Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Специальность: «Программное обеспечение информационных технологий»

**КОНТРОЛЬНАЯ РАБОТА**

По курсу: «Технология разработки программного обеспечения»

На тему: «Интернет-магазин»

Студент-заочник 2 курса

Группы № 581072

ФИО: Шершень Николай Михайлович

Проверила: Бакунова О.М.

Минск, 2017

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc472637182)

[1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc472637183)

[1.1 Постановка задачи 5](#_Toc472637184)

[1.2 Описание и обоснование выбранного языка программирования 5](#_Toc472637185)

[2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА 7](#_Toc472637186)

[3 ЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ 11](#_Toc472637187)

[3.1 Диаграмма вариантов использования 11](#_Toc472637188)

[3.2 Диаграмма классов 12](#_Toc472637189)

[3.3 Диаграмма деятельности 12](#_Toc472637190)

[3.4 Диаграмма коопераций 13](#_Toc472637191)

[3.5 Диаграмма последовательности 14](#_Toc472637192)

[3.6 Диаграмма развёртывания 15](#_Toc472637193)

[3.7 Диаграмма компонентов 16](#_Toc472637194)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17](#_Toc472637195)

[Генерация кода 18](#_Toc472637196)

# ВВЕДЕНИЕ

Отличительной чертой современного человечества является: производство, накопление, распределение и обработка имеющейся и получаемой информации. Наш век смело можно было бы назвать веком компьютеров, мультиварок, плазменных экранов и веком интеграции.

Спрос на покупку товаров в сети постоянно растет, все больше людей способны и желают приобретать ее, понимая, что без нее невозможно обходиться. Ведь заказ товаров онлайн стала неотъемлемой частью нашей жизни.

В связи со спросом на бытовую технику появляются организации способные удовлетворить потребность в товарах, и с каждым днем подобных организаций становиться все больше и больше. Бытовая техника в наши дни - это необходимость, которая дает возможность комфортно отдыхать и работать. Поэтому продажа бытовой техники - область весьма интересная и прибыльная, актуальная во все времена, начиная с появления первого холодильника.

Наличие магазинов, которые готовы предоставить любой вид бытовой техники и услуг - это огромный плюс. К необходимым товарам вам предоставляется неимоверное количество сопутствующих товаров, запасных частей, расходных материалов.

Интернет, как средство массовой информации существует достаточно давно. Для большинства людей, использующих Интернет, это просто средство для получения какой-либо информации, либо средство коммуникации. Одним из лежащих на поверхности способов использования Интернета для бизнеса является интернет-продажа товаров и услуг.

Интернет магазин - очень удобная система демонстрации и продажи товаров и услуг в Интернет. Сегодня в белорусской сети Интернет представлены более 700 электронных магазинов. Интернет-магазин подходит для размещения большого количества информации, позволяет оперативно обновлять ассортимент, четко контролировать рабочие процессы, например, автоматически обновлять прайс-листы. Причем Интернет-магазин может быть, как продолжением традиционного бизнеса, так и совершенно независимой структурой. В любом случае, при правильной организации работы, он будет приносить прибыль.

Таким образом, главная цель работы интернет магазина - привлечение большего числа клиентов, расширение рынка товаров и услуг, возможность автоматической обработки данных, получение прибыли. И главной задачей для достижения этой цели является его хорошее проектирование.

# 1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

# Постановка задачи

Много лет прошло с тех пор, как появилась Всемирная сеть Интернет. С течением времени изменилось многое: технические возможности Сети расширились, аудитория выросла, цели и задачи, для решения которых была создана Сеть, перестали быть исключительно военными или научными. Современный Интернет - это огромное хранилище, в котором каждый по своему желанию может найти нужный текст, музыку, учебные материалы, а с некоторого времени и просто купить все, что необходимо - от продуктов, до программного обеспечения. В данное время многие дальновидные руководители компаний совершенно справедливо считают необходимым создание небольшого Интернет-представительства компании, другими словами все больше появляется коммерческих сайтов, основной целью которых является продвижение товаров и услуг с помощью Интернет.

Уже несколько лет в Беларуси успешно развивается электронная коммерция. Как и в обычной коммерции, в электронной коммерции можно выделить два направления: межфирменное деловое партнерство и розничная торговля. Первое направление развивается довольно успешно, так как многие фирмы и компании поняли, что заключать договора и вести переговоры с помощью технологий Интернет гораздо быстрее и дешевле, нежели обычными способами. Что касается розничной торговли, то в Беларуси она находится на начальном этапе развития по сравнению с аналогичным направлением в мире. Однако уже сейчас можно сказать, что конкуренция в розничной торговле через Интернет такая же напряженная, как и в обычной розничной торговле.

Однако, что же представляет собой Интернет-магазин, как он создается, в чем его преимущества и в чем недостатки? Прежде всего, Интернет-магазин - это сайт, содержащий подробный каталог товаров с описанием и изображением. Основное отличие от обычного интернет-каталога состоит в том, что товары, представленные в интернет-магазине можно не только увидеть, но и заказать, не вставая с места и не прерывая увлекательного путешествия по просторам Интернет.

Для всех интернет-магазинов свойственен определенный обязательный набор элементов, таких как:

- специализированный каталог с подразделами, в которых представлены все имеющиеся в наличии товары. Внешний вид каталога может быть разным - дерево, выпадающие или вложенные списки меню.

- система регистрации пользователя, которая создает для каждого нового клиента его собственную «корзину», в которую можно «положить» выбранный товар и впоследствии заказать. По мере передвижения клиента по каталогу система также отслеживает предпочтения клиента, на основе которых в будущем может строиться не только ассортимент магазина, но и структура выдачи сопутствующей информации каталога. Например, если человек заказал книгу, система может ненавязчиво предложить посмотреть другие книги из той же серии или той же тематики или предложить почитать отзывы читателей. Система может «доложить», что люди, которые заказывали подобный товар, также интересовались сопутствующими товарами - например, при покупке стиральной машины предлагается выбрать стиральный порошок.

- система оплаты товара: покупателю предлагается использовать различные способы оплаты - кредитные карты, электронные деньги, оплата наличными (курьеру или при получении на почте).

- система доставки товара: здесь тоже широкий выбор возможностей: пересылка по электронной почте (программное обеспечение), доставка курьерской службой, обычная почта.

- система технической поддержки: покупатель может обратиться за помощью с выбором товара или вопросом по эксплуатации уже купленного товара.

Анализ предметной области позволил выявить минимальный набор из семи сущностей. Это будут:

1. Пользователь – информация о покупателе и его данных для входа.
2. Товар – непосредственно продаваеваемые товары.
3. Категория товара – возможные категории товаров.
4. Бренд – возможные бренды товаррв.
5. Отзыв – отзыв о товарах интернет магазина.
6. Корзина – данные о выбранных пользователем товаров для последующего заказа.
7. Заказ – данные о товарах для доставки интернет-магазином.

# 1.2 Описание и обоснование выбранного языка программирования

Java — объектно-ориентированный язык программирования, разрабатываемый компанией Sun Microsystems с 1991 года и официально выпущенный 23 мая 1995 года. Изначально новый язык программирования назывался Oak (James Gosling) и разрабатывался для бытовой электроники, но впоследствии был переименован в Java и стал использоваться для написания апплетов, приложений и серверного программного обеспечения

Отличительной особенностью Java в сравнении с другими языками программирования общего назначения является обеспечение высокой продуктивности программирования, нежели производительность работы приложения или эффективность использования им памяти.

В Java используются практически идентичные соглашения для объявления переменных, передачи параметров, операторов и для управления потоком выполнением кода. В Java добавлены все хорошие черты C++.

Три ключевых элемента объединились в технологии языка Java

Java предоставляет для широкого использования свои апплеты (applets) — небольшие, надежные, динамичные, не зависящие от платформы активные сетевые приложения, встраиваемые в страницы Web. Апплеты Java могут настраиваться и распространяться потребителям с такой же легкостью, как любые документы HTML

Java высвобождает мощь объектно-ориентированной разработки приложений, сочетая простой и знакомый синтаксис с надежной и удобной в работе средой разработки. Это позволяет широкому кругу программистов быстро создавать новые программы и новые апплеты

Java предоставляет программисту богатый набор классов объектов для ясного абстрагирования многих системных функций, используемых при работе с окнами, сетью и для ввода-вывода. Ключевая черта этих классов заключается в том, что они обеспечивают создание независимых от используемой платформы абстракций для широкого спектра системных интерфейсов

Огромное преимущество Java заключается в том, что на этом языке можно создавать приложения, способные работать на различных платформах. К сети Internet подключены компьютеры самых разных типов - Pentium PC, Macintosh, рабочие станции Sun и так далее. Даже в рамках компьютеров, созданных на базе процессоров Intel, существует несколько платформ, например, Microsoft Windows версии 3.1, Windows 95, Windows NT, OS/2, Solaris, различные разновидности операционной системы UNIX с графической оболочкой X­Windows. Между тем, создавая сервер Web в сети Internet, хотелось бы, чтобы им могло пользоваться как можно большее число людей. В этом случае выручат приложения Java, предназначенные для работы на различных платформах и не зависящие от конкретного типа процессора и операционной системы.

# 2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

В работе проводится моделирование с использование IDEF0(BPWin), UML (Rational Rose 2000), IDEF1x (ErWin).

1. Важная роль отводится процессу функционального проектирования.

Для регламентирования создания функциональных моделей ПС предназначен стандарт IDEF0 (Integrated Definition Function Modeling), который и реализован в пакете BpWin.

В основе IDEF0 лежит понятие блока, который реализует некую конкретную функцию. Четыре стороны блока имеют разное назначение. Слева отображаются входные данные (исходные данные). Справа – выходные данные (результат выполнения функции). Сверху – управление (управляющие воздействия на функцию). Снизу – механизм (посредством чего реализуется данная функция).

Функция – это управляемое действие над входными данными, результатом которого являются выходные данные, при этом используется некий механизм. Взаимодействие между функциями отображается в виде стрелок. Иногда стороны блока называют направлениями, а стрелки потоками. Стрелки можно подписывать. Подписи связываются с конкретной стрелкой при помощи зигзага.

В основе IDEF0 лежит три базовых принципа:

1) принцип функциональной декомпозиции – любая функция может быть разбита (декомпозирована) на более простые функции (более понятен термин детализация);

2) принцип ограничения сложности – количество блоков на диаграмме должно быть не менее двух, но не более шести (условие удобочитаемости);

3) принцип контекста – моделирование делового процесса начинается с построения контекстной диаграммы, на которой отображается только один блок – главная функция моделирующей системы, ограничивающая область границы моделирующей системы (регламентирует начальный этап построения модели).

Под субъектом понимается сама система, при этом необходимо точно установить, что входит в систему, а что является внешним воздействием на систему. Т. е. первоначально нужно определить область моделирования. Описание области как системы в целом, так и ее компонентов является основой построения модели.

IDEF1X является методом для разработки реляционных баз данных и использует условный синтаксис, специально разработанный для удобного построения концептуальной схемы. Концептуальной схемой мы называем универсальное представление структуры данных в рамках коммерческого предприятия, независимое от конечной реализации базы данных и аппаратной платформы. Будучи статическим методом разработки, IDEF1X изначально не предназначен для динамического анализа по принципу "AS IS", тем не менее, он иногда применяется в этом качестве, как альтернатива методу IDEF1. Использование метода IDEF1X наиболее целесообразно для построения логической структуры базы данных после того, как все информационные ресурсы исследованы (скажем с помощью метода IDEF1) и решение о внедрении реляционной базы данных, как части корпоративной информационной системы, было принято.

Хотя терминология IDEF1X практически совпадает с терминологией IDEF1, существует ряд фундаментальных отличий в теоретических концепциях этих методологий. Сущность в IDEF1X описывает собой совокупность или набор экземпляров похожих по свойствам, но однозначно отличаемых друх от друга по одному или нескольким признакам. Каждый экземпляр является реализацией сущности. Таким образом, сущность в IDEF1X описывает конкретный набор экземпляров реального мира, в отличие от сущности в IDEF1, которая представляет собой абстрактный набор информационных отображений реального мира. Примером сущности IDEF1X может быть сущность "ПОСЕТИТЕЛЬ", которая представляет собой всех посетителей предприятия, а один из них, скажем, купивший пылесос, является конкретной реализацией этой сущности.

Сase-средство BPwin значительно облегчает задачу создания информационной системы, позволяя осуществить декомпозицию сложной системы на более простые с тем, чтобы каждая из них могла проектироваться независимо, и для понимания любого уровня проектирования достаточно было оперировать с информацией о немногих ее частях.

Рассмотрим процесс моделирования в методологии IDEF0 на примере контекстной диаграммы разработанной модели процесса учета кредитных операций в банке.

Цель модели – моделирование процесса функционирования интернет-магазина.

Входами модели являются:

1. Авторизация (авторизационные данные) – данные, которые используются при авторизации любого пользователя системы;
2. Информация о товарах – содержит такую информация как название, бренд, категория товара;
3. Информация о заказах – та информация, которая используется для последующей отправке товаров пользователям;
4. Информация о пользователях – информация, которая отображает текущих пользователей, информацию о них – имя, адрес, электронная почта и их авторизационные данные.

Процесс регламентируется информацией о наличии товаров на складе.

На выходе получается запись в БД, а также отчетная документация. В качестве пользователей системы могут выступать администратор, который владеет самыми широкими правами, и пользователь (владелец), который может смотреть и заказывать товары.

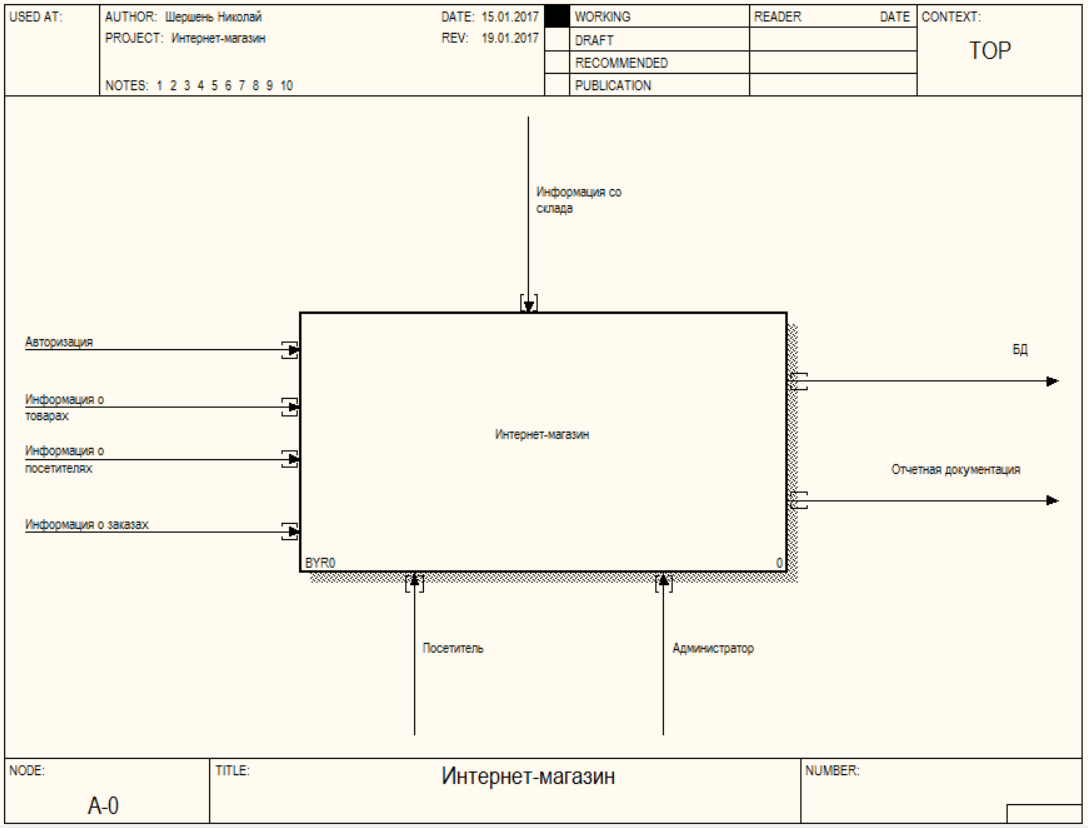


Рисунок 2.1 – Контекстная диаграмма модели

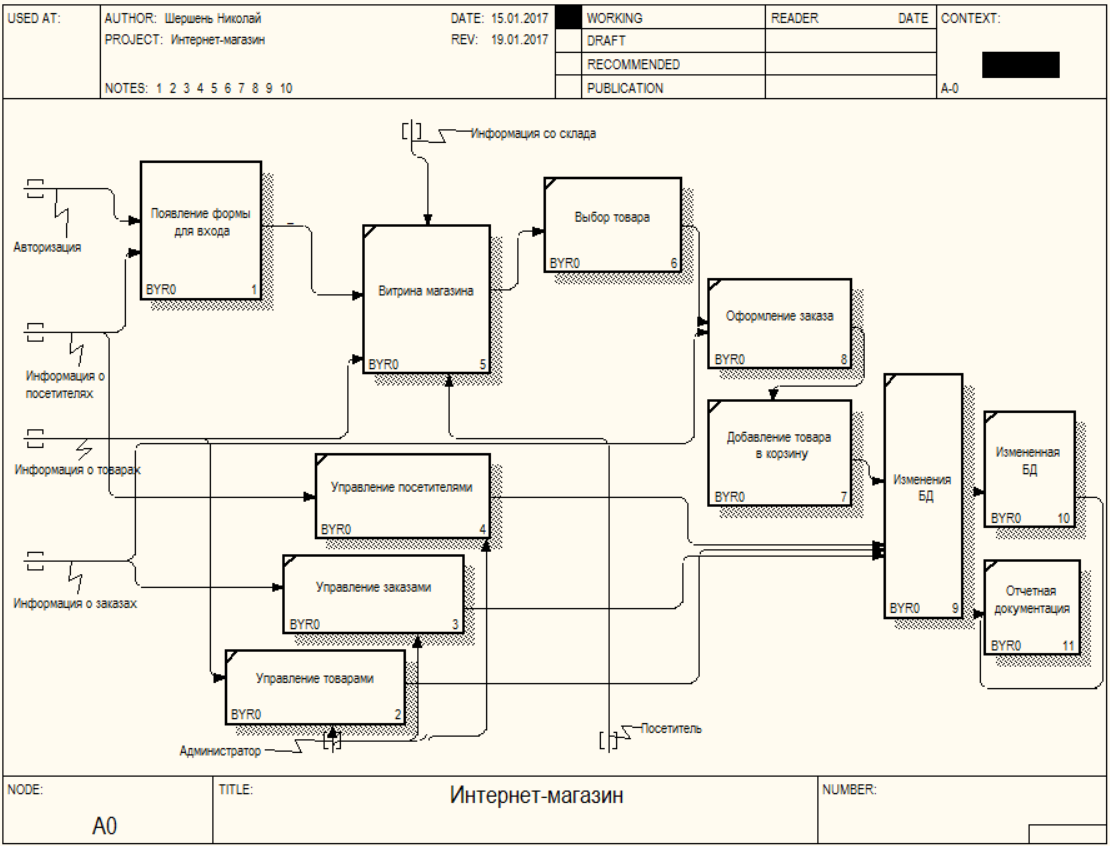


Рисунок 2.2– Декомпозиция контекстной диаграммы

функциональной модели

Декомпозиция модели делится на внутренние процессы такие как:

1. Появление формы для входа;
2. Управление товарами;
3. Управление заказами;
4. Управление посетителями;
5. Просмотр товаров магазина;
6. Выбор товара;
7. Добавление товара в корзину;
8. Оформление заказа
9. Изменения БД;
10. Измененная БД;
11. Отчетная документация;

# 3 ЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Различные элементы логического представления, такие как классы, ассоциации, состояния, сообщения, не существуют материально или физически. Они лишь отражают наше понимание структуры физической системы или аспекты ее поведения.

# 3.1 Диаграмма вариантов использования

Модель вариантов использования предназначается для определения требований к системе. Она включает в себя актеров, варианты использования и связи между ними. Для отображения этой модели язык UML предлагает использовать диаграммы Use Case (вариант использования) совместно с моделями State Diagram (диаграммы состояний) и Activity Diagram (диаграммы деятельности/активности). Последние используются для конкретизации вариантов использования системы

Разработка диаграммы вариантов использования преследует цели:

* определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
* сформулировать общие требования к функциональному поведению проектируемой системы;
* разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
* подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Диаграмма вариантов использования представлена на рисунке 3.1.

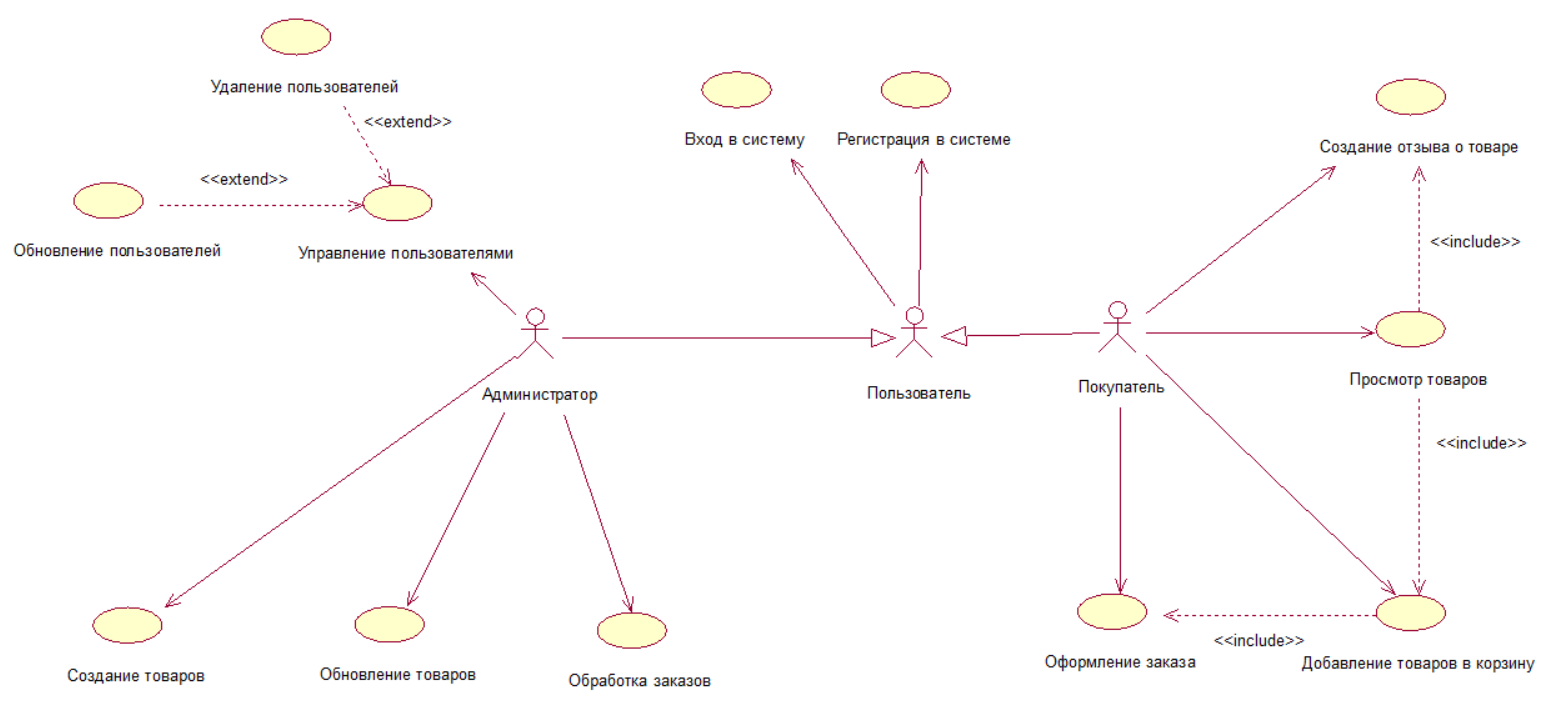


Рисунок 3.1 – Диаграмма вариантов использования

# 3.2 Диаграмма классов

Диаграмма классов — диаграмма, демонстрирующая классы системы, их атрибуты, методы и взаимосвязи между ними. Входит в UML.

Существует два вида:

* статический вид диаграммы рассматривает логические взаимосвязи классов между собой;
* аналитический вид диаграммы рассматривает общий вид и взаимосвязи классов, входящих в систему.

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

* концептуальная точка зрения — диаграмма классов описывает модель предметной области, в ней присутствуют только классы прикладных объектов;
* точка зрения спецификации — диаграмма классов применяется при проектировании информационных систем;
* точка зрения реализации — диаграмма классов содержит классы, используемые непосредственно в программном коде (при использовании объектно-ориентированных языков программирования).

Диаграмма классов представлена на рисунке 3.2.

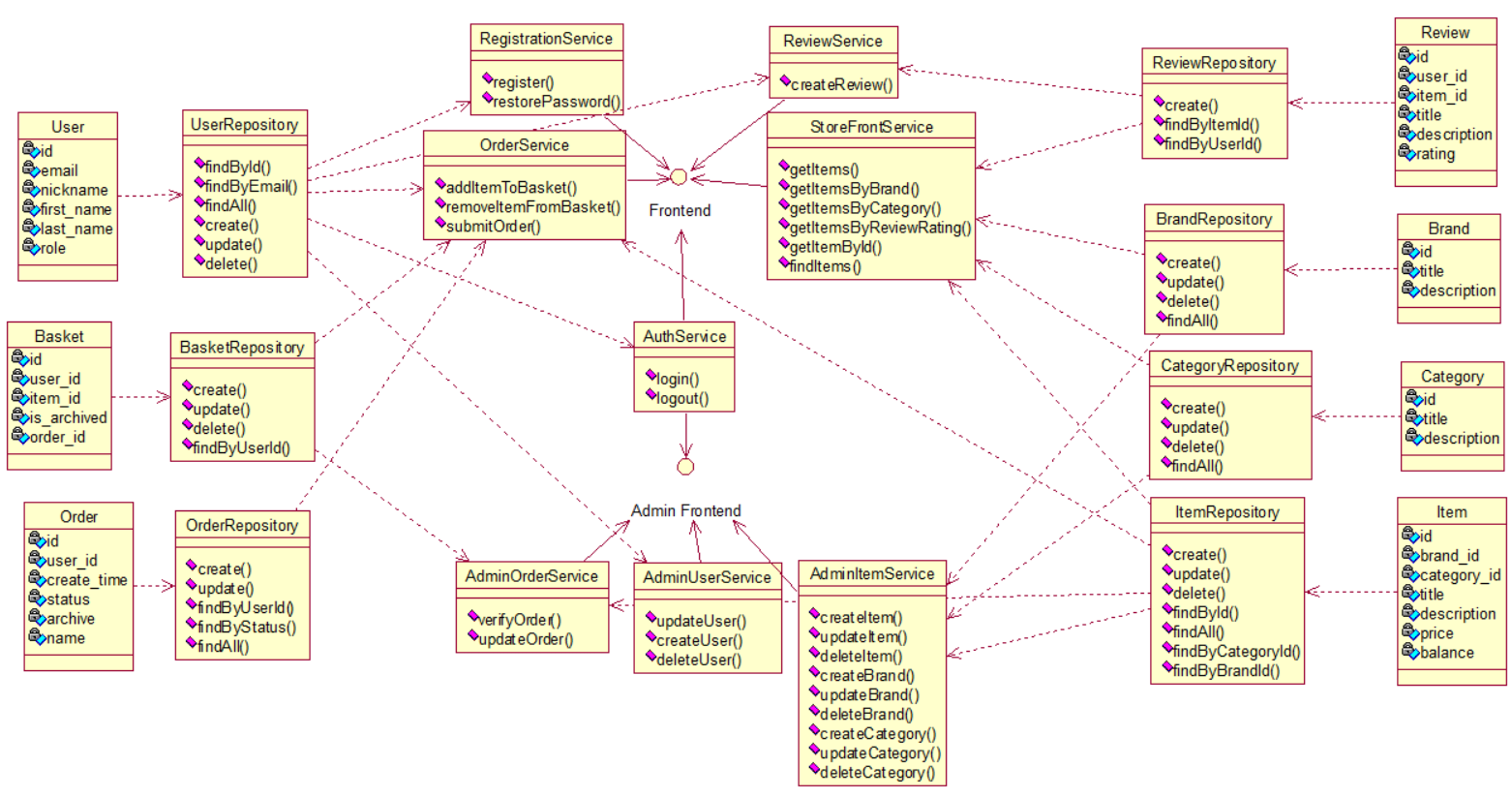


Рисунок 3.2 – Диаграмма классов

# 3.3 Диаграмма деятельности

Диаграмма деятельности — UML-диаграмма, на которой показано разложение некоторой деятельности на её составные части. Под деятельностью (англ. activity) понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов — вложенных видов деятельности и отдельных действий англ. action, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Диаграмма деятельности представлена на рисунке 3.3.

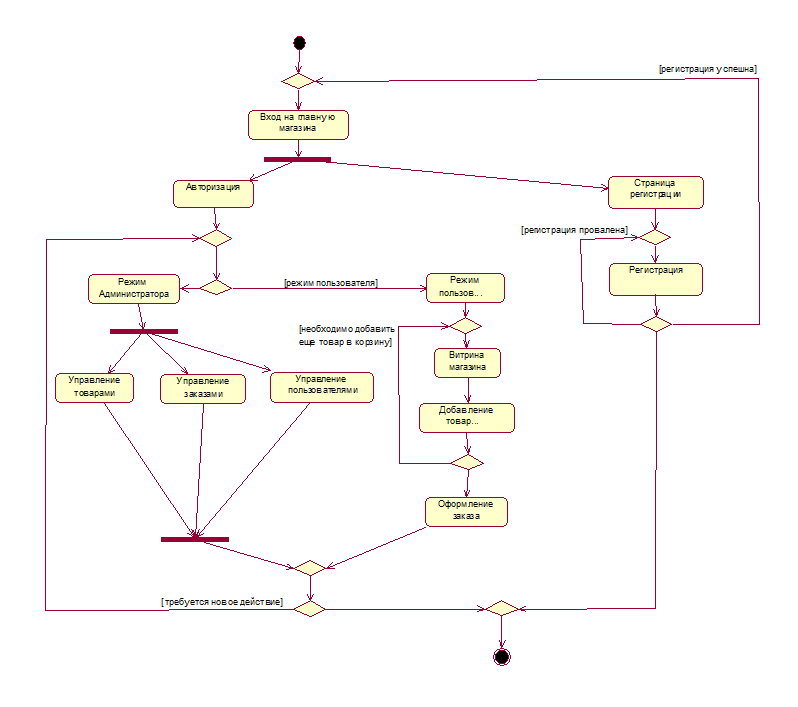


Рисунок 3.3 – Диаграмма деятельности

# 3.4 Диаграмма коопераций

Главная особенность диаграммы кооперации заключается в возможности графически представить не только последовательность взаимодействия, но и все структурные отношения между объектами, участвующими в этом взаимодействии.

Прежде всего, на диаграмме кооперации в виде прямоугольников изображаются участвующие во взаимодействии объекты, содержащие имя объекта, его класс и, возможно, значения атрибутов. Далее, как и на диаграмме классов, указываются ассоциации между объектами в виде различных соединительных линий. При этом можно явно указать имена ассоциации и ролей, которые играют объекты в данной ассоциации. Дополнительно могут быть изображены динамические связи - потоки сообщений. Они представляются также в виде соединительных линий между объектами, над которыми располагается стрелка с указанием направления, имени сообщения и порядкового номера в общей последовательности инициализации сообщений.

В отличие от диаграммы последовательности, на диаграмме кооперации изображаются только отношения между объектами, играющими определенные роли во взаимодействии. На этой диаграмме не указывается время в виде отдельного измерения. Поэтому последовательность взаимодействий и параллельных потоков может быть определена с помощью порядковых номеров. Следовательно, если необходимо явно специфицировать взаимосвязи между объектами в реальном времени, лучше это делать на диаграмме последовательности. Диаграмма коопераций представлена на рисунке 3.4.

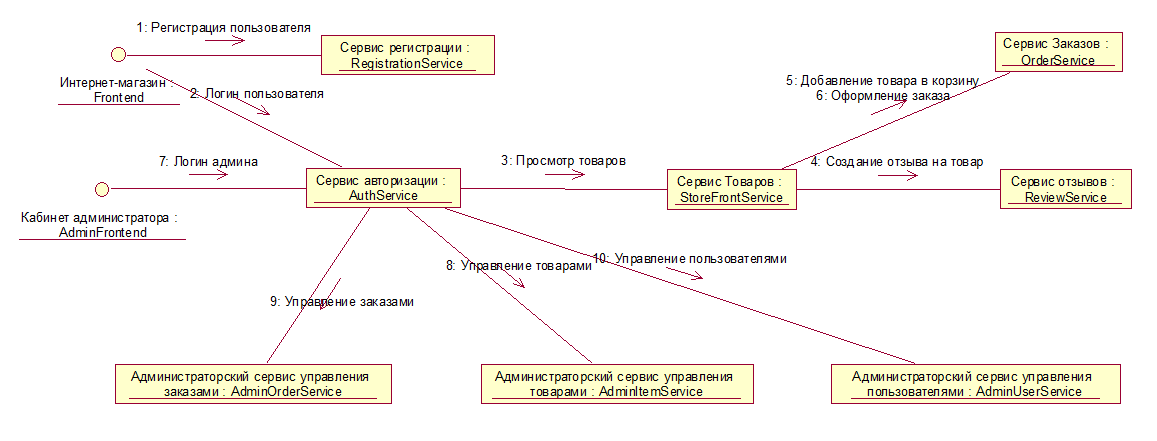


Рисунок 3.4 – Диаграмма коопераций

# 3.5 Диаграмма последовательности

Диаграмма последовательности — диаграмма, на которой показано взаимодействие объектов, упорядоченное по времени, с отражением продолжительности обработки и последовательности их проявления. Используется в языке UML.

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные «линии жизни», отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелки, показывающие обмен сигналами или сообщениями между объектами.

Диаграмма последовательности представлена на рисунке 3.5.

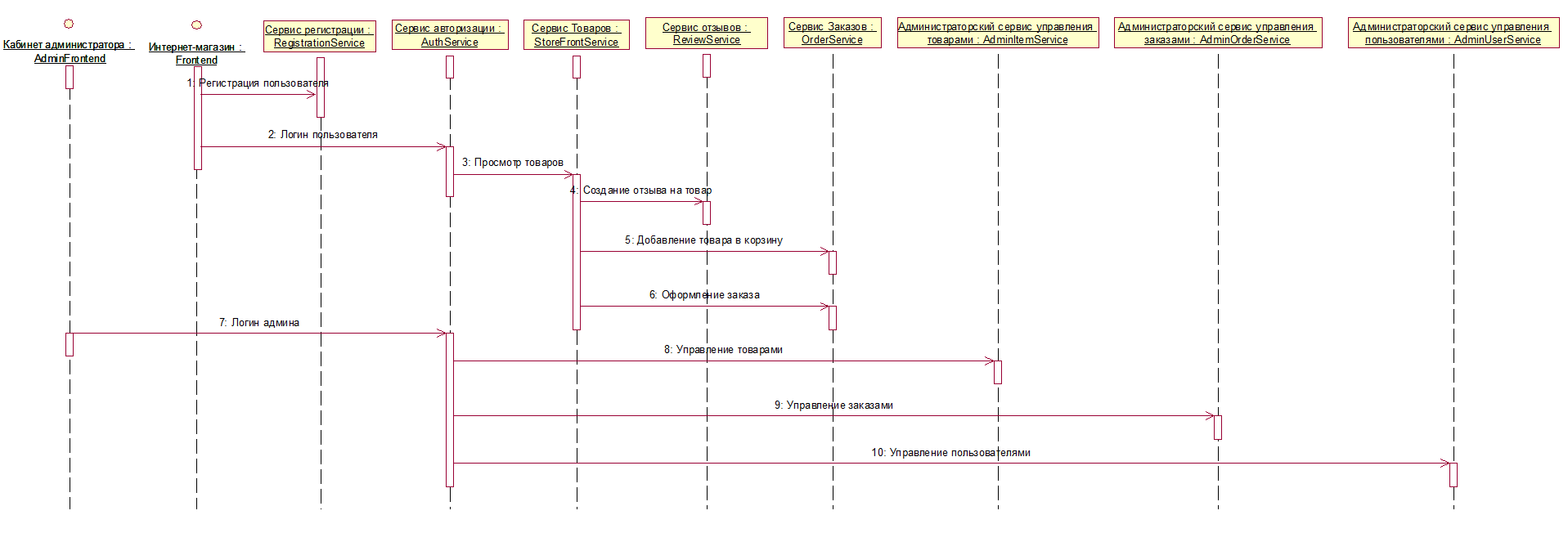


Рисунок 3.5 – Диаграмма последовательности

# 3.6 Диаграмма развёртывания

Диаграмма развёртывания в UML моделирует физическое развертывание артефактов на узлах. Например, чтобы описать веб-сайт диаграмма развертывания должна показывать, какие аппаратные компоненты («узлы») существуют (например, веб-сервер, сервер базы данных, сервер приложения), какие программные компоненты («артефакты») работают на каждом узле (например, веб-приложение, база данных), и как различные части этого комплекса соединяются друг с другом (например, JDBC, REST, RMI).

Узлы представляются как прямоугольные параллелепипеды с артефактами, расположенными в них, изображенными в виде прямоугольников. Узлы могут иметь под узлы, которые представляются как вложенные прямоугольные параллелепипеды. Один узел диаграммы развертывания может концептуально представлять множество физических узлов, таких как кластер серверов баз данных.

Существует два типа узлов:

* узел устройства;
* узел среды выполнения.

Узлы устройств — это физические вычислительные ресурсы со своей памятью и сервисами для выполнения программного обеспечения, такие как обычные ПК, мобильные телефоны. Узел среды выполнения — это программный вычислительный ресурс, который работает внутри внешнего узла и который предоставляет собой сервис, выполняющий другие исполняемые программные элементы. Диаграмма развёртывания представлена на рисунке 3.6.

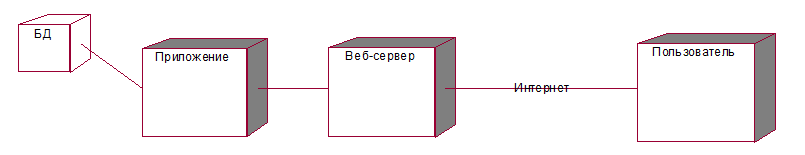


Рисунок 3.6 – Диаграмма развёртывания

# 3.7 Диаграмма компонентов

Диаграмма компонентов — статическая структурная диаграмма, показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п.

Компоненты связываются через зависимости, когда соединяется требуемый интерфейс одного компонента с имеющимся интерфейсом другого компонента. Таким образом иллюстрируются отношения клиент-источник между двумя компонентами.

Зависимость показывает, что один компонент предоставляет сервис, необходимый другому компоненту. Зависимость изображается стрелкой от интерфейса или порта клиента к импортируемому интерфейсу.

Когда диаграмма компонентов используется, чтобы показать внутреннюю структуру компонентов, предоставляемый и требуемый интерфейсы составного компонента могут делегироваться в соответствующие интерфейсы внутренних компонентов.

Делегация показывается связь внешнего контракта компонента с внутренней реализацией этого поведения внутренними компонентами.

Диаграмма компонентов представлена на рисунке 3.7.

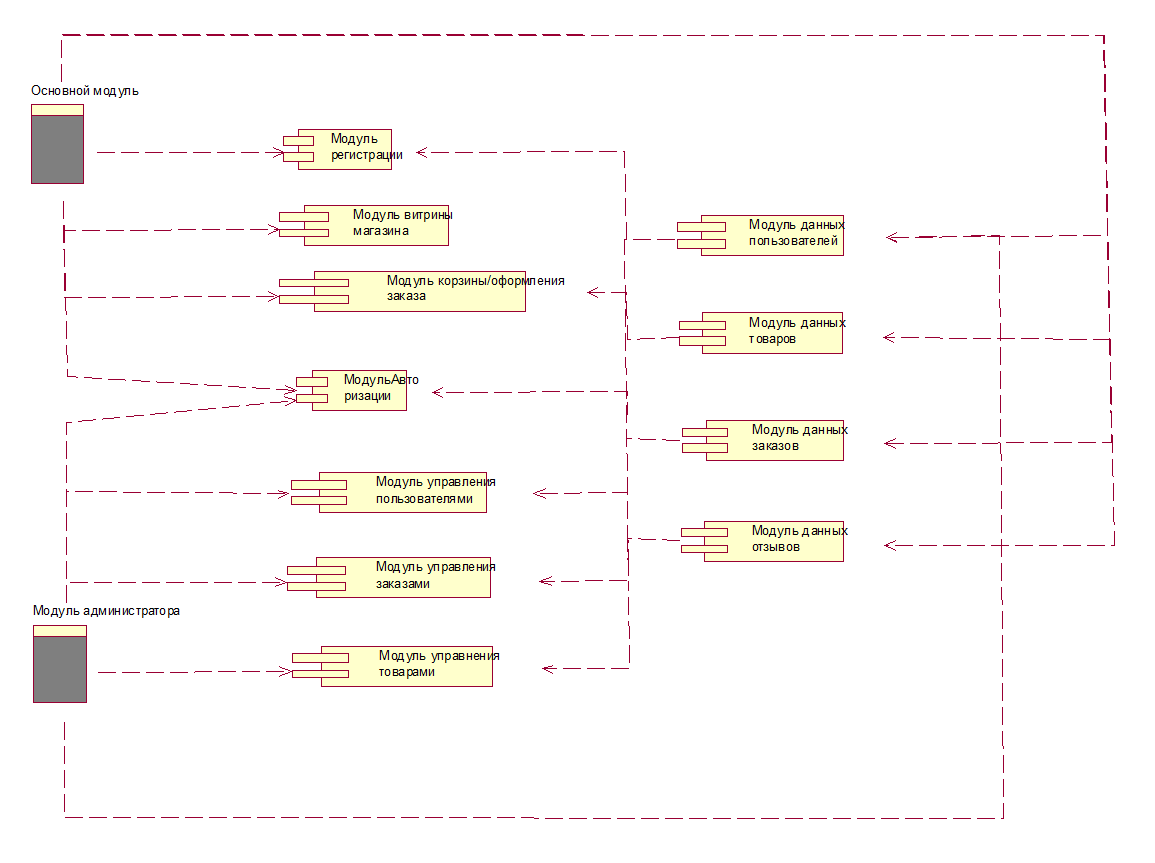


Рисунок 3.7 – Диаграмма компонентов

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. М. Боггс. UML и Rational Rose. - «Лори». 2008. 580 с.
2. Х. Гома. UML Проектирование систем реального времени, параллельных и распределенных приложений ДМК. – «». 2016, 700 с.
3. Маклаков С.В. BPwin и ERwin: CASE - средства для разработки информационных систем.
4. Федотов Д.Э., Семенов Ю.Д., Чижик К.Н. Практикум для высших учебных заведений. CASE-технологии. -157с.
5. Интернет обучение [Электронный ресурс] / Разработка диаграммы кооперации и редактирование свойств ее элементов – Режим доступа: http://www.intuit.ru/ – Дата доступа: 12.01.2017.
6. Интернет обучение [Электронный ресурс] / Информационные системы и технологии – Режим доступа: <http://www.narfu.ru> – Дата доступа: 12.01.2017.
7. Интернет обучение [Электронный ресурс] / Нормализация структурны данных – Режим доступа: http://infostart.ru/public/269803/ – Дата доступа: 12.01.2017.
8. Р. Мюллер. [Базы данных и UML: Проектирование](javascript:parent.descr(window,%22d160587%22)).– Лори, 2002г. 432 с.
9. Фельдман С.К. Система программирования Java без секретов: Как создать безопасное приложение с "нуля". – Новый издательский дом" , 2005 г. , 347 с.
10. Дейтел П.Дж., Дейтел Х.М. Как програмировать на Java. Книга 2. Файлы, сети, базы данных. – "Бином" · 2005 г., 672 с.

# Генерация кода

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\AdminFrontend.java

public interface AdminFrontend

{

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\AdminItemService.java

public class AdminItemService

{

public AdminFrontend theAdminFrontend;

/\*\*

\* @roseuid 58812B5D0236

\*/

public AdminItemService()

{

}

/\*\*

\* @roseuid 5879FC47000E

\*/

public void createItem()

{

}

/\*\*

\* @roseuid 5879FC4A02EA

\*/

public void updateItem()

{

}

/\*\*

\* @roseuid 5879FC54003A

\*/

public void deleteItem()

{

}

/\*\*

\* @roseuid 5879FC5603A2

\*/

public void createBrand()

{

}

/\*\*

\* @roseuid 5879FC60018A

\*/

public void updateBrand()

{

}

/\*\*

\* @roseuid 5879FC640366

\*/

public void deleteBrand()

{

}

/\*\*

\* @roseuid 5879FC6C0126

\*/

public void createCategory()

{

}

/\*\*

\* @roseuid 5879FC7102F6

\*/

public void updateCategory()

{

}

/\*\*

\* @roseuid 5879FC780076

\*/

public void deleteCategory()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\AdminOrderService.java

public class AdminOrderService

{

public AdminFrontend theAdminFrontend;

/\*\*

\* @roseuid 58812B5D01A8

\*/

public AdminOrderService()

{

}

/\*\*

\* @roseuid 5879FD6E01A4

\*/

public void verifyOrder()

{

}

/\*\*

\* @roseuid 587A00B001BF

\*/

public void updateOrder()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\AdminUserService.java

public class AdminUserService

{

public AdminFrontend theAdminFrontend;

/\*\*

\* @roseuid 58812B5D01E2

\*/

public AdminUserService()

{

}

/\*\*

\* @roseuid 5879FD8D0398

\*/

public void updateUser()

{

}

/\*\*

\* @roseuid 5879FD9402D1

\*/

public void createUser()

{

}

/\*\*

\* @roseuid 5879FD9903E1

\*/

public void deleteUser()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\AuthService.java

public class AuthService

{

public Frontend theFrontend;

public AdminFrontend theAdminFrontend;

/\*\*

\* @roseuid 58812B5D016C

\*/

public AuthService()

{

}

/\*\*

\* @roseuid 5879FDC402CE

\*/

public void login()

{

}

/\*\*

\* @roseuid 5879FDCB0199

\*/

public void logout()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Basket.java

public class Basket

{

private int id;

private int user\_id;

private int item\_id;

private int is\_archived;

private int order\_id;

/\*\*

\* @roseuid 58812B5C0202

\*/

public Basket()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\BasketRepository.java

public class BasketRepository

{

/\*\*

\* @roseuid 58812B5C03A4

\*/

public BasketRepository()

{

}

/\*\*

\* @roseuid 5879FEB40281

\*/

public void create()

{

}

/\*\*

\* @roseuid 587A00DB0253

\*/

public void update()

{

}

/\*\*

\* @roseuid 5879FEB7011C

\*/

public void delete()

{

}

/\*\*

\* @roseuid 5879FF0A038D

\*/

public void findByUserId()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Brand.java

public class Brand

{

private int id;

private int title;

private int description;

/\*\*

\* @roseuid 58812B5C01AB

\*/

public Brand()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\BrandRepository.java

public class BrandRepository

{

/\*\*

\* @roseuid 58812B5C02C7

\*/

public BrandRepository()

{

}

/\*\*

\* @roseuid 5879FF36028D

\*/

public void create()

{

}

/\*\*

\* @roseuid 5879FF39013B

\*/

public void update()

{

}

/\*\*

\* @roseuid 5879FF3B00FB

\*/

public void delete()

{

}

/\*\*

\* @roseuid 5879FF3D028B

\*/

public void findAll()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Category.java

public class Category

{

private int id;

private int title;

private int description;

/\*\*

\* @roseuid 58812B5C01D2

\*/

public Category()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\CategoryRepository.java

public class CategoryRepository

{

/\*\*

\* @roseuid 58812B5C030E

\*/

public CategoryRepository()

{

}

/\*\*

\* @roseuid 5879FF5400E9

\*/

public void create()

{

}

/\*\*

\* @roseuid 5879FF560317

\*/

public void update()

{

}

/\*\*

\* @roseuid 5879FF5A03DF

\*/

public void delete()

{

}

/\*\*

\* @roseuid 5879FF5C0387

\*/

public void findAll()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Frontend.java

public interface Frontend

{

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Item.java

public class Item

{

private int id;

private int brand\_id;

private int category\_id;

private int title;

private int description;

private int price;

private int balance;

/\*\*

\* @roseuid 58812B5C0158

\*/

public Item()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\ItemRepository.java

public class ItemRepository

{

/\*\*

\* @roseuid 58812B5C0354

\*/

public ItemRepository()

{

}

/\*\*

\* @roseuid 5879FF7E02D8

\*/

public void create()

{

}

/\*\*

\* @roseuid 5879FF870363

\*/

public void update()

{

}

/\*\*

\* @roseuid 5879FF8D031F

\*/

public void delete()

{

}

/\*\*

\* @roseuid 5879FF8F020F

\*/

public void findById()

{

}

/\*\*

\* @roseuid 5879FF930097

\*/

public void findAll()

{

}

/\*\*

\* @roseuid 5879FF9A014F

\*/

public void findByCategoryId()

{

}

/\*\*

\* @roseuid 5879FFA001B7

\*/

public void findByBrandId()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Order.java

public class Order

{

private int id;

private int user\_id;

private int create\_time;

private int status;

private int archive;

private int name;

/\*\*

\* @roseuid 58812B5C0226

\*/

public Order()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\OrderRepository.java

public class OrderRepository

{

/\*\*

\* @roseuid 58812B5C03E5

\*/

public OrderRepository()

{

}

/\*\*

\* @roseuid 5879FEC3038B

\*/

public void create()

{

}

/\*\*

\* @roseuid 5879FEC60154

\*/

public void update()

{

}

/\*\*

\* @roseuid 5879FEC9032C

\*/

public void findByUserId()

{

}

/\*\*

\* @roseuid 5879FED500D0

\*/

public void findByStatus()

{

}

/\*\*

\* @roseuid 5879FEE000D0

\*/

public void findAll()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\OrderService.java

public class OrderService

{

public Frontend theFrontend;

/\*\*

\* @roseuid 58812B5D00B4

\*/

public OrderService()

{

}

/\*\*

\* @roseuid 5879FD3B0237

\*/

public void addItemToBasket()

{

}

/\*\*

\* @roseuid 5879FD4202CD

\*/

public void removeItemFromBasket()

{

}

/\*\*

\* @roseuid 5879FD4B0129

\*/

public void submitOrder()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\RegistrationService.java

public class RegistrationService

{

public Frontend theFrontend;

/\*\*

\* @roseuid 58812B5D0041

\*/

public RegistrationService()

{

}

/\*\*

\* @roseuid 5879FDA80102

\*/

public void register()

{

}

/\*\*

\* @roseuid 5879FDAB02A5

\*/

public void restorePassword()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\Review.java

public class Review

{

private int id;

private int user\_id;

private int item\_id;

private int title;

private int description;

private int rating;

/\*\*

\* @roseuid 58812B5C025E

\*/

public Review()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\ReviewRepository.java

public class ReviewRepository

{

/\*\*

\* @roseuid 58812B5C0290

\*/

public ReviewRepository()

{

}

/\*\*

\* @roseuid 5879FF1D0163

\*/

public void create()

{

}

/\*\*

\* @roseuid 5879FF2200BF

\*/

public void findByItemId()

{

}

/\*\*

\* @roseuid 5879FF2703B8

\*/

public void findByUserId()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\ReviewService.java

public class ReviewService

{

public Frontend theFrontend;

/\*\*

\* @roseuid 58812B5D0076

\*/

public ReviewService()

{

}

/\*\*

\* @roseuid 5879FC1D029A

\*/

public void createReview()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\StoreFrontService.java

public class StoreFrontService

{

public Frontend theFrontend;

/\*\*

\* @roseuid 58812B5D010E

\*/

public StoreFrontService()

{

}

/\*\*

\* @roseuid 5879FCA10353

\*/

public void getItems()

{

}

/\*\*

\* @roseuid 5879FCA601A6

\*/

public void getItemsByBrand()

{

}

/\*\*

\* @roseuid 5879FCAE03CE

\*/

public void getItemsByCategory()

{

}

/\*\*

\* @roseuid 5879FCF802E6

\*/

public void getItemsByReviewRating()

{

}

/\*\*

\* @roseuid 5879FCB700AE

\*/

public void getItemById()

{

}

/\*\*

\* @roseuid 5879FCBF0186

\*/

public void findItems()

{

}

}

//Source file: C:\\Users\\shnm\\Documents\\BSUIR\_2017\_1\\TRPO\\UserRepository.java

public class UserRepository

{

/\*\*

\* @roseuid 58812B5C00FD

\*/

public UserRepository()

{

}

/\*\*

\* @roseuid 5879FE5A0081

\*/

public void findById()

{

}

/\*\*

\* @roseuid 5879FE5F0144

\*/

public void findByEmail()

{

}

/\*\*

\* @roseuid 5879FEED0329

\*/

public void findAll()

{

}

/\*\*

\* @roseuid 5879FE690254

\*/

public void create()

{

}

/\*\*

\* @roseuid 5879FE750344

\*/

public void update()

{

}

/\*\*

\* @roseuid 5879FE8C00D8

\*/

public void delete()

{

}

}