителей, применяемых на блок-схемах;
* группирующие элементы – к группирующим элементам (англ. activity groups) относятся разделы деятельности (англ. activity partitions) и прерываемые регионы (англ. interruptible activity regions).

На рисунке 4.2.1 изображена диаграмма деятельности, описывающая процесс добавление нового товара. На ней наглядно видно взаимодействие между разделами деятельности.

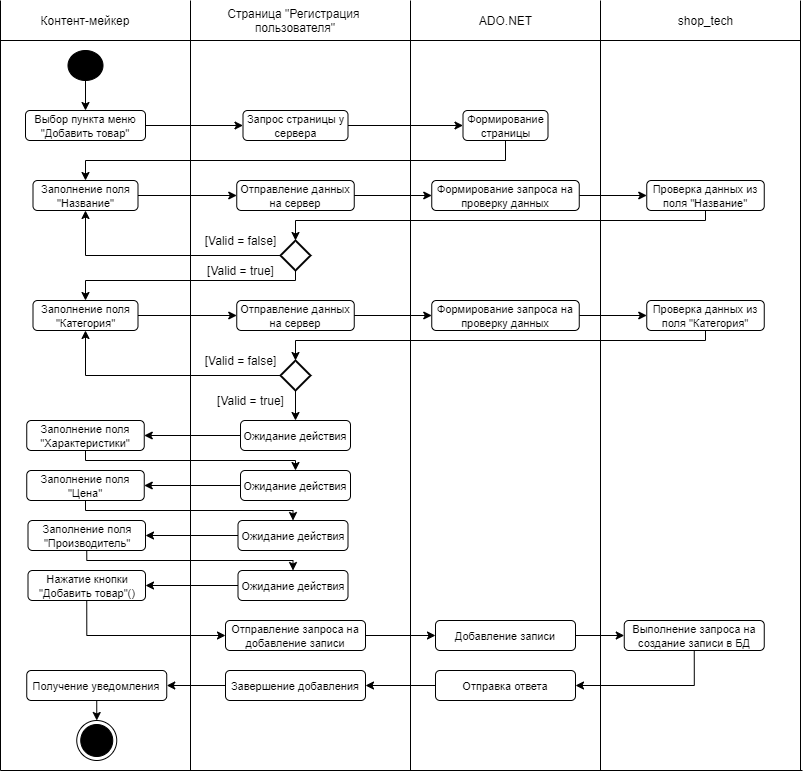


Рисунок 4.2.1. Диаграмма деятельности, описывающая процесс добавление нового товара

На рисунке 4.2.2 изображена диаграмма деятельности, описывающая процесс регистрации пользователя. На ней наглядно видно взаимодействие между разделами деятельности.

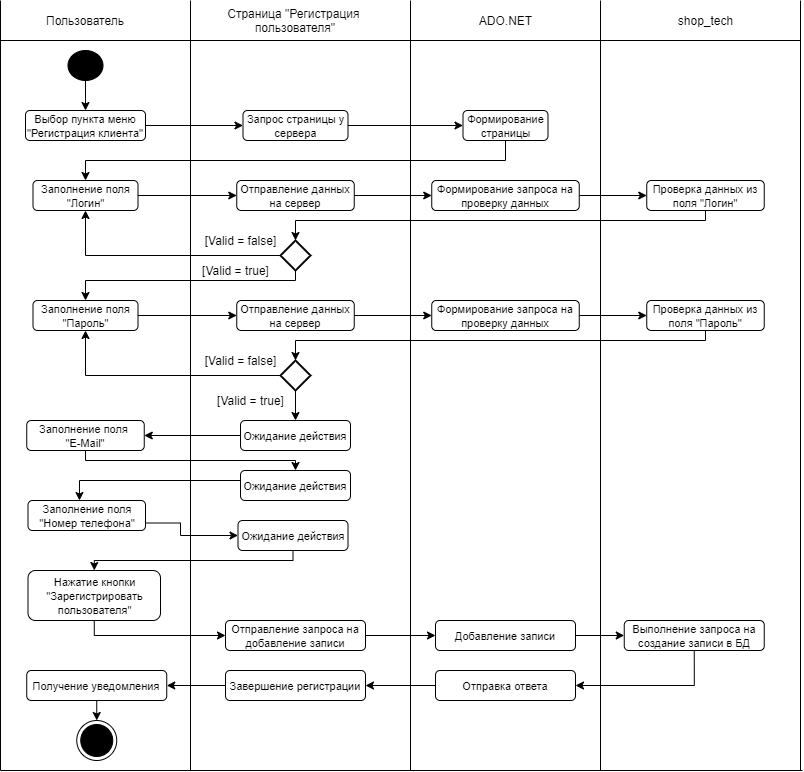


Рисунок 4.2.2. Диаграмма деятельности, описывающая процесс регистрации пользователя

# 5. Модель реализации

Модель реализации представляет физическую структуру реализации в терминах подсистем реализации и элементов реализации (каталогов и фалов, включая файлы исходного кода, файлы данных и исполняемые файлы и т.д).

Модель реализации идентифицирует физические компоненты реализации с целью облегчения их восприятия и управления ними.

Модель реализации определяет основные блоки интеграции, вокруг которых организованы рабочие группы, а также блоки, которым можно по отдельности присваивать версии, отдельно развертывать и заменять.

Модель реализации представляется диаграммами компонентов и развертывания.

Таким образом, при разработке модели преследуются цели:

* определение окончательного состава, структуры и кода классов;
* распределение классов по компонентам и подсистемам;
* определение топологии распределенной системы и распределение подсистем по узлам сети;
* планирование итераций (версий) сборки системы;
* сборка версий системы.

При разработке модели реализации рекомендуется построить диаграммы:

* компонентов;
* развертывания.

## 5.1 Диаграммы компонентов

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.

Диаграмма компонентов разрабатывается для следующих целей:

* визуализации общей структуры исходного кода программной системы;
* спецификации исполняемого варианта программной системы;
* обеспечения многократного использования отдельных фрагментов программного кода;
* представления концептуальной и физической схем баз данных.

В языке UML для компонентов определены следующие стереотипы:

* «file» – любой файл, кроме таблицы;
* «executable» – программа (исполняемый файл);
* «library» – статическая или динамическая библиотека;
* «source» – файл с исходным текстом программы;
* «document» – остальные файлы (например, файл справки);
* «table» – таблица базы данных.

На рисунке 5.1.1 показана диаграмма компонентов. На рисунке 5.1.2 показана диаграмма компонентов серверного приложения.

На данной диаграмме отражены основные элементы разрабатываемой информационной системы, а также основной исполняемый файл, файлы компонентов клиентского приложения, служебные файлы с дополнительной информацией, подключаемые библиотеки.

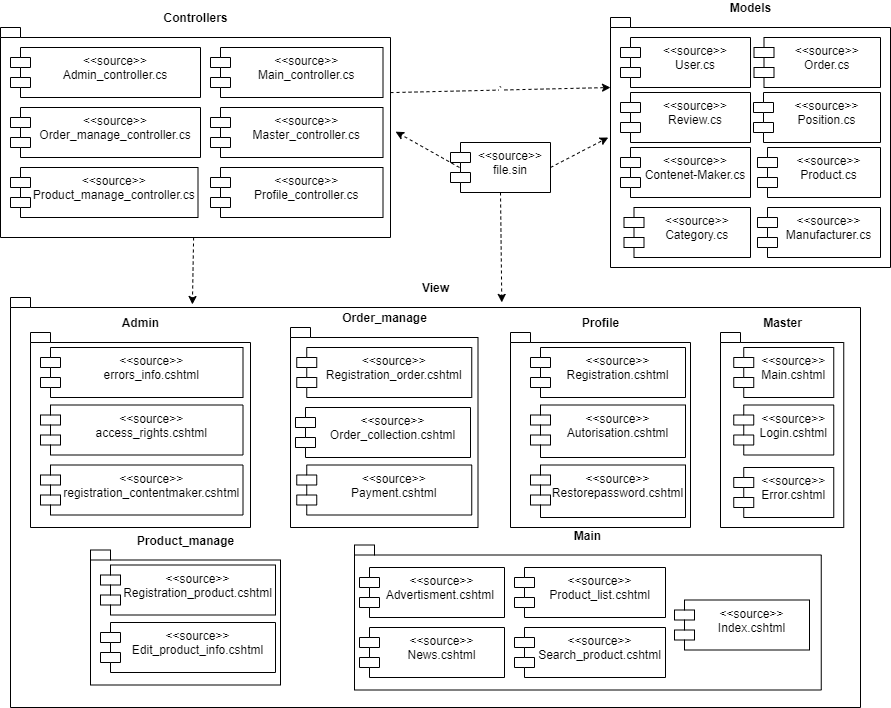


Рисунок 5.1.1. Диаграмма компонентов клиентского приложения

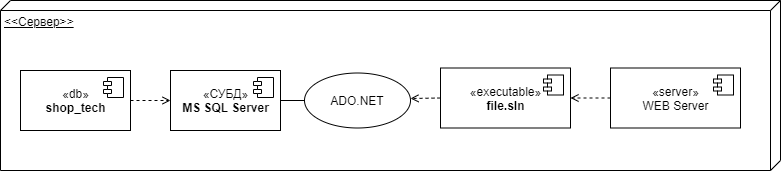


Рисунок 5.1.2. Диаграмма компонентов клиентского серверного приложения

## 5.2 Диаграмма развертывания

Целью диаграммы развертывания является представление физического расположения системы, взаимодействия физического оборудования, на котором запускается та или иная составляющая программного обеспечения.

Основные цели, преследуемые при разработке диаграммы развертывания:

* распределение компонентов системы по ее физическим узлам;
* отображение физических связей между узлами системы на этапе исполнения;
* выявление узких мест системы и реконфигурация ее топологии для достижения требуемой производительности.

Элементами диаграммы реализации являются узлы, компоненты и связи между ними.

На рисунке 5.2.1 представлена диаграмма развертывания, которая отражает физические компоненты информационной системы.

Для работы с информационной системой, на машинах пользователей должен быть установлен браузер. Само приложение и база данных устанавливаются на сервер.

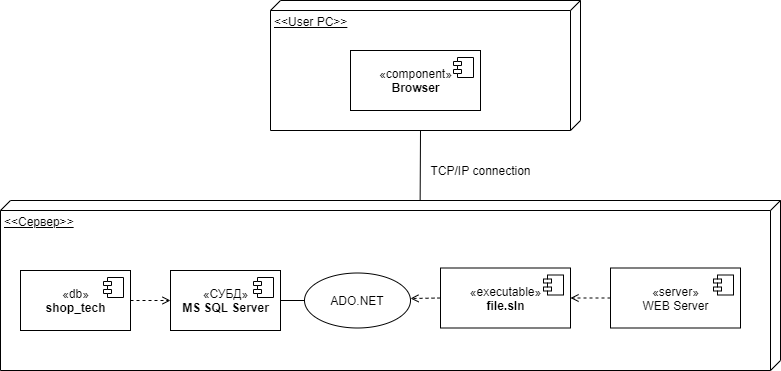


Рисунок 5.2.1 Диаграмма развертывания ИС

# 6. Сгенерированный программный код

Средствами программного продукта АSTAH-PROFESSIONAL для сформированной статической модели классов был автоматически сгенерирован программный код.

Класс 1. Пользователь

class User

{

int user\_id;

string login;

string password;

string surname;

string name;

string pathromynic;

string email;

string phone\_number;

public User() { }

public User(string login, string password, string surname, string name, string pathromynic, string email, string phone\_number) { }

public void AddProduct(Product product, int quantity) { }

public void WriteReview(string review\_text) { }

}

Класс 2. Заказ

class Order

{

int id\_order;

Position position;

float summ;

bool status\_payment;

User user;

string requestes;

string order\_date;

public Order() { }

public Order(int id\_order, Position position, float summ, bool status\_payment, User user, string requestes, string order\_date) { }

public void Delete() { }

}

Класс 3. Позиция

class Position

{

int id\_position;

Product product;

int id\_user;

int quantity;

public Position() { }

public Position(int id\_position, Product product, int id\_user, int quantity) { }

public void Delete() { }

}

Класс 4. Товар

class Product

{

int id\_product;

string name\_product;

Category category;

string specification;

float price;

Manufacturer manufacturer;

public Product() { }

public Product(int id\_product, string name\_product, Category category, string specification, float price, Manufacturer manufacturer) { }

}

Класс 5. Категория

class Category

{

int id\_category;

string name\_category;

string discription;

public Category() { }

public Category(int id\_category, string name\_category, string discription) { }

}

Класс 6. Производитель

class Manufacturer

{

int id\_manufacturer;

string name\_manufacturer;

string note;

public Manufacturer() { }

public Manufacturer(int id\_manufacturer, string name\_manufacturer, string note) { }

}

Класс 7. Отзыв

class Review

{

int id\_review;

int id\_product;

User user;

string review\_date;

string review\_text;

bool status\_review;

string last\_change\_date;

Content\_Maker content\_maker;

public Review () { }

public Review(int id\_review, int id\_product, User user, string review\_date, string review\_text, bool status\_review, string last\_change\_date, Content\_Maker content\_maker) { }

public void Change (string review\_text, int id\_content\_maker) { }

}

Класс 8. Контент-мейкер

class Content\_Maker

{

int id\_content\_maker;

string login;

string password;

string surname;

string name;

string pathromynic;

string email;

bool right\_products;

bool right\_reviews;

public Content\_Maker() { }

public Content\_Maker(int id\_content\_maker, string login, string password, string surname, string name, string pathromynic, string email, bool right\_products, bool right\_reviews) { }

public void Moderate(Review review, string review\_text) { }

}

Класс 9. Database\_table

class Database\_table

{

string quary;

public void Execute\_quary(string quary) { }

public void Read\_data() { }

public void Write\_data() { }

}

Класс 10. Database\_connection

class Database\_connection

{

int id\_user;

string password;

string connection\_string;

public void Start\_session() { }

public void End\_session() { }

}

# 7. Руководство пользователя

## 7.1 Авторизация

После открытия сайта в браузере, сперва пользователь попадает на страницу Авторизации (рис 7.1.1), если пользователь не был авторизован до этого в системе. Пользователю необходимо указать логин и пароль.

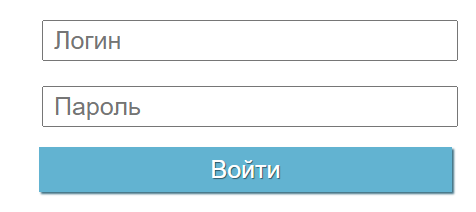


Рисунок 7.1.1. Форма авторизации

После ввода пароля, пользователю необходимо нажать кнопку «Вход». В случае если логин и пароль верный, пользователь увидит главную страницу (рис. 7.3.1), иначе пользователя уведомят о неправильности введенных данных.

## 7.2 Главная страница

Функционал программы зависит от того, кто был авторизирован. Так, например, если процесс авторизации прошел пользователь, он увидит главную страницу в следующем формате (рис 7.2.1). Тут мы видим функционал, который доступен пользователю.

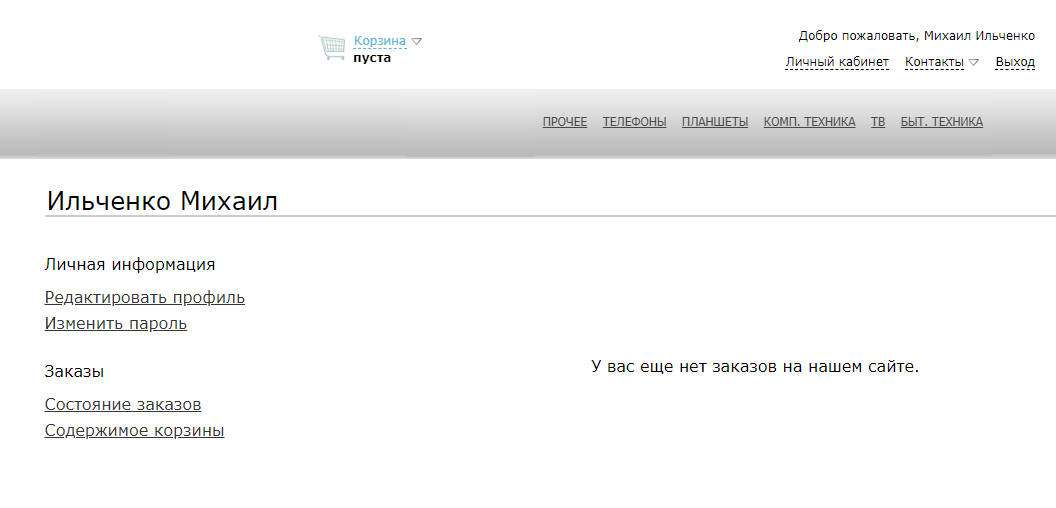


Рисунок 7.2.1 Главная страница пользователя

Действия пользователя могут быть различными:

1. Редактирование профиля;
2. Изменение пароля;
3. Просмотри страницы товаров магазина в различных категориях;
4. Просмотр состояния заказов;
5. Просмотр содержимого корзины;
6. Выйти из системы, нажав кнопку «Выход».

На рисунке 7.2.2 изображена главная страница для кониент-мейкера. После авторизации контент-мейкер попадает на главную страницу. Здесь мы видим функционал, доступный контент-мейкеру.

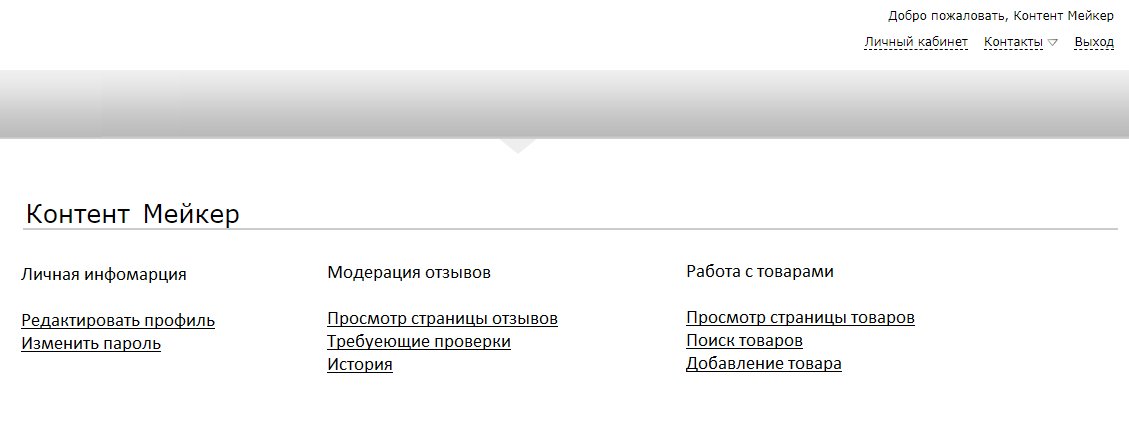


Рисунок 7.2.2. Главная страница контент-мейкера

Действия контент-мейкера могут быть следующими:

1. Просмотр страницы отзывов, в том числе отзывов требующих проверки, а так же история редактирований;
2. Просмотр страницы товаров;
3. Поиск товаров;
4. Редактирование профиля;
5. Изменение пароля;
6. Выйти из системы, нажав кнопку «Выход».

На рисунке 7.2.3 изображена главная страница администратора. После авторизации администратор попадает на свою главную страницу. Здесь мы видим функционал, доступный администратору.

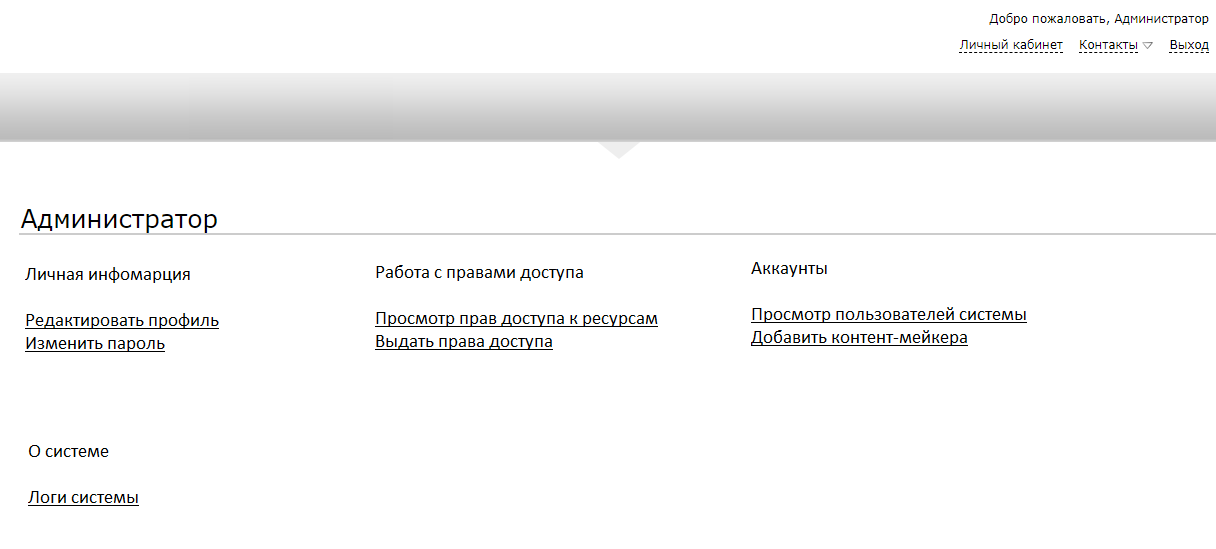


Рисунок 7.2.3. Главная страница администратора

Действия администратора могут быть следующими:

1. Редактирование профиля;
2. Изменение пароля;
3. Просмотр логов системы;
4. Изменение и добавление прав доступа контент-мейкерам;
5. Просмотр всех пользователей информационной системы;
6. Добавление (регистрация) контент-мейкера;
7. Выйти из системы, нажав кнопку «Выход».

## 7.3 Процесс работы в системе пользователя

Процесс работы пользователя в системе заключается в том, что с помощью категорий в верхнем меню, а так же поиска клиент выбирает интересующие его товары, добавляет их в корзину. На Рис. 7.3.1 показана форма представления товаров пользователю. После того, как пользователь выбрал все интересующие его товары, он может их оплатить. Этот процесс показан на Рис. 7.3.2.

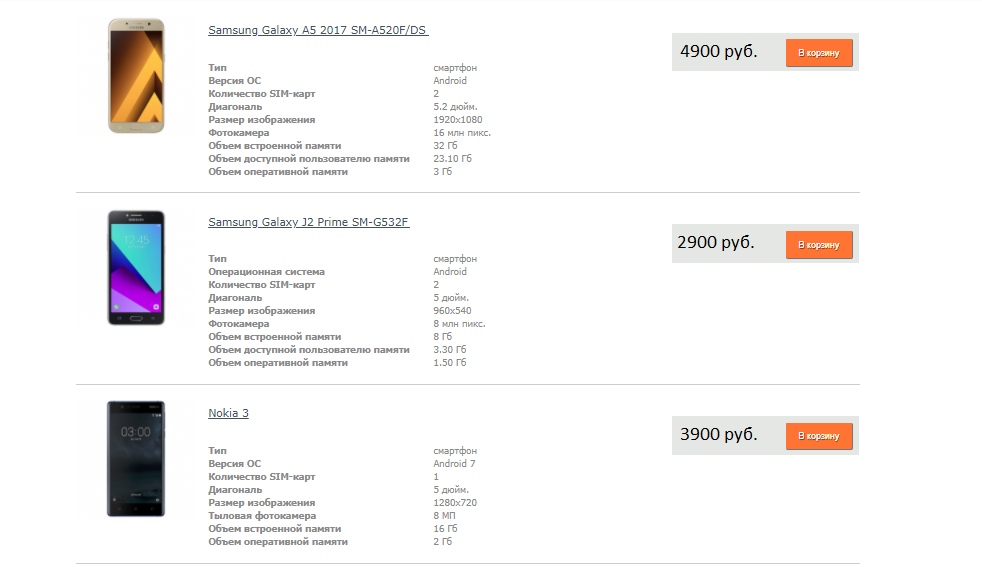


Рисунок 7.3.1. Форма просмотра товаров

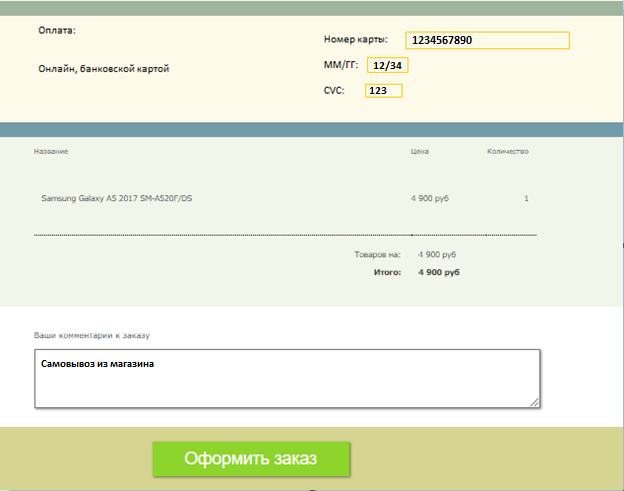


Рисунок 7.3.2. Форма оплаты заказов

Для того чтобы оплатить заказ онлайн банковской картой необходимо ввести данные карты (номер, дата действия, код CVC) в специальные поля на форме оплаты. После этого можно указать комментарий к заказу и нажать кнопку “Оформить заказ” для завершения оформления заказа и оплаты.

## 7.4 Процесс работы контент-мейкера

Процесс работы контент-мейкера заключается в изменении информации имеющихся товаров и добавлении новых (Рисунок 7.4.1), а так же мацерации отзывов (Рисунок 7.4.2)

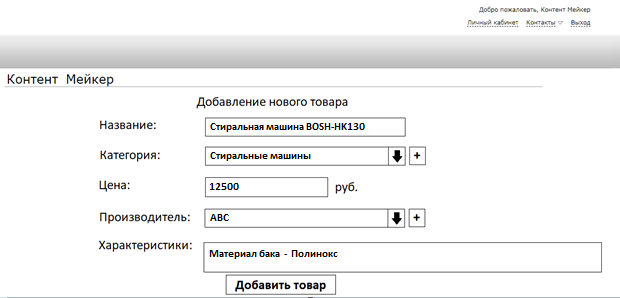


Рисунок 7.4.1. Форма создания маршрутного листа

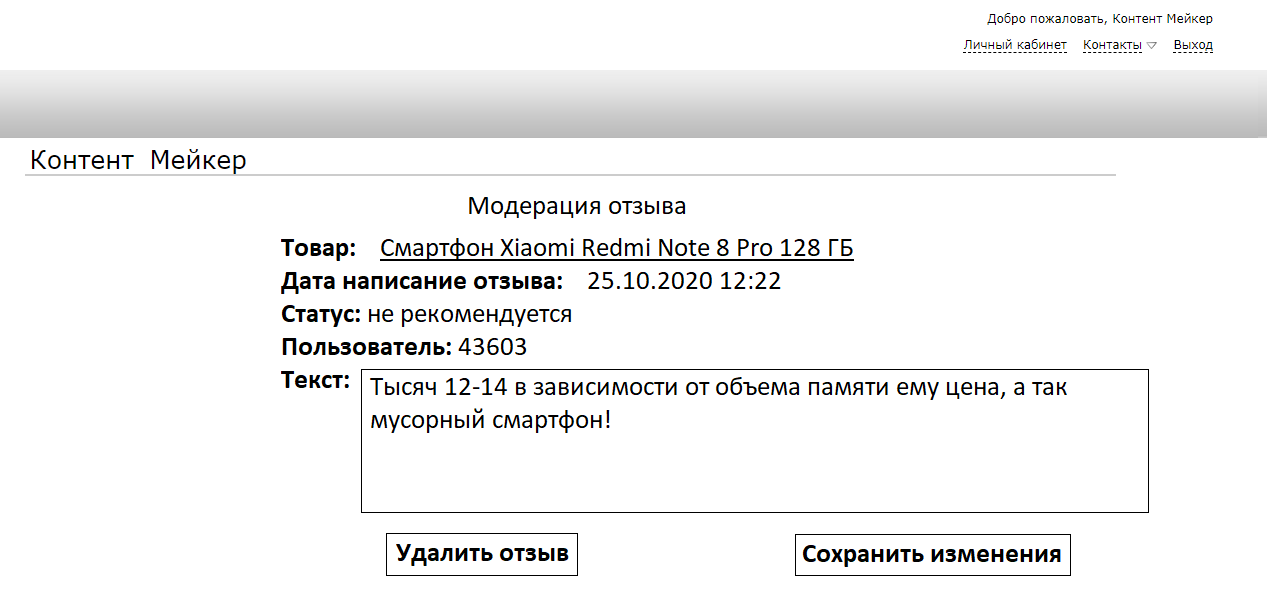


Рисунок 7.4.1. Форма модерации отзывов

Для того чтобы контент-мейкеру перейти на страницу уже имеющегося товара, необходимо с помощью поиска перейти на страницу выбранного товара, изменить необходимые поля, нажать кнопку “Сохранить изменения”.

Во время работы с товарами возле полей “Категория” и “Производитель” есть специальные кнопки навигации, с помощью которых можно либо открыть список уже имеющихся записей в БД, либо создать новую запись.

## 7.5 Процесс работы администратора

Процесс работы администратора заключается в регистрации контент-мейкеров (Рисунок 7.5.1), а так же изменение прав доступа контент-мейкеров.

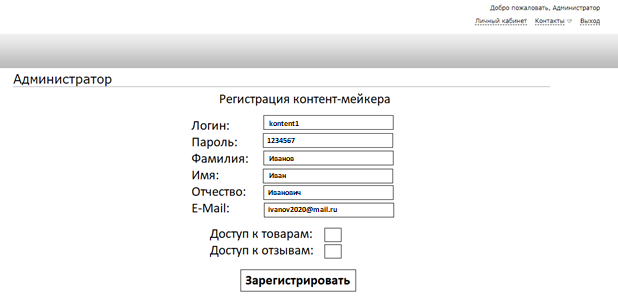


Рисунок 7.5.1. Форма создания отчёта о доставке

После того, как при регистрации будут введены значения логина и пароля они будут автоматически проверены. Логин проверяется на уникальность в базе, а пароль на соответствование требованиям политики безопасности.

Для того, чтобы изменить права доступа контент-мейкера нужно выбрать его аккаунт в списке всех пользователей системы.

# Заключение

Целью данного курсового проект являлась разработка проекта информационной системы «Магазин цифровой и бытовой техники».

В ходе курсового проекта было проведено исследование предметной области, беседы с заказчиком и в полной мере раскрыта тема поставленной цели.

Разработаны и построены различные модели для проекта информационной системы. С учетом построенных моделей была разработана информационная система. Для разработанной информационной системы было написано руководство пользователя.

Разработанное приложение построено с использованием актуальных технологий веб-разработки на сегодняшний день.

# Список используемых источников

1. Анисимов, В. В. Проектирование информационных систем: курс лекций [Текст] : в 2 ч. / В.В. Анисимов. – Хабаровск: Изд-во ДВГУПС, 2006. – Ч. 1; Структурный подход. – 2006. – 112 с.

2. Анисимов, В. В. Проектирование информационных систем: курс лекций [Текст] : в 2 ч. / В. В. Анисимов. – Хабаровск: Изд-во ДВГУПС, 2007. – Ч. 2; Обектно-ориентированный подход. – 2007. – 100 с.

3. Гурвиц, Г. А. Microsoft Access 2010. Разработка приложений на реаль-ном примере [Текст] / Г. А. Гурвиц. Санкт-Петербург : БХВ-Петербург, 2010. - 493 с.