

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

Профиль подготовки: «Вычислительные машины, комплексы, системы и сети»

БАКАЛАВРСКАЯ РАБОТА

на тему

WEB-ПОРТАЛ
ИНТЕРНЕТ-МАГАЗИНА

Студент

 31.05.2022
(подпись, дата)

Арсентьева Ульяна Сергеевна
(ФИО полностью)

Руководитель

 31.05.2022
(подпись, дата)

Дубравин А.В.
(фамилия, инициалы)

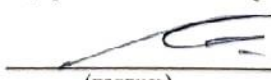
Нормоконтролёр

 31.05.2022
(подпись, дата)

Бычков А.С.
(фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 02.06.2022 г. № 13)

Заведующий кафедрой

 (подпись)

Митрохин М.А.
(фамилия, инициалы)

Работа защищена с отметкой  (протокол заседания ГЭК от

20 июня 2022 № 18)

Секретарь ГЭК

 (подпись)

Гурин Е.И.
(фамилия, инициалы)

Пенза 2022

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
« 26 » апреля 2022г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Арсентьева Ульяна Сергеевна гр. 18ВВ1 факультета Вычислительной техники.

2. Тема работы «Web-портал интернет-магазина».

Тема утверждена приказом ПГУ №357/о от "26" апреля 2022 г.

3. Руководитель бакалаврской работы к.т.н., доцент кафедры ВТ ПГУ Дубравин А.В.

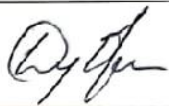
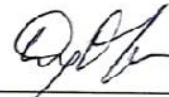
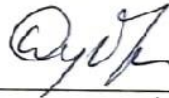
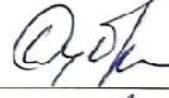
4. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение разработки: Система предназначена для разработки клиентского серверного приложения для приема и обработки онлайн товарооборота.

5. Перечень подлежащих разработке вопросов

В процессе работы определен список требований программного обеспечения, разработаны функции, позволяющие приложению вести обработку товаров, каталогов товаров. В системе предусмотрена работа с системой каталогов товаров и авторизацией пользователей.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	10	26.04.2022-28.05.2022	
2.	Выбор архитектуры приложения	10	29.04.2022-30.05.2022	
3.	Составляющие требования к системе	20	01.05.2022-03.05.2022	
4.	Реализация приложения	10	04.05.2022-08.05.2022	
5.	Отладка системы	10	09.05.2022-15.05.2022	
6.	Моделирование и поиск уязвимостей, исправление ошибок в работе системы	10	16.05.2022-23.05.2022	
7.	Оформление выпускной работы	30	24.05.2022-31.05.2022	

Дата выдачи задания "26" апреля 2022г.

Руководитель бакалаврской работы  Дубравин А.В.
(подпись, дата)

Задание к исполнению принял студент  31.05.2022 Арсентьева У.С.
(подпись, дата)

Работу к защите допустить
Декан факультета  20.06.2022 Фионова Л.Р.
(подпись, дата)

Реферат

Работа содержит 148 листов, 39 рисунков, 12 таблиц, 6 источников, 3 приложения.

WEB, ИНТЕРНЕТ МАГАЗИН, БАЗА ДАННЫХ, *PHP*.

Объектом исследования является разработка *web*-портала интернет-магазина.

Цель работы – разработка программного обеспечения для реализации приложения интернет торговли.

Разработка велась на языке *PHP*.

В процессе разработки был проведен анализ требований к разрабатываемому продукту, анализ существующих современных средств проектирования программных продуктов, анализ аналогичных систем, изучены способы реализации *web* проектов.

В результате выполнения дипломного проекта было разработано программное обеспечение, осуществляющее учет поставок, хранение товара, отправки и всех операций, связанных с заказом, обработкой и поставкой товара. Осуществлено функциональное тестирование разработанного приложения, которое показало корректность его работы.

В пояснительной записке приведен анализ предметной области, обзор технологии разработки и отладки программы.

Графическая часть содержит схемы системы, данных, структуры базы данных, алгоритма работы программных модулей.

					ПГУ 1.090301.03.001 ПЗ				
Изм.	ЛистЛ	№ докум.№	Подпись	Дата	Web-портал интернет-магазина Пояснительная записка	Лит.	Лист	Листов	
Разраб.		Арсентьева У.С.							
Провер.		Дубравин А.В.					2	148	
Н. Контр.		Бычков А.С.				ФВТ, каф. ВТ гр. 18ВВ1			
Утверд.		Митрохин М.А.							

Содержание

Перечень сокращений и обозначений	4
Введение.....	5
1 Web-портал интернет-магазина.....	7
1.1 Анализ предметной области	7
1.2 Обзор технологий веб-приложений	11
1.3 Разработка web-портала интернет-магазина	22
1.4 Технология разработки	59
1.5 Выводы	66
2 Безопасность жизнедеятельности.....	68
Заключение	70
Список используемых источников	71
Приложение А - UML-диаграммы	72
Приложение Б - Листинг приложения	77
Приложение В - Презентация	142

Перечень сокращений и обозначений

ASF (Apache Software Foundation) - web-сервер

HTML(HyperTextMarkupLanguage)–язык гипертекстовой разметки

MVC(Model-View-Controller)–«Модель-Представление-Контроллер»или
«Модель-Вид-Контроллер»

UML (Unified Modeling Language) – унифицирован язык моделирования

БД– базаданных

ООП – объектно-ориентированное программирование

СУБД–система управления базами данных

ЭВМ–электронно-вычислительная машина

Введение

Невозможно представить современный мир без глобального внедрения интернет-технологий в различные сферы жизни, что обуславливает формирование глобальной информационной технологии.

Развиваясь, страна переходит в стадию, где огромное внимание уделяется информатизации, массовому распространению интернет-технологий. И в этом процессе огромная роль принадлежит интернет-магазинам, которые дают мощный импульс для развития и формирования экономики в целом, и в частности цифровой. Поэтому, подчиняясь законам экономики, большинство предприятий стремится овладеть электронным рынком. Интернет-торговля стремится создать собственные бизнес -процессы и модели, которые являются ее сущностью.

Любая точка мира становится доступной благодаря электронному взаимодействию продавцов и покупателей. Преимущество интернет-магазинов также и в том, что они доступны в любое время суток. Достигается высокий уровень взаимодействия продавцов и покупателей путем быстрого удовлетворения их требований и запросов. К тому же интернет-магазины имеют возможность осуществлять быструю реакцию на изменение рынка.

Формирование интернет-торговли взаимосвязано с тенденциями экономики: осуществляется доступ к информации и знаниям, а они перерабатываются в товарные формы. Таким образом, экономические отношения преобразуются на новейшем постиндустриальном уровне.

Интернет торговля — это сфера экономики, включающая в себя финансовые и торговые транзакции, которые осуществляются при помощи компьютерных сетей, а также бизнес-процессы, при помощи которых осуществляются такие транзакции.

Неоспоримо то, что Интернет - важный источник для ведения

бизнеса организаций или компаний.

Интернет используется не только для общения или поиска информации, но и для покупки товаров и услуг. Большинство компаний использует Интернет для рекламы, чтобы успешнее продвигать на рынке свои товары. Но в последнее десятилетие наиболее широко Интернет используют интернет-магазины, чтобы продавать свои товары, и это настолько широко входит в жизнь населения, что все более становится необходимой потребностью.

Цель данной выпускной квалификационной работы заключается в разработке интернет-магазина для торговли, которая занимается продажей вещей.

					<i>ПГУ1.090301.03.001ПЗ</i>	<i>Лист</i>
						6
<i>Изм.</i>	<i>Лист</i>	<i>№докум.</i>	<i>Подпись</i>	<i>Дата</i>		

1 Web-портал интернет-магазина

1.1 Анализ предметной области

Для того чтобы правильно и объективно сформулировать задачи, которые стоят перед разработчиком программного продукта, представлять себе предметную область необходима ясность разработки.

На рисунке 1 представлена схема разработки программного продукта.

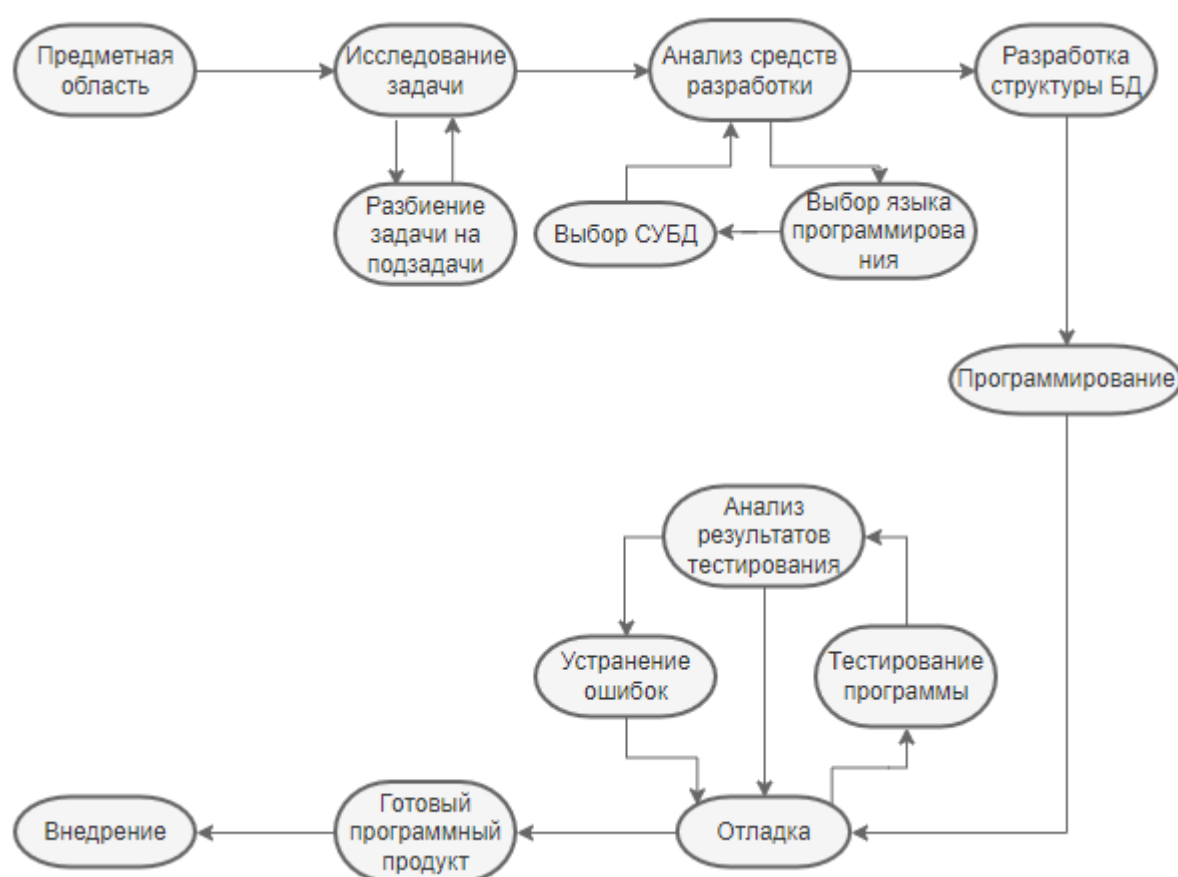


Рисунок 1 – Разработка программного продукта

Первый раздел выпускной квалификационной работы (ВКР) посвящен анализу задач, которые решаются при реализации web-магазина для продажи товаров и услуг в интернет-сети. Задачами ВКР, в соответствии с целью, являются:

- анализ и описание предметной области;
- создание способа и метода решения задачи по созданию интернет-магазина;
- постановка задачи на создание интернет-магазина;
- подбор программных средств для осуществления создания интернет-магазина;
- изложение и описание архитектуры интернет-магазина;
- программная реализация интернет-магазина.

В данной ВКР разработан интернет-магазин, который после установки, размещения его на хостинге и наполнении товарами и услугами будет выполнять свои цели, функции и задачи.

К функциям магазина относятся:

- возможность обслуживания заказа и его редактирования;
- возможность создания, просмотра и редактирования категорий товаров и услуг;
- иметь право выбора необходимых видов и способов их доставки;
- иметь право выбора отобранных товаров и услуг и способов их оплаты;
- присутствие панели управления администратора интернет-магазином.

Основными способами исполнения задачи считаются изучение и анализ аналогичных проектов темы, изучение и разработка интернет-магазина с дальнейшей его программной реализацией.

1.1.1 Обзор аналогичных решений веб-порталов интернет-магазинов

Как показал первоначальный анализ, необходимой информацией для разработки интернет-магазина помимо программной части являются :

- архитектура создаваемого интернет-магазина;

					ПГУ1.090301.03.001ПЗ	Лист
						8
Изм.	Лист	№докум.	Подпись	Дата		

- принцип её работы;
- оформление меню навигации;
- информация о каталогах товаров и услуг;
- расположение, отображение ;
- размещение новостей, связанных с товарами компании определенной тематики;
- ссылки на контактную информацию, информацию о компании и методах её работы.

Помимо всего вышеперечисленного в интернет-магазинах имеются статьи о продаваемых товарах и новости рынка, касающиеся продаваемых товаров и услуг.

Рассмотрим подробно несколько известных интернет-магазинов. На рисунке 2 представлена торговая площадка *Wildberries*.

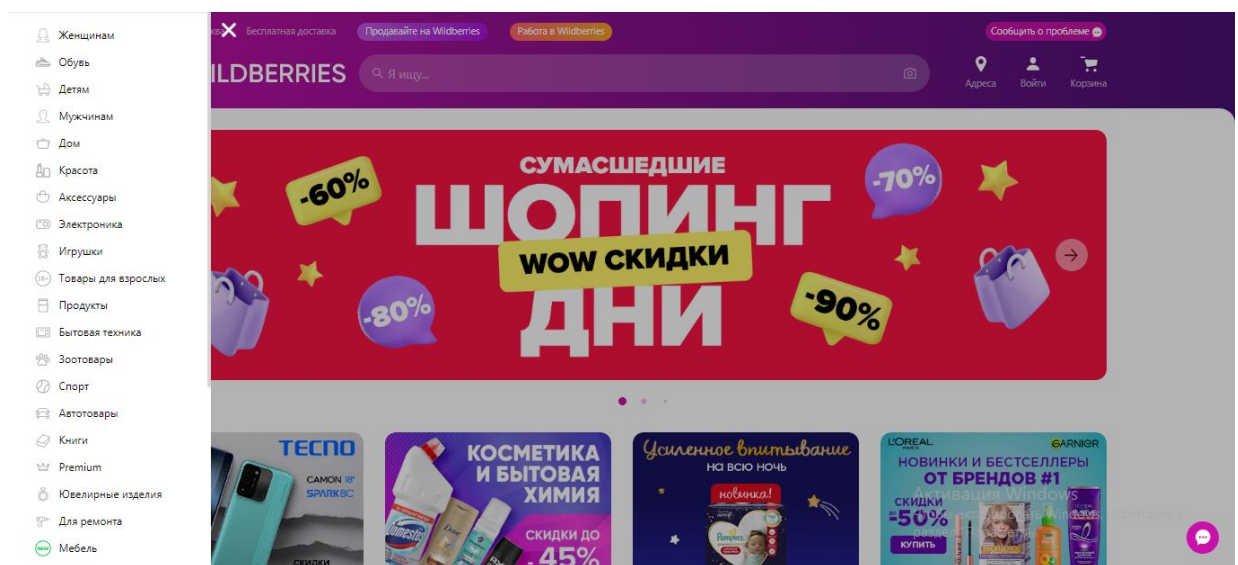


Рисунок 2 – Торговая площадка *Wildberries.ru*

В этом интернет-магазине продаются товары в сети интернет, покупатели – люди из тринадцати стран мира.

На главной странице располагаются новые и рекомендуемые товары, имеется переход по категориям товаров, например: ювелирные изделия,

бытовая техника, игрушки и другие.

Кроме того, на сайте имеется информация о расположении пунктов выдачи товаров, контактные данные, способы оплаты и вся необходимая дополнительная информация, нужная покупателю.

Есть подкатегории для удобства пользователей в навигации и поиске необходимых товаров.

Этот интернет-магазин был основан в России в 2004 году Владиславом и Татьяной Бакальчук.

Нельзя не упомянуть о крупном интернет-магазине, который называется *Aliexpress* (рисунок 3).

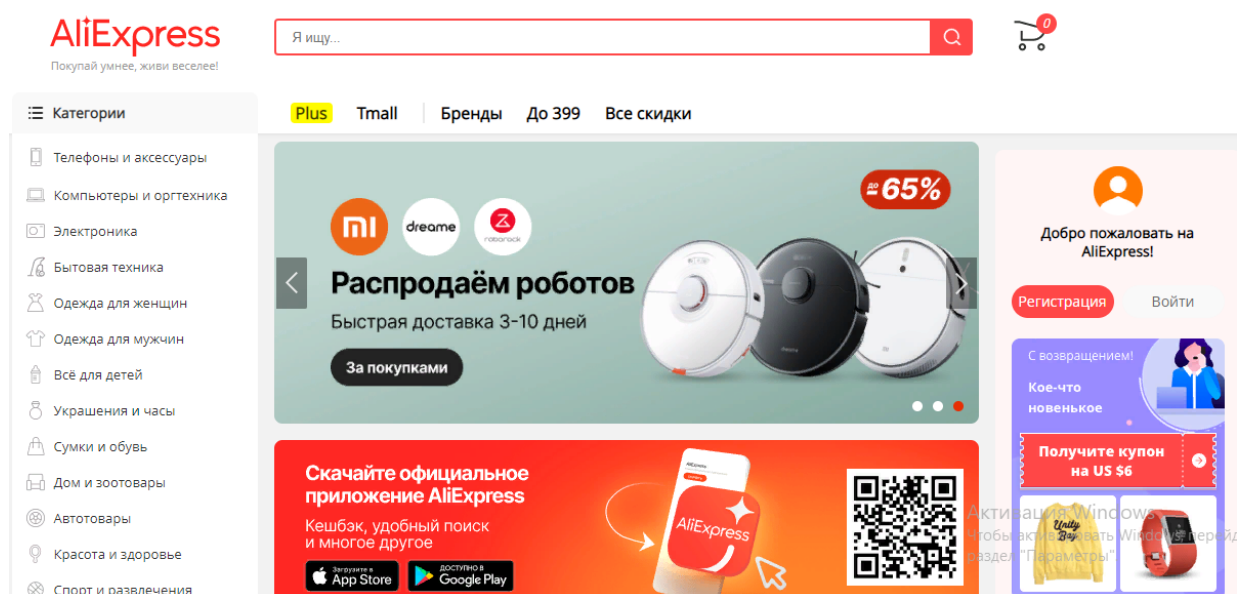


Рисунок 3 – Торговая площадка Aliexpress

Aliexpress – крупнейший интернет-магазин в Китае. В нем продаются товары и услуги по всему миру.

Главная страница отображает меню, в которое входят популярные категории и подкатегории, а также новые, рекомендуемые и популярные товары и услуги. Имеются важные разделы для обратной связи: контактная информация, обучение покупке, другие нужные для покупателя многочисленные разделы.

Главное меню для поиска и выбора товара – меню навигации. Работая с ним, покупатель легко находит товар. Здесь же, на главной странице, клиент авторизуется, и он легко входит в личный кабинет, а также есть корзина, в которую помещается отобранный товар.

При анализе данных интернет-магазинов основными плюсами являются: указаны цены, доступный выбор товара (ассортимент), возможные способы доставки, а также возможность легко овладеть нужной информацией. Кроме того, в интернет-магазине обязательно должны присутствовать каталог, карточка товара, функция оформления заказа, личный кабинет пользователя и администратора и обязательно должны быть оговорены условия возврата товара.

1.2 Обзор технологий веб-приложений

Выбор стека технологий определяется парадигмой. Рассмотрим паттерны – это повторяемые архитектурные конструкции, помогающие разобраться со структурой разрабатываемого приложения.

1.2.1 Выбор паттерна проектирования

Существует три группы паттернов. На рисунке 4 показаны основные паттерны проектирования. Паттерны классов отвечают за создание объектов паттернам классов, а в свою очередь порождающие паттерны объектов передают ответственность другому объекту.

Для составления классов используют структурные паттерны классов, в то же время сборки объектов из частей описывают структурные паттерны объектов.

Для описания алгоритмов и потока управления используют поведенческие паттерны классов, а поведенческие паттерны объектов описывают, как объекты, они принадлежат некоторой группе, вместе работающие и выполняющие задачу, которую ни одному объекту не под силу.



Рисунок4–Паттерныпроектирования

При разработке интернет-магазина важным инструментом выступает паттерн *MVC*.

Model-view-controller(*MVC*, Модель-представление-контроллер) – структура программного обеспечения, в которой имеются: модель данного приложения, пользовательский интерфейс и управляющая логика, они разделены на три отдельные части: представление (*View*), модель (*Model*), поведение-контроллер (*Controller*).

Структура модели *MVC*представлена на рисунке 5.

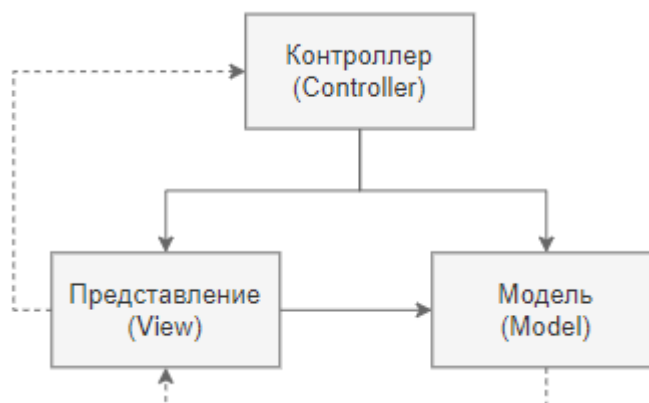


Рисунок5–Структурапаттернапроектирования *MVC*.

Система разделяется при помощи *MVC* на три независимые части: ввод данных, компонент обработки данных и вывод информации. Это видно на диаграмме.

Модель, инкапсулирует ядро данных и основной функционал по их обработке. Также компонент Модель не зависит от процесса ввода или вывода данных. Компонент вывода Вид может представлять собой несколько взаимосвязанных областей, например, различные таблицы и поля форм, в которых отображается информация.

В функции Контроллера входит мониторинг за событиями, возникающими в результате действий пользователя (изменение положения курсора мыши, нажатие кнопки или ввод данных в текстовое поле).

Регистрируемые события транслируются в различные запросы, направляемые компонентам Модели или объектам, ответственным за отображение данных. Разделение модели от представления данных позволяет независимо использовать различные компоненты для отображения информации. Таким образом, если пользователь через Контроллер внесет изменения в Модель данных, то информация, представленная одним или несколькими визуальными компонентами, будет автоматически откорректирована в соответствии с произошедшими изменениями.

Диаграмма взаимодействия *MVC* представлена на рисунке 6.

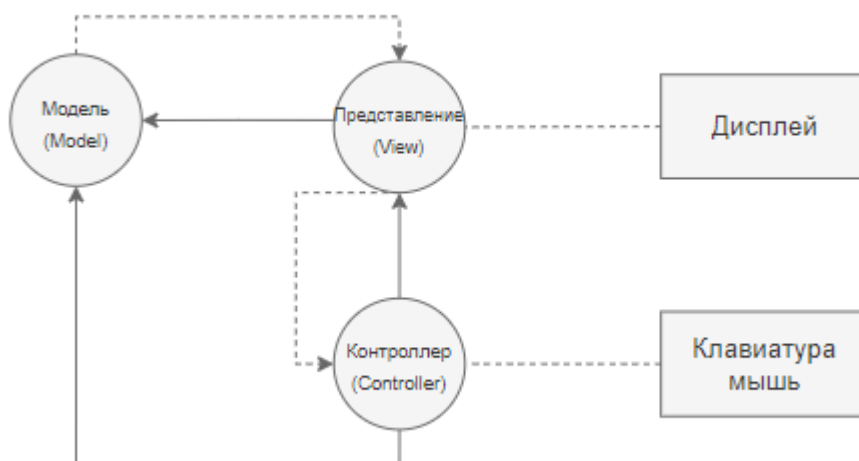


Рисунок 6 - Диаграмма взаимодействия *MVC*

И представление, и поведение зависят от модели. Сама же она не зависит от них. Разделяя вид и модель, *MVC* отделяет их друг от друга, устанавливая между ними протокол взаимодействия «подписка/оповещение», не переделывая модель. Можно отобразить новый вид, не переделывая модель.

1.2.2 Обзор веб-сервера *Apache*

Действительно, нет такой области человеческой жизни, куда бы ни проник за последние десятилетия Интернет. Распространение его многочисленно: это и государственные, и общественные организации, он также доступен и частным лицам.

Эта система имеет множество сервисов, программ, разработок, притом постоянно работающих. И это не только *WWW* и *Web*-браузеры.

Web-сервер – это программа, которая отвечает и за прием, и за переработку сообщений, которые приходят на 80-й порт (стандартная настройка; можно, конечно, настроиться и на любой другой порт), и работающая с протоколом *HTTP (HypertextTransferProtocol)*. Основным

для WWW. является именно этот протокол. Он основан на принципе «запрос-ответ» и отвечает за набор правил для обмена данными. Идя от клиента к серверу, он содержит служебную информацию о типе запроса (данные, заголовок, форма), заголовок запроса (типы файлов, авторизация, версия клиента, адрес, активация ссылки на этот ресурс и сам адрес), а также запрос данных. От сервера к клиенту идет ответ, в нем имеется служебный код (число), который показывает состояние обработки запроса, ответный заголовок (версию сервера, дату, длину и тип данных) и сами данные.

Сейчас в мире существует несколько десятков Web-серверов, но популярностью пользуются лишь несколько. Один из самых распространенных веб-серверов – Apache от ASF (Apache Software Foundation), это самый распространенный Web-сервер в мире. Популярность Apache в мире повысила поддержка Windows NT, которая появилась в последней версии. Надежность, мобильность и огромный набор функций и дополнительных модулей привлекают многих клиентов. А бесплатное распространение позволяет экономить средства, быстро исправлять ошибки и вносить в код программы нужные дополнения.

Версия 1.1.x считается одной из самых мобильных, потому что основными свойствами Apache являются поддержка кросс-платформ, протокол HTTP/1 и HTTP/2, модульная структура, защита, запись log-файлов. А стандартная поставка и огромное количество модулей расширяет возможности Apache.

Открытый код, быстрое распространение дают возможность создавать наиболее защищенные модули. Любой коммерческий сервер проигрывает в этом с Apache. Единственный пробел – у Apache нет красивой программы для установки и для управления. Поэтому приходится применять командную строку во время установки.

Устанавливая под UNIX, приходится корректировать в include-файлы.

Работая, приходится иметь дело с текстовыми конфигурационными файлами, а не с оконной программой.

Но все это не делает *Apache* менее известным, потому что она имеет преимущества:

- свободный, бесплатный код;
- операционные системы;
- создает несколько *log*-файлов;
- под *WindowsNT* запускается как сервис и/или программа, под *UNIX* может запускаться из *inetd*;
- может иметь настройку на несколько портов;
- поддерживает *WindowsCGI*, *HTTP*, в том числе и *HTTPPUT*;
- может самостоятельно ответить при изменении документа;
- поддерживает *MicrosoftISAPI*;
- может произвести запрет с конкретных адресов, к конкретным документам, запретить запуск *CGI*-скриптов, может произвести запрет доступа конкретным пользователям;
- может поставляться вместе с полным исходным кодом;
- может поддерживать другие протоколы (*ftp*, *telnet*);
- может иметь настройку на пользовательские директории;
- содержит модуль *proxy*.

Данные *Webcompare* (<http://webcompare.internet.com>) и *Swatch* (<http://serverwatch.internet.com>) для сервера *Apache*:

- 58,85% – процент на рынке *Web*-серверов;
- количество клиентов – 3 570 377.

Рейтинг (по пятибалльной системе):

- надежность — 5;
- производительность — 5;

- простота использования — 3;
- техническая поддержка — 4,5.

1.2.3 Выбор языка программирования

Для генерации *HTML*-страниц на веб-сервере и работы с базами данных существует *PHP*— интерпретируемый скриптовый язык программирования. На сегодняшний день в области веб-программирования *PHP* является одним из самых популярных и распространённых технологий, из-за его простоты, скорости и владения богатыми функциями. *PHP* распространяется свободно. Язык похож на синтаксис *C++*. Его используют многие представители сетевого хостинга. Название "*PHP*" представляет собой самоповторяющуюся аббревиатуру "*PHP: HypertextPreprocessor*", или "*PersonalHomePage*". Он существует для более легкой работы на веб-страницах, и *PHP* был создан в качестве надстройки над *Perl*. Он много раз менялся, потому что имеет возможность расширения ядра, а это самая сильная сторона *PHP*. Интерфейс написания расширений привлек к *PHP* много других разработчиков, разрабатывающих свои модули. Так что у *PHP* появилась возможность владеть огромным количеством баз данных, протоколов, поддерживать большое число *API*. Также *PHP* сотрудничает с ООП (деструкторы, открытые, закрытые и защищённые члены и методы, *final*- члены и методы, интерфейсы и клонирование объектов). *PHP* поддерживает *XML*.

Существует *JAVA*-объектно-ориентированный язык программирования, который разработала компания *Sun Microsystems*, этот язык имеет строгую систему типов, реализует многопоточность, запрещает множественное наследование, содержит самостоятельно запускающийся «сборщик мусора».

Основа языка *Java*- синтаксис языка *C++*. А уменьшение «избыточности» языка позволяет более быстро проникать в код, написанный иным программистом.

Сравнение языков программирования представлено в таблице 1.

Таблица 1 - Сравнение языков программирования

<i>PHP</i>	<i>Java</i>
Разделение объявления и определения	Нет
<i>Const</i>	есть <i>final</i>
>>-нет разницы между логическими арифметическим сдвигом	>>-арифметический >>>-логический
0,1	false,true
строки как таковые отсутствуют	<i>ReadOnly</i> , в памяти под новую строку формируется место, старая исчезает из области видимости
<i>goto</i>	нет <i>goto</i> , зато есть <i>break</i> <метка>

Архитектура *Java* представлена на рисунке 7.

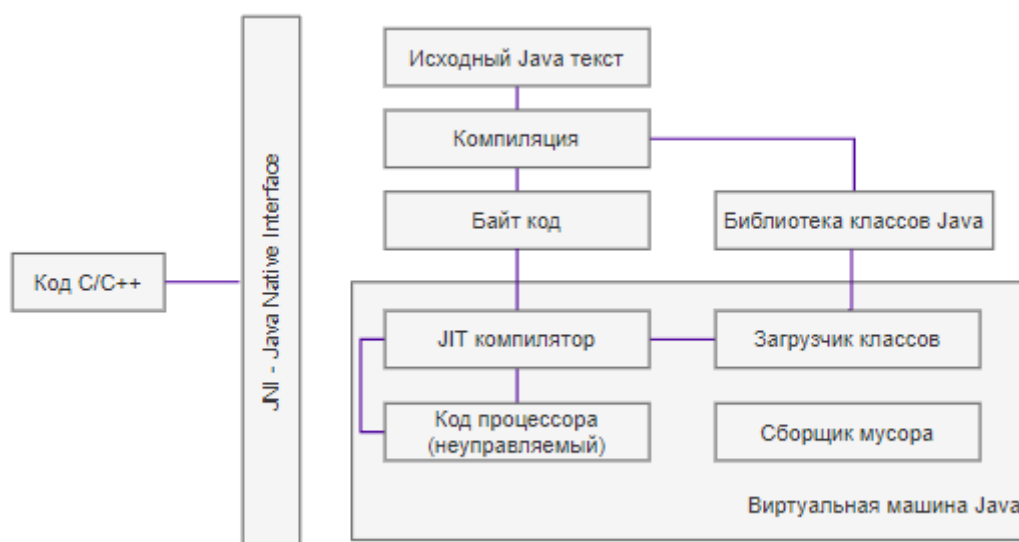


Рисунок 7 – Архитектура *Java*

Приложения *Java* обычно программируются в специальный байт-код, поэтому они работают на любой виртуальной *Java*-машине (*JVM*), не завися от компьютерной структуры. Обязательно нужен сборщик мусора. *Java* поддерживает оперативного и интерактивного наполнения *WorldWideWeb*, согласно «*just-in-time*» доступа к программным ресурсам. *Java* имеет набор *API*. При исполнении имеется менеджер безопасности.

Особенности *Java*:

- мобильный код – это одна программа, в ней много вычислительных платформ;
- безопасный код – в ее язык встроена защита от неразрешенного доступа к ресурсам компьютера;
- устойчивый код – при проектировании может обнаружить ошибки;
- управление памятью не имеет ошибок.

Язык *Java* для числовых операций может иметь типы данных, не зависящие от платформы, четко определённую ИИЭРА (Институт по электротехнике и радиоэлектронике) арифметику, проверку границ массива.

Язык *Java* обладает встроенной поддержкой многопоточности, делает динамическое связывание подклассов с методами, которые отменяют или добавляют возможности функций.

У платформы *Java* имеются две основных части: *JavaVirtualMachine* (виртуальная *Java*-машина) и *JavaAPI* (прикладной программный интерфейс *Java*).

Работа с базами данных поддерживается приложениями на *Java* через интерфейс *JavaDatabaseConnectivity* (*JDBC*). *JDBC* (*JavaDataBaseConnectivity* — соединение с базами данных на *Java*) — платформенно-независимый промышленный стандарт взаимодействия *Java*-приложений с различными СУБД, реализованный в виде пакета *java.sql*, входящего в состав *JavaSE*.

JDBC основывается на концепции так называемых драйверов, которые

позволяют получать соединение с базой данных по специально описанному *URL*. Драйверы могут загружаться автоматически (во время работы программы). Когда драйвер загрузится, он сам регистрирует себя и вызывается автоматически, когда программа требует *URL*, содержащий протокол, за который драйвер «отвечает». Модель *JDBC* базируется на принципах *ODBC*: программа имеет дело с диспетчером драйверов *JDBC*, который выбирает подходящий в данном случае драйвер (например, драйвер *Oracle*) или мост (например, мост *JDBC/ODBC*).

Модель *JDBC* работает одинаково независимо от платформы и от поставщика данных. Вся работа с БД осуществляется с помощью *SQL* и *Java*. *DBC API* содержит два основных типа интерфейсов: первый - для разработчиков приложений и второй (более низкого уровня) - для разработчиков драйверов.

Класс, реализующий интерфейс *java.sql.Connection* выбирает подходящий в данном случае драйвер (например, драйвер *Oracle*) или мост (например, мост *JDBC/ODBC*) описывает соединение с базой данных.

Соединяясь с базой данных, можно создавать объекты типа *Statement*, служащие для исполнения запросов к базе данных на языке *SQL*.

Виды типов различаются по назначению:

- *java.sql.Statement* — *Statement* общегоназначения;
- *java.sql.PreparedStatement* — *Statement*, служащий для выполнения запросов, содержащих подставляемые параметры (обозначаются символом '?' в теле запроса);
- *java.sql.CallableStatement* — *Statement*, предназначенный для вызова хранимых процедур.

Интерфейс *java.sql.ResultSet* позволяет легко обрабатывать результаты запроса.

Классы *JDBC* располагаются в пакетах *java.sql* и *javax.sql*. Подключение к БД:

```
Connection con = DriverManager.getConnection(url, username, password);
```

Выполнение запроса:

```
Statement st = con.createStatement(); ResultSet rs = st.executeQuery("SELECT ...");
```

Если указываются особые параметры при создании *Statement*, то *ResultSet* может делаться обновляемым (то есть данные изменения отражаются в БД).

Каждая *SQL*-команда может выполняться в своей транзакции по умолчанию, но возможно отключение:

```
con.setAutoCommit(false); con.commit(); con.rollback();
```

Преимуществами *JDBC* считают:

- лёгкость разработки: разработчик может не знать специфики базы данных, с которой работает;
- код не меняется, если компания переходит на другую базу данных;
- не нужна громоздкая клиентская программа;
- подключение к любой базе данных через легко описываемый *URL*.

В *web*-программировании *Java*-платформа используется широко, потому что у нее есть преимущества, которые связаны с администрированием корпоративных компьютерных систем, а также может создавать: пользовательское соглашение, может запускаться с определенной скоростью на других программируемых языках, таких как *C++*, но значительно быстрее, чем *JavaScript*, может переносить работу с сервера к клиенту, что помогает найти Интернет-решение со многими числом пользователями.

Чтобы упростить и расширить программирование на *Java*, созданы различные фреймворки (шаблоны), такие как *Spring. Framework*, *Turbine* и другие. Сейчас детально рассмотрим *Apache Struts*, потому что именно он будет использоваться в разработке выпускной квалификационной работы.

1.3 Разработка web-портала интернет-магазина

1.3.1 Формализация задачи

В дипломном проекте разработан web-портал интернет-магазина, для того чтобы создать цикл товарооборота и его учет. Имеется прямая необходимость для включения программного обеспечения в состав системы средства интернет-коммерции

Разрабатывая программное обеспечение, необходимо помнить о процессе движения товарно-материальных ценностей в торговом предприятии. Анализируя, необходимо также иметь в виду формы входных и выходных документов, которые самым прямым образом влияют на разработку подсистемы хранения данных.

На рисунках 8 и 9 представлены функции покупателя и администратора.



Рисунок8 –UML-диаграмма прецедентов пользователя

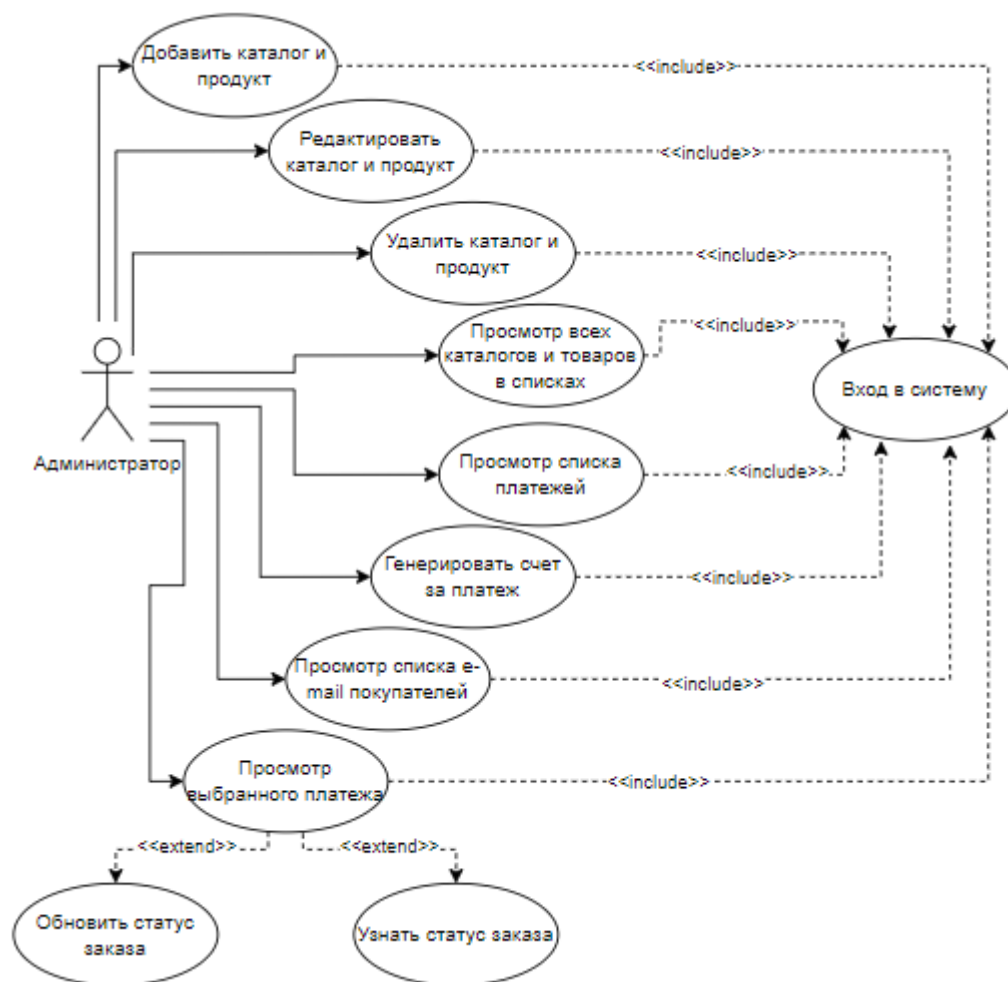


Рисунок 9 –UML-диаграмма прецедентов администратора

Необходимо знать, что является первичными документами для учета движения товарно-материальных ценностей. Входными (первичными) документами служат:

- накладная и счет-фактура на поставку товара от поставщика;
- накладная и счет-фактура на поставку товара покупателю;
- счет на поставку;
- счет на продажу;
- договор поставки или купли-продажи на товар от поставщика;
- договор поставки;
- доверенность на получение товара от поставщика.

По каждому поставщику и покупателю, а также по отдельным наименованиям (или группам, при большом ассортименте) на товар должны формироваться выходные документы. К ним относятся:

- ведомость поступления товара на склад, сгруппированная по поставщикам, приходным документам и основным счетам;
- ведомость расхода товара со склада, сгруппированная по получателям и расходным документам, при этом в расход по складу включаются все материалы, отпущенные со склада;
- оборотная ведомость движения материалов по складу;
- сводная ведомость движения материалов (в суммовом выражении);
- книга покупок и книга продаж с указанием суммы поступивших товаров по установленной форме;
- книга учета доверенностей на получение товаров от поставщиков;
- кассовая книга по установленной форме (учет наличных денежных средств от поступивших и проданных товаров).

Все программное обеспечение можно разделить на несколько частей:

- база данных, где хранятся все нужные данные для четкого и правильного товарооборота;
- программные модули, выполняющие расчеты;
- визуальный интерфейс взаимодействия с пользователем.

От каждого правильно реализованного из этих компонентов зависит качество работы программного комплекса.

Такая правильная программная система является надежной и нужной основой базы данных.

Для ее управления используют системы управления базами данных (СУБД). Поэтому необходимым условием является при проектировании использование для ее управления системы управления базами данных СУБД MySQL.

Для сбора, хранения, поиска, обработки и выдачи данных по запросам конечных пользователей предназначена СУБД — совокупность информационных массивов, технических, программных и языковых средств.

Клиент, который обращается к системе для получения необходимых ему данных, называется конечным. Поэтому важно использовать СУБД для упорядочения хранения данных и исключения потери.

Для обеспечения сохранности базы данных СУБД выполняет необходимые функции:

- вводит данные, используя формы ввода;
- формирует отчетные документы;
- выводит документы на экран, а затем печатает их и отчеты;
- осуществляет поиск и отбор данных по запросам пользователя.

Когда разрабатывается программная система, то основной задачей является проектирование основных компонентов базы данных, так как они связывают части функциональных частей системы, потому что входные данные одной из частей системы являются входными данными для другой. При проектировании необходимо учитывать три главные области:

– таблицы, представления, индексы и хранимые процедуры – это конкретные объекты, которые после проектирования будут отражены в базе данных;

– конкретные экранные формы, отчеты и программы после проектирования будут присутствовать в базе данных, чтобы обеспечивать их выполнение;

– при их проектировании нужно иметь в виду конкретную среду:

- 1) топологию сети;
- 2) конфигурацию аппаратных средств;
- 3) использование реляционной СУБД;

4) использование параллельной обработки и распределенной архитектуры базы данных.

Нужно учитывать, что главной задачей при проектировании базы данных бывает задача разделения различных уровней представления и описания данных. Существуют несколько уровней представления данных:

- для обеспечения информационных потребителей приложения существует совокупность информационных моделей, описывающих необходимую часть предметной области;

- концептуальное представление, системно независимое от конкретной СУБД, операционной системы и аппаратного обеспечения ЭВМ, изображает все информационные потребности всех приложений;

- существующее логическое представление - это представление в конкретной СУБД. Определенный тип модели данных, (сетевой, иерархический или реляционный) поддерживается современными типами ЭВМ;

- внутреннее представление – это представление, которое дает понятие о данных, помещенных в среду хранения и о средствах доступа к ним. При этом должны быть определены для хранимых записей и их физической последовательности, индексов, хранимых полей различные типы.

При построении концептуальных моделей предметной области, решаются следующие задачи:

- определяются цели, необходимые администратору и принимается решения о создании информационной системы;

- на их основе определяются потенциальные пользователи системы;

- происходит выявление и устранение конфликтных ситуаций, а также превращение их в концептуальное представление;

- появляется одна из логических моделей данных, записанная на языке, позволяющем легко отображать это представление.

Сформированная концептуальная модель базы данных включает в себя:

- осуществление равенства функциональной деятельности данной предметной области;
- осуществление равенства объектов и формирует последовательность событий, осуществляющих эту функциональную деятельность;
- осуществление взаимосвязи между сущностями.

Необходима исходная информация о предметной области, чтобы произвести проектирование структуры базы данных и создать искусственный язык и графические средства для описания предметной области.

Для автоматизации проектирования исходные данные представляются в виде совокупности графических схем. Это наиболее простое представление для восприятия и анализа, поэтому существует частое использование отображения модели "объект - свойство - отношение". В дипломном проекте осуществляется использование модели «сущность-связь» (ER- модель) как наиболее популярная. В приложении Б представлена схема разработанной базы данных.

Визуальная система пользователя может осуществлять взаимодействие с СУБД средствами графических схем.

В СУБД MySQL существуют специальных клиентские плагины (HeidiSQL).

Используются также формы для отображения данных в базе данных, кроме того их можно использовать как кнопочную форму, чтобы открывать другие формы или отчеты базы данных, использовать как диалоговое окно для ввода данных и для вывода, а также их можно открывать в режиме сводной таблицы или в режиме диаграммы. Можно изменять заголовки строк и столбцов, а также применять фильтры к полям.

В связи с новым расположением данных при каждом изменении макета сводная форма немедленно начинает выполнять вычисления заново. Кроме того, можно в режиме сводной таблицы отсматривать исходные данные или

сводные данные, можно правильно расположить поля в областях фильтра, строк, столбцов и данных. Кроме этого можно упорядочить данные, выбрать тип диаграммы, расположив поля в областях фильтра, рядов, категорий и данных.

UML-диаграмма последовательностей приведена на рисунке 10.

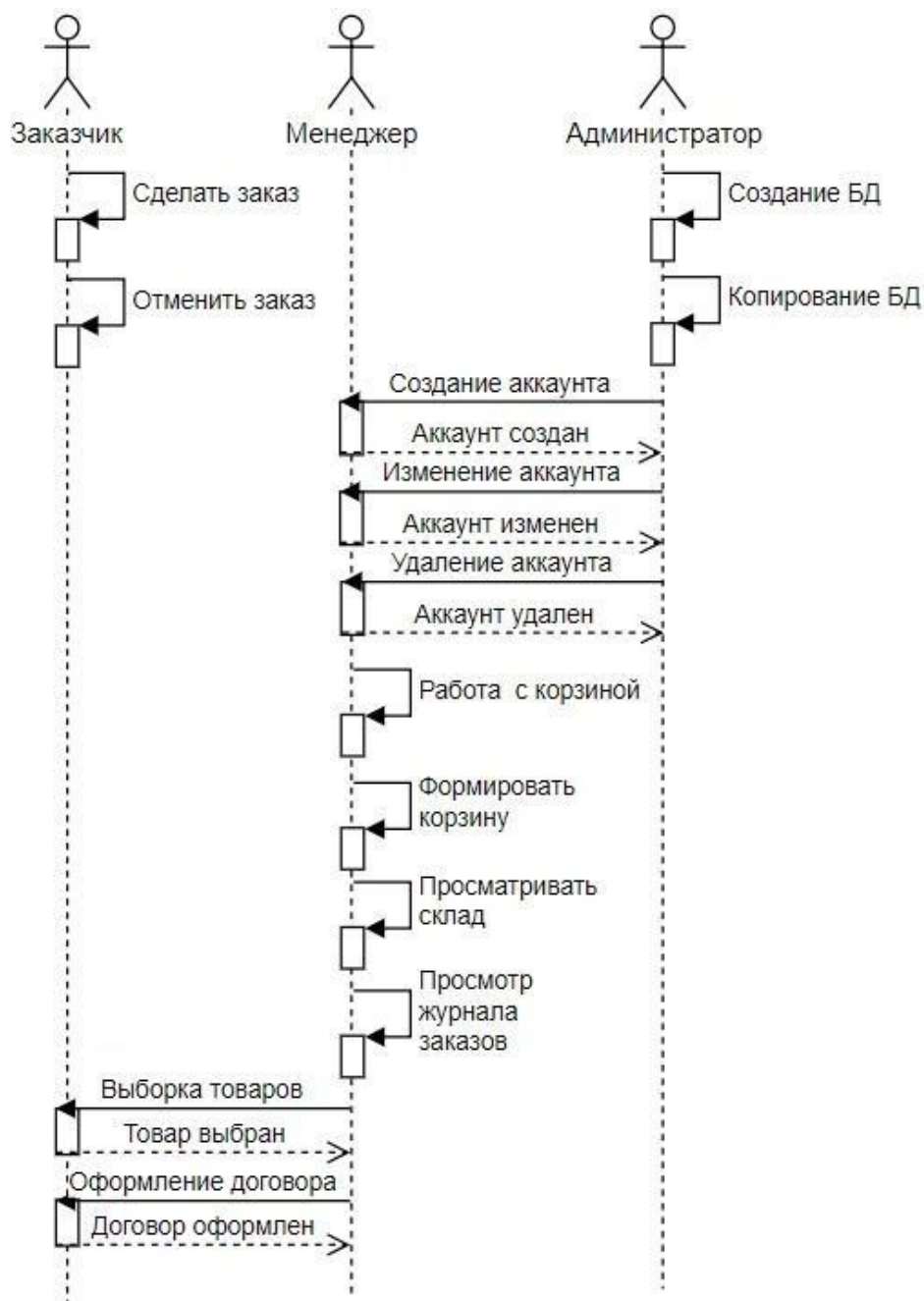


Рисунок 10 - *UML*-диаграмма последовательности

1.3.2 Архитектура программно-аппаратного комплекса

На базе платформы *Intel*. этот программный комплекс нужен для развертывания. Рекомендованная конфигурация для сетевой работы *ATPentiumIV*– 2 ГГц, с объемом оперативной памяти не менее 2 ГБ (рекомендуется 4 ГБ), постоянного запоминающего устройства не менее 500 ГБ, видеокарта с объемом памяти не менее 2 ГБ, дисковод *CD-ROM*, *USB* накопитель. Эти системные требования применяются в связи с требованиями ОС.

Для однопользовательской работы необходима данная конфигурация – *IBM*-совместимый компьютере типа *IBMPC/ATPentiumIVCeleron*– 1,7 ГГц, с объемом оперативной памяти не менее 2 ГБ (рекомендуется 4 ГБ), постоянного запоминающего устройства не менее 10 ГБ, видеокарта с объемом памяти не менее 128 Мб, дисковод *CD-ROM*, *USB* накопитель, монитор 17'', средства мультимедиа (звуковая карта, колонки) для распечатки отчетной информации формата А4 рекомендуется любой тип принтера. Для работы клиента необходима ОС *Win32 - Windows 8* и наличие СУБД *MySQL*.

Сетевая версия программного комплекса может быть установлена на *WEB*серверах с ОС *Windows*или ОС *UNIX*, использоваться на бездисковых рабочих местах, в которых имеются средства мультимедиа.

1.3.3 Разработка структуры данных

Существует структура базы данных в среде *MySQLHeidiSQL* для учета движения товаров и структуры приходных, расходных и расчетно-платежных документов. Эта базу данных отражает следующие таблицы 2-6:

Таблица 2 -*Categories* - предназначена для хранения категорий товаров

Наименование поля	Назначение
<i>ID_категории</i>	Идентификатор категории
Название	Название категории
<i>IsVisible</i>	Переменная, в зависимости от значения которой, категория будет (1) или не будет (0) отображаться в списке для просмотра покупателями

Таблица 3 -*Products* - предназначена для хранения товаров

Наименование поля	Назначение
<i>ID_товара</i>	Идентификатор товара
Название	Название товара
Производитель	Производитель товара
<i>ID_категории</i>	Идентификатор категории, которой принадлежит текущий товар
Фото	Путь к фотографии товара
Цена	Цена товара (в рублях)
На_складе	Количество единиц текущего товара на складе
Описание	Текстовое описание товара
Скидка	Процент скидки текущего товара (0-100%)
<i>IsRecommended</i>	Переменная, в зависимости от значения которой, товар будет (1) или не будет (0) отмечен, как рекомендованный
Дата_добавления	Дата добавления товара в базу данных

Таблица 4 -*Customers* - предназначена для хранения списка покупателей

Наименование поля	Назначение
<i>ID_покупателя</i>	Идентификатор покупателя
Фамилия	Фамилия покупателя
Имя	Имя покупателя
Отчество	Отчество покупателя
Адрес	Адрес покупателя
Логин	Логин аккаунта покупателя
Пароль	Пароль от аккаунта покупателя
Телефон	Номер телефона покупателя
Почта	Адрес электронной почты покупателя
Дата_регистрации	Дата регистрации покупателя

Таблица 5 -*ShoppingCart* - предназначена для хранения товаров в корзине покупателей

Наименование поля	Назначение
<i>ID_корзины</i>	Идентификатор тройки значений (Покупатель, Товар, Количество)
<i>ID_покупателя</i>	Идентификатор покупателя, добавивший товар в корзину
<i>ID_товара</i>	Идентификатор товара, который был добавлен в корзину
Количество	Количество единиц выбранного товара в корзине

Таблица 6 -*Orders* - предназначена для хранения заказов покупателей

Наименование поля	Назначение
<i>ID_заказа</i>	Идентификатор заказа
<i>ID_покупателя</i>	Идентификатор покупателя, совершившего заказ
Товары	Множество пар значений (<i>ID_товара</i> , Количество), представленных в формате JSON

На рисунке 11 приведена структура базы данных.

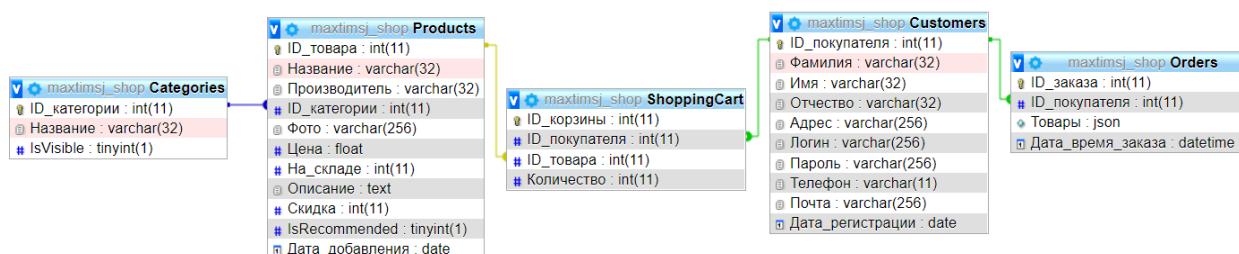


Рисунок 11–Структурабазы данных

1.3.4 Разработка программной части

Программное обеспечение осуществлено на основе паттерна MVC (Модель – Вид -Контроллер):

- модель (*Model*), реагируя на команды контроллера, предоставляет данные и изменяет своё состояние;

- представление (*View*), реагируя на изменение модели, отвечает за отображение данных модели пользователю;

- контроллер (*Controller*), оповещая модель о необходимости изменений, перерабатывает действия пользователя.

На рисунке 12 приведено сформированное приложение MVC.

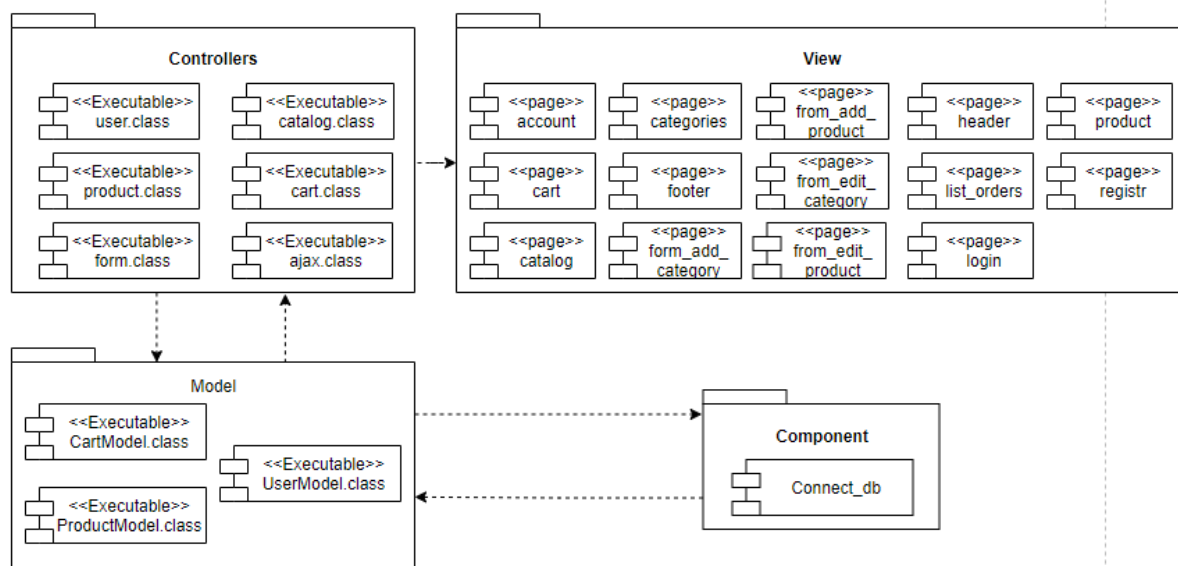


Рисунок 12 - Сформированное приложение MVC

Разработанное приложение состоит из следующих цепочек MVC–таблицы 7-12:

Таблица 7-Авторизация

Модель	Представление	Контроллер
Данные отправляются на сервер для авторизации пользователя. Производится отправка запроса на сервер для выхода.	Выводится экран авторизации.	Вводится логин и пароль. Производится выход из пользователя.

Таблица 8-База данных

Модель	Представление	Контроллер
Производится запрос списка <i>backup</i> 'ов с сервера. Отправляется запрос на создание <i>backup</i> 'а. Отправляется запрос на удаление <i>backup</i> 'а	Выводится список <i>backup</i> 'ов.	Создается, скачивается и удаляется <i>backup</i> . Производится поиск по имени и дате.

Таблица 9- Пользователи

Модель	Представление	Контроллер
Производится отправка запроса списка на создание, изменение и удаление пользователя.	Выводится список пользователей.	Изменяется, создается и удаляется пользователь. Производится поиск по имени и дате.

Таблица 10- Склад

Модель	Представление	Контроллер
Производится отправка списка вешей корзины на сервер.	Выводится список товаров.	Производится добавление товара в корзину и изменяется количество товаров в корзине.

Таблица 11- Заказы

Модель	Представление	Контроллер
Производится отправка новых состояний и запросов на повторзаказа на сервер.	Выводится список заказов.	Изменяется состояние заказов и их повтор

Таблица 12- Заказы

Модель	Представление	Контроллер
Производится отправка нового количества товаров, измененных данных заказчика и запросов на создание заказа на сервер.	Выводится список товаров заказа.	Изменяется данные заказчика, количество товаров.

1.3.5 Описание функций бизнес логики

Ниже описаны функции бизнес-логики приложения.

Вход пользователя в систему реализует и выдаёт сессию функция `[set]Login`, входными данными является логин пользователя и пароль, выходных данных нет.

Выход пользователя из системы и завершение сессии реализует функция `[get]Logout`, входных и выходных данных нет.

Вывод всех путей, доступных группе пользователя, выполняет функция `[get]Routes`, группа клиента выбирается из созданной сессии, входных данных нет.

Создание резервной копии базы данных, выполняет функция `[set]Backup`, входных и выходных данных нет.

Вывод всех созданных резервных копий баз данных выполняет

функция *[get]Backups*, входных данных нет.

Удаление резервной копии выполняет функция *[delete]Backup*, идентификационный номер резервной копии является входным параметром, выходных данных нет.

Вывод данных о пользователях выполняет функция *[get]Users*, входные данные отсутствуют, выходными данными являются имя, фамилия, отчество, логин, наименование группы являются выходными данными.

Вывод данных о текущем клиенте выполняет функция *[get]User*, входные данные отсутствуют, идентификационный номер, логин, фамилия, имя, отчество, пути и наименование группы являются выходными данными.

Создание пользователя выполняет функция *[set]User*, входными данными являются: логин, пароль, имя, фамилия, отчество, группу, выходных данных нет.

Изменение данных пользователя, выполняет функция *[change]User*, идентификационный номер пользователя, логин, пароль, имя, фамилия, отчество, группа являются входными данными.

Удаление пользователя выполняет функция *[delete]User*, идентификационный номер является входными данными, выходные параметры отсутствуют.

Вывод всех существующих категорий товаров выполняет функция *[get]Categories*.

Вывод категории выполняет функция *[get]Category*, идентификационный номер является входными данными.

Создание категории выполняет функция *[set]Category*, наименование категории является входными параметрами, выходные параметры отсутствуют.

Изменение категории выполняет функция *[change]Category*, идентификационный номер и наименование категории являются входными параметрами, выходных данных нет.

Изм.	Лист	№ докум.	Подпись	Дата	
------	------	----------	---------	------	--

Удаление категории выполняет функция *[delete]Category*, входными данными является идентификационный номер категории, выходных данных нет.

Получение всех товаров выполняет функция *[get]Products*, входных данных нет.

Получения списка всех товаров выполняет функция *[get]Product* Получение данных о товаре, идентификационный номер являются выходным параметром .

Получение данных о товарах определённой категории выполняет функция *[get]ProductsByCategory*, идентификационный номер категории товаров является входными данными .

Добавление товара выполняет функция *[set]Product*, входными данными являются: наименование, идентификационный номер категории, идентификационный номер меры, цена, количество на складе, идентификационный номер бренда, идентификационный номер производителя, выходных данных нет.

Изменение данных пользователя выполняет функция *[change]Product*, идентификационный номер товара, наименование, идентификационный номер категории, идентификационный номер меры, цена, количество на складе, идентификационный номер бренда, идентификационный номер производителя являются входными данными, выходных данных нет.

Удаление товара выполняет функция *[delete]Product* идентификационный номер товара является входными данными , выходных данных нет.

Получение цены товара выполняет функция *[get]Price*, идентификационный номер продукта является выходными данными, а цена товара. является выходным параметром.

Получение списка производителей выполняет функция *[get]Producers*, входных данных нет.

Получение данных о производителе выполняет функция *[get]Producer* идентификационный номер производителя является входными данными.

Создание производителя выполняет функция *[set]Producer*, наименование, телефон, адрес являются входными данными, выходных данных нет.

Удаление производителя выполняет функция *[delete]Producer*, идентификационный номер производителя является входными параметрами, выходных данных нет.

Изменение данных производителя, выполняет функция *[change]Producer* идентификационный номер производителя, наименование, телефон, адрес являются входными данными, выходных данных нет.

Вывод всех брендов выполняет функция *[get]Brands*, входных данных нет.

Создание бренда выполняет функция *[set]Brand*, идентификационный номер производителя и наименование являются входными параметрами .

Изменение бренда выполняет функция *[change]Brand*, идентификационный номер производителя, наименование являются входными параметрами, выходных данных нет.

Удаление бренда выполняет функция *[delete]Brand*, идентификационный номер бренда является входным параметром , выходных параметров нет.

Получение списка брендов производителя выполняет функция *[get]BrandByProducer*, идентификационный номер производителя является входным параметром. Установку статуса заказа, выполняет функция *[set]Status*, идентификационный номер заказа и идентификационный номер статуса являются входными параметрами, выходных параметров нет.

Установку привилегии выполняет функция *[set]Privilege*, идентификационный номер пользователя и идентификационный номер категории являются входными параметрами является, выходных данных нет.

Удаление привилегии пользователя выполняет функция *[delete]PrivilegeUser*, идентификационный номер пользователя является входными параметрами, выходных параметров нет.

Изменение привилегий пользователя выполняет функция *[change]Privilege*, идентификационный номер пользователя, идентификационный номер категории, идентификационный номер привилегии являются входными параметрами.

Вывод привилегий по пользователю выполняет *[get]PrivilegeByUser*, идентификационный номер пользователя. является входным параметром.

Вывод всех привилегий выполняет функция *[get]Privileges*, входных и выходных параметров нет.

Вывод категорий по пользователю выполняет функция *[get]CategoriesByUser*, входные и выходные параметры отсутствуют.

Добавление товара в корзину выполняет функция *[set]ProductInCart*, идентификационный номер и количество товара являются входными данными.

Получение корзины пользователя выполняет функция *[get]UserCart*, входные и выходные параметры отсутствуют.

Получение корзины пользователя выполняет функция *[change]ProductInCart*, идентификационный номер корзины, идентификационный номер товара, количество, являются входными параметрами, выходных данных нет.

1.3.6 Разработка модели

Функции контроллера выполняют функции *view, insert, delete, read, wired* для моделей.

Модели – это классы, реализованные на PHP, для связи вида, бизнес логики и СУБД (рисунок 12, см. выше).

Фактически данные классы (рисунок 13) транслируют данные в таблицы 7 - 12.

В результате реализации приложения на основании паттерна Модель-Вид-Контроллер, выполняется приложение, структура которого показана на рисунке 13.

С помощью стандартного механизма *RESTAPI*. данные и команды между блоками приложения передаются в формате *JSON*.

Всё взаимодействие с сервером сводится к 4 операциям (4 - это необходимый и достаточный минимум, в конкретной выполнение типов операций может быть больше):

- получение данных с сервера (в формате *JSON*);
- добавление новых данных на сервер;
- модификация существующих данных на сервере,
- удаление данных на сервере.

Операция получения данных не может приводить к изменению состояния сервера.

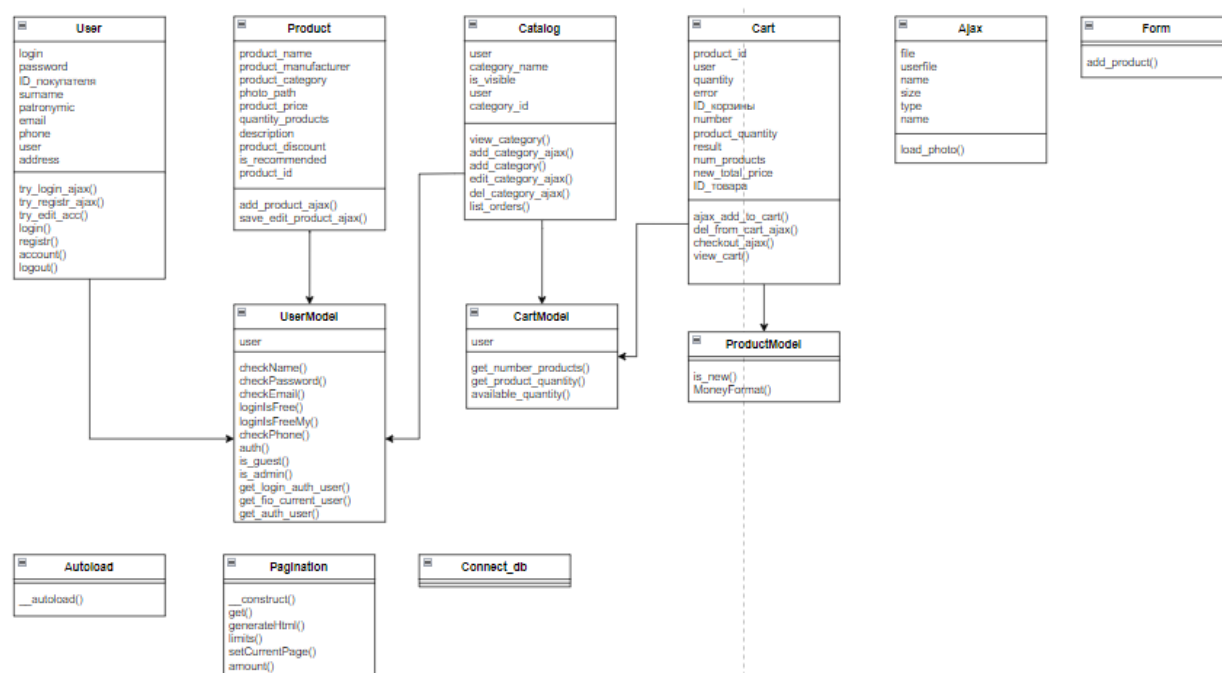


Рисунок13 -UML-диаграмма классов

Взаимодействие интерфейса и сервера выглядят следующим образом:

```
// Объявляем нужные константы

define('DB_HOST','localhost;
define('DB_USER', 'root');
define('DB_PASSWORD', '');
define('DB_NAME', 'store');

define('EMAIL_ADMIN','arsenteva@gmail.com');
define('EMAIL_FROM_NAME', 'Web-порталинтернет-магазина');
define('SITE', 'localhost');

//При нажатии на кнопку "Вход"
$("#btn-login").click(function()
{
    var params =
    {
        login: $("#user_login").val(),
        password: $("#user_password").val()
    };
    $.post("/try_login", params, function(data)
    {
        var errors = (JSON.parse(data));
        $("#error_login").html(errors.login);
        $("#error_password").html(errors.password);
        if ((errors.login + errors.password).isEmpty())
        {
            window.location.href = "/catalog/1/page-1";
        }
    });
});

//При нажатии на кнопку "Регистрация"
$("#btn-registr").click(function()
{
    $("#success").html("");

    var params =
    {
        surname: $("#user_surname").val(),
        name: $("#user_name").val(),
        patronymic: $("#user_patronymic").val(),
        login: $("#user_login").val(),
        password: $("#user_password").val(),
        email: $("#user_email").val()
    };
    $.post("/try_registr", params, function(data)
    {
        var errors = (JSON.parse(data));
        $("#error_surname").html(errors.surname);
        $("#error_name").html(errors.name);
        $("#error_patronymic").html(errors.patronymic);
        $("#error_login").html(errors.login);
    });
});
```

```

        $("#error_password").html(errors.password);
        $("#error_email").html(errors.email);
        if ((errors.surname + errors.name +
errors.patronymic + errors.login + errors.password +
errors.email).isEmpty())
        {
            $("#success").html("Регистрация успешно
завершена");
        }

    });
});

```

1.3.7 Описание работы приложения

После запуска web-портала неавторизованный пользователь может использовать только две вкладки: «Вход на сайт» и «Регистрация». При этом любая попытка перейти на другую страницу введением URL-адреса будет вести к перенаправлению пользователя обратно на страницу авторизации.

```

<section>
<div class="container">
<div class="row pull-center">
<div class="col-sm-4 col-sm-offset-4 signup-col">
<div class="signup-form center">
<h2>Вход на сайт</h2>
<font color="Red"><ruby id="error_login"></ruby></font>
<input type="login" id="user_login" placeholder="Логин"
title="Логин"/>
<font color="Red"><ruby id="error_password"></ruby></font>
<input type="password" id="user_password" placeholder="Пароль"
title="Пароль"/>
<br>
<input type="button" class="btn btn-default btn-login"
value="Вход"/>
</div>
</div>
</div>
</div>
</section>

```

Для кнопки «Вход» был прописан обработчик события нажатия на кнопку на языке *JavaScript*, он считывает введенные пользователем логин и пароль и отправляет их на сервер. На сервере выполняется проверка корректности введенных данных (Логин должен быть не короче 2-х символов, пароль должен быть не короче 4-х символов, выполняется проверка существования введенного логина и соответствия введенного логина и пароля для текущего пользователя из базы данных). В случае возникновения ошибок, они отправляются сервером обратно клиенту и на стороне клиента выводятся на экран. В случае отсутствия ошибок пользователя авторизуют и перенаправляют на страницу с категориями товаров.

Форма «Авторизация» представлена на рисунке 14.

The image shows a login form with a dark gray background and rounded corners. At the top, the title 'ВХОД НА САЙТ' is written in orange. Below the title are two light blue input fields with rounded ends. The first field is labeled 'Логин' and the second is labeled 'Пароль'. At the bottom of the form is a light blue button with rounded ends labeled 'Вход'.

Рисунок 14 - Форма «Авторизация»

Для кнопки «Регистрация» был также прописан обработчик события нажатия на кнопку, который считывает введенные данные и отправляет их на сервер. На сервере проверяется корректность всех введенных данных (Фамилия, имя, отчество, логин не должны быть короче 2-х символов, пароль не должен быть короче 4-х символов, *e-mail* проверяется с использованием регулярного выражения, логин должен быть не занят другими пользователями). В случае корректности всех введенных данных в базу

данных добавляется запись с данными пользователя. Все ошибки отправляются обратно клиенту, где они выводятся на экран без перезагрузки страницы. В случае, если ошибки отсутствуют, выводится сообщение "Регистрация успешно завершена".

Форма «Регистрация» представлена на рисунке 15.

Рисунок 15 - Форма «Регистрация»

```
<section>
<div class="container">
<div class="signup-form center signup-row">
<div class="row pull-center">
<h2>Регистрация на сайте</h2>
<div class="col-sm-4 col-sm-offset-2">
<font color="Red"><ruby id="error_surname"></ruby></font>
<input type="text" name="surname" id="user_surname"
placeholder="Фамилия" title="Фамилия"/>
<font color="Red"><ruby id="error_login"></ruby></font>
<input type="login" name="login" id="user_login"
placeholder="Логин" title="Логин"/>
<font color="Red"><ruby id="error_patronymic"></ruby></font>
<input type="text" name="patronymic" id="user_patronymic"
placeholder="Отчество" title="Отчество"/>
</div>
<div class="col-sm-4">
<font color="Red"><ruby id="error_name"></ruby></font>
```

```

<input type="text" name="name" id="user_name" placeholder="Имя"
title="Имя"/>

<font color="Red"><ruby id="error_password"></ruby></font>
<input type="password" name="password" id="user_password"
placeholder="Пароль" title="Пароль"/>

<font color="Red"><ruby id="error_email"></ruby></font>

<input type="email" name="email" id="user_email" placeholder="E-
mail" title="E-mail"/>

</div>

<div class="col-sm-6 pull-center col-sm-offset-3 center">
<br>

<font color="Cyan"><ruby id="success"></ruby></font>

<input type="button" class="btn btn-default btn-registr"
value="Регистрация"/>

</div>

</div>

</div>

</div>

</section>

```

Для авторизованного пользователя существуют три вкладки: «Категории», «Аккаунт», «Корзина» и возможность выхода из аккаунта (рисунок 16).

При авторизации пользователь автоматически перенаправляется на первую страницу каталога. При этом генерируется список видимых пользователям категорий, список товаров, отображаемых на данной странице.

```

<div class="col-sm-3">
<div class="left-sidebar">
<h2>Категории товаров</h2>
<div class="panel-group category-products">
<? if(UserModel::is_admin()): ?>
<div class="panel panel-default">
<div class="panel-heading">

```

```

<h4 class="panel-title">
<i class="fa fa-plus-circle"></i><a href="/catalog/add"
class="<?=( $action == 'add_category')? 'active': '' ; ?>">
Добавить категорию</a>
<br><br>
</h4>
</div>
</div>

```

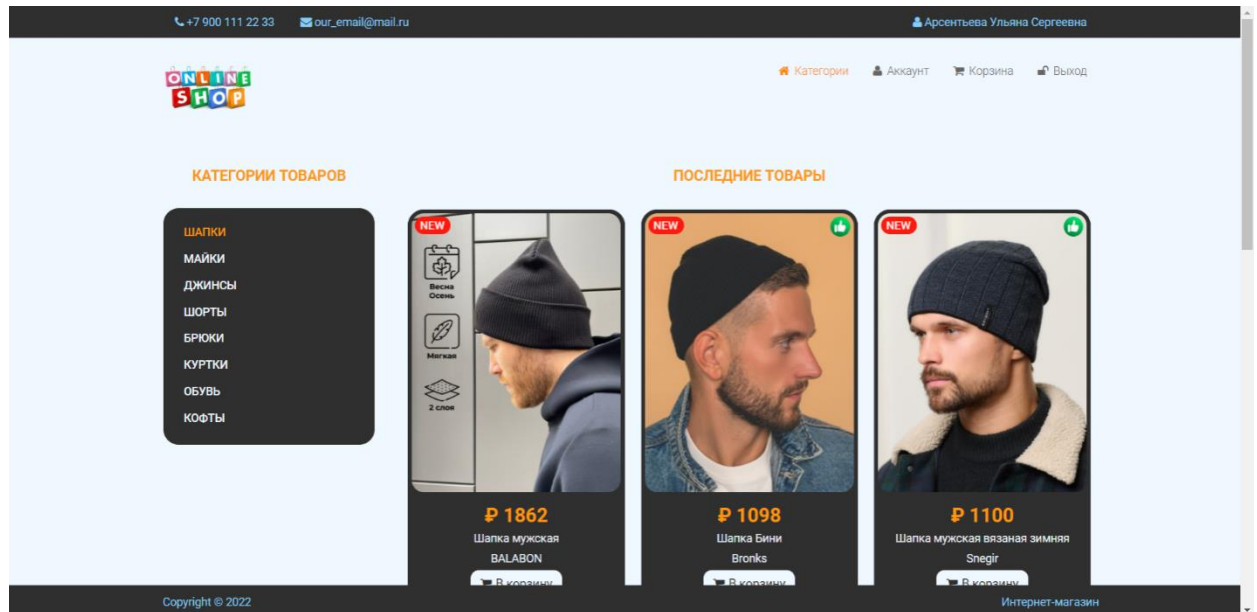


Рисунок 16 - Интерфейс авторизованного пользователя

В левой колонке «Категории товаров» можно переходить к другим категориям по ссылкам. В нижней части существует возможность перехода между страницами, рисунок 17.

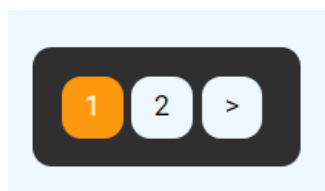


Рисунок 17 - Переход между страницами

Представление отображает список товаров. При этом на фото в левом верхнем углу отображается значок «Новинка», если товар был добавлен в течение двух недель, а в правом углу отображается значок «Рекомендован»,

если это было указано при добавлении товара на сайт.

Для кнопки «В корзину» был добавлен обработчик события нажатия на кнопку, который считывает идентификатор добавляемого в корзину товара и отправляет данные на сервер. На сервере проверяется возможность добавления товара в корзину (товаров в корзине не может быть больше, чем товара имеется на складе) и товар добавляется в корзину, если до этого он отсутствовал в ней, или увеличивает количество товара на единицу, если данный товар уже был добавлен ранее. При этом на стороне клиента обновляется значение количества товаров в корзине и выводятся на экран сообщение об ошибке при попытке добавления товара в количестве большем, чем имеется на складе.

Сообщение об ошибке представлено на рисунке 18.

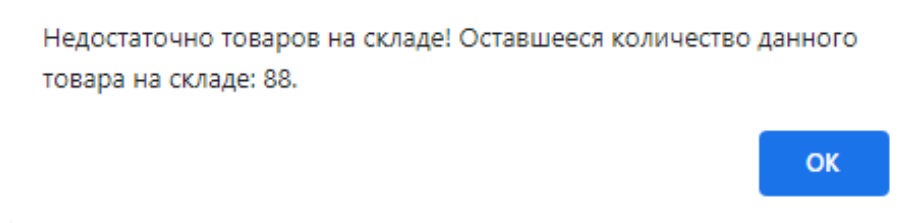


Рисунок 18 - Сообщение об ошибке

При нажатии на фотографию товара выполняется переход по ссылке к просмотру товара. В данном случае вызывается метод *view_product*, который выполняет обращение к базе данных и получает список категорий товаров и выбранного товара. Также проверяется количество выбранного товара в корзине с количеством того же товара на складе. В случае, если в корзине имеется больше товара, чем на складе, из корзины удаляется определенное число товаров. Данный случай возможен, если несколько единиц выбранного товара были куплены другим пользователем.

Просмотр товара представлен на рисунке 19.

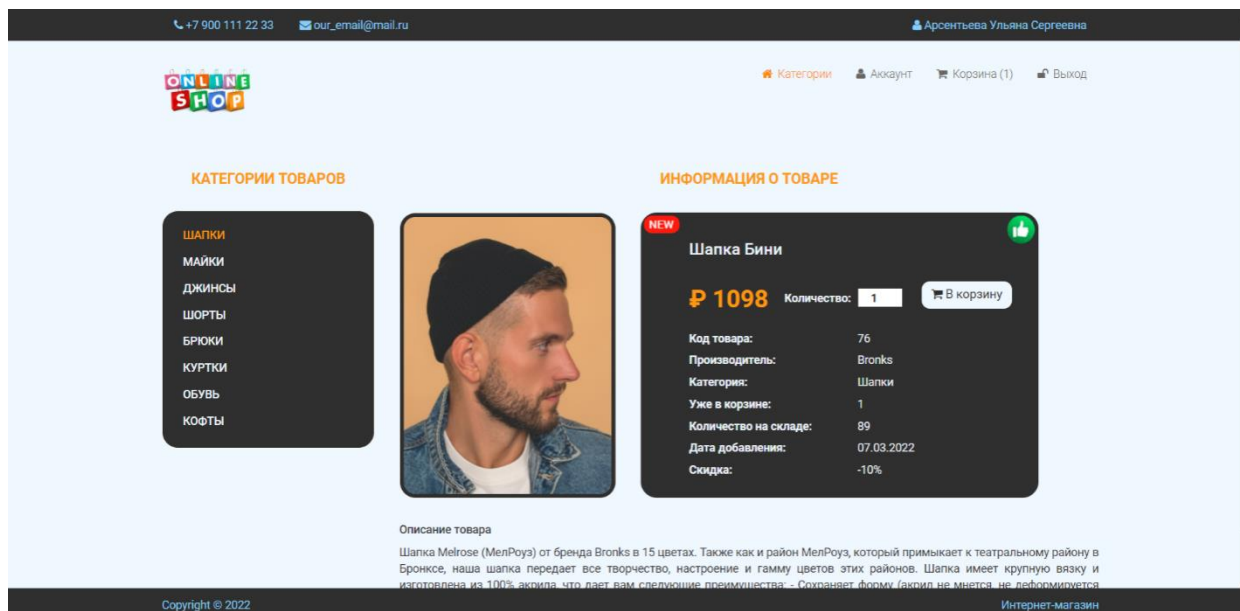


Рисунок 19 - Описание товара

В данном случае для кнопки «В корзину» также прописан обработчик события нажатия на кнопку, позволяющий добавлять в корзину введенное количество товаров. Обработчик считывает введенное количество и идентификатор товара и передает их на сервер. В данном случае требуется также добавить, что в методе также выполняется проверка и на введенное пользователем отрицательное значение количества. В случае корректности данных значение количества в базе данных обновляется.

При переходе на вкладку «Мой аккаунт» управление передается методу *account*, который получает информацию об авторизованном пользователе из базы данных и вызывает представление *views/account*.

```
<input type="text" name="surname"
      id="user_surname" placeholder="Фамилия" value="<?=$user['Фамилия']; ?>" title="Фамилия"/>
<font color="Red"><ruby id="error_email"></ruby></font>
<input type="email" name="email" id="user_email" placeholder="E-mail" value="<?=$user['Почта']; ?>" title="E-mail"/>
<input type="text" name="address" id="user_address" placeholder="Адрес" value="<?=$user['Адрес']; ?>" title="Адрес"/>
<font color="Red"><ruby id="error_name"></ruby></font>
```

```

<input type="text" name="name" id="user_name" placeholder="Имя"
value="<?= $user['Имя']; ?>" title="Имя"/>

<font color="Red"><ruby id="error_login"></ruby></font>

<input type="text" name="login" id="user_login"
placeholder="Логин" value="<?= $user['Логин']; ?>" title="Логин"/>

<font color="Red"><ruby id="error_phone"></ruby></font>

<input type="number" name="phone" id="user_phone"
placeholder="Телефон" value="<?= $user['Телефон']; ?>"
title="Телефон"/>

</div>

<div class="col-sm-4">

<font color="Red"><ruby id="error_patronymic"></ruby></font>

<input type="text" name="patronymic" id="user_patronymic"
placeholder="Отчество" value="<?= $user['Отчество']; ?>"
title="Отчество"/>
<font color="Red"><ruby

        id="error_password"></ruby></font>

<input type="text" name="password" id="user_password"
placeholder="Пароль" value="<?= $user['Пароль']; ?>"
title="Пароль"/>

<input type="text" name="text" id="user_date"
title="Дата регистрации" value="<?=
date_create($user['Дата_регистрации'])->format('d.m.Y'); ?>"
readOnly/>

```

Для кнопки «Сохранить изменения», представленной на рисунке 20, был написан обработчик события нажатия на кнопку, который считывает все введенные данные и отправляет их на сервер. На сервере производится проверка на корректность (проверка практически аналогична процессу регистрации) и, в случае их корректности, обновляет запись о пользователе в базе данных. Список ошибок отправляется клиенту, который выводит их все на экран или в случае, если ошибки отсутствовали, выводит сообщение "Изменения успешно сохранены".

При переходе на вкладку «Корзина» вызывается метод *view_cart* контроллера *cart*. В данном методе происходит обращение к базе данных, получение списка всех добавленных в корзину товаров и вызов отображения

views/cart с отображением в нем информации обо всех товарах, добавленных в корзину.

The screenshot shows a web interface for an online shop. At the top, there's a header with contact information (+7 900 111 22 33, our_email@mail.ru) and the user's name (Арсентьева Ульяна Сергеевна). Below the header is a navigation bar with links for 'Категории', 'Аккаунт', 'Корзина', and 'Выход'. The main content area is titled 'МОЙ АККАУНТ' and contains a form with the following fields:

Арсентьева	Ульяна	Сергеевна
arsenteva.ulya@yandex.ru	89656308948	89656308948
Адрес	Телефон	11.03.2022

At the bottom of the form is a button labeled 'Сохранить изменения'.

Рисунок 20 - Кнопка «Сохранить изменения»

```
<section id="cart_items">
  <div class="container">
    <div class="table-responsive cart_info">
      <h2 class="cart_form">Корзина</h2>

      <table class="table table-condensed">
        <thead>
          <tr class="cart_menu">
            <td class="image">Фото</td>
            <td class="description">Описание товара</td>
            <td class="price">Цена</td>
            <td class="discount">Скидка</td>
            <td class="price_discount">С скидкой</td>
            <td class="quantity">Количество</td>
            <td class="total">Итого</td>
            <td class="delete"></td>

```

В случае, представленном на рисунке 21, имеется три различных обработчиков событий нажатий на кнопки.

КОРЗИНА						
Фото	Описание товара	Цена	Скидка	Со скидкой	Количество	Итого
	Шапка Бини Код товара: 76	₽ 1 098.00	10 %	₽ 988.20	1 <input type="text"/> <input type="button" value="✓"/> На складе: 89	₽ 988.20 <input type="button" value="x"/>
	Шапка мужская зимняя вязаная Код товара: 79	₽ 2 599.00	20 %	₽ 2 079.20	1 <input type="text"/> <input type="button" value="✓"/> На складе: 12	₽ 2 079.20 <input type="button" value="x"/>
	Шапка Бини, зимняя Код товара: 78	₽ 2 500.00	10 %	₽ 2 250.00	1 <input type="text"/> <input type="button" value="✓"/> На складе: 63	₽ 2 250.00 <input type="button" value="x"/>
Всего:						₽ 5 317.40
<input type="button" value="Оформить заказ"/>						

Рисунок 21 - Корзина покупателя

При нажатии на «крест» (удаление товара из корзины) на сервер передается идентификатор товара в корзине. На сервере вызывается метод, в котором происходит удаление товара в корзине из базы данных. Также на сервере рассчитывается новое количество товаров в корзине и суммарная стоимость всех товаров в корзине. Эти данные отправляются клиенту для их обновления на странице.

При нажатии на «галочку» (Подтверждение введенного количества единиц выбранного товара) на сервер передается количество товара и идентификатор товара. На сервере вызывается метод, который выполняет проверку на введенное количество товаров в корзине (значение не должно быть отрицательным и введенное количество не должно превышать количество единиц товара на складе) и обновляет это количество в базе данных. Также на сервере рассчитывается новое количество данного товара, общее число товаров в корзине, стоимость товаров с учетом нового количества и суммарная стоимость всех товаров. Все эти значения передаются клиенту для обновления данных на странице.

При нажатии на «Оформить заказ» вызывается обработчик события

нажатия на кнопку, который не считывает данные со страницы, а сразу переходит на сервер, где происходит обращение к базе данных, получение списка товаров в корзине с их количеством, из которого формируется заказ, который сохраняется в виде множества пар значений в формате json, где первое значение пары – идентификатор товара, а второе значение – его количество. Далее запись о заказе добавляется в базу данных, а все товары корзины удаляются из базы данных. При этом для каждого купленного товара уменьшается его количество на складе (таким образом имитируется процесс покупки). Также на сервере обрабатывается случай оформления заказа при отсутствии товаров в корзине – в этом случае запись с заказом не добавляется в базу данных, а клиенту передается сообщение об ошибке. После получения клиентом ответа, на странице происходит очищение товаров из корзины и выводится сообщение "Заказ успешно оформлен!". В случае ошибки выводится соответствующее сообщение.

В случае выхода вызывается метод, который удаляет авторизованного пользователя из сессии и перенаправляет пользователя на страницу авторизации (Вход на сайт).

Имеется возможность зайти на сайт от имени администратора. Для этого необходимо ввести логин и пароль: «Admin». В этом случае имеется более широкий функционал – имеется возможность просматривать список заказов пользователей, добавлять/редактировать/ удалять категории или отдельные товары.

На верхней правой панели имеется вкладка «Добавить товар». При переходе по вкладке вызывается метод, который обращается к базе данных, формирует список категорий товара и вызывает форму-представление для добавления товара.

```
<section>
<div class="container">
<div class="signup-form">
```

```

<div class="row signup-row">
  <? if (UserModel::is_admin()): ?>
  <h2>Добавление товара</h2>
<div class="col-sm-3 pull-center not-hover">
  
  <input type="button" id="btnUpload" class="btn btn-default"
  value="Загрузить фото" />
  <table><tr>
  <td style="text-align:right"><input type="checkbox"
  class="checkbox" id="is_recommended"
  checked><label>Рекомендовать</label></td>
  </tr></table>
</div>

```

При нажатии на кнопку «Загрузить фото» выполняется обработчик события нажатия на кнопку, который открывает окно для выбора изображения. После выбора изображения вызывается метод, который выполняет загрузку изображения на сервер.

При нажатии на кнопку «Добавить товар» на стороне клиента выполняется проверка всех введенных данных на корректность (В данном случае значения просто должны быть заполнены, а не пустыми) и отправляются на сервер. На сервере вызывается метод *add_product_ajax* контроллера *product*, который принимает полученные данные, обращается к базе данных и добавляет новую запись товара.

В режиме администратора при просмотре каталога у товара были добавлены были добавлены две новые кнопки «Редактировать товар» и «Удалить товар». Кнопка редактирования представляет собой ссылку, которая перенаправляет пользователя на страницу редактирования товара, а для кнопки удаления был прописан соответствующий обработчик события нажатия на кнопку.

Добавление товара представлено на рисунке 22.

—

Рисунок 22 - Добавление товара

При нажатии на кнопку «Сохранить изменения» срабатывает обработчик события нажатия на кнопку, который считывает и проверяет введенные данные на корректность прямо на стороне клиента (Данный процесс проверки корректности аналогичен процессу проверки корректности при добавлении товара). После этого введенные данные отправляются на сервер, где их принимает метод, в котором происходит обновление записи о товаре в базе данных.

При нажатии на кнопку «Удалить товар» вызывается обработчик события нажатия на кнопку, который считывает идентификатор товара и передает его на сервер, где его принимает метод `del_product_ajax` контроллера `product`. В этом методе происходит удаление товара из базы данных. После получения ответа от сервера на стороне клиента происходит удаление товара со страницы.

В режиме администратора в список категорий товаров добавлены возможности добавления/удаления/редактирования категорий, а также возможность создавать категории, которые не будут видны обычным пользователям.

Редактирование товара представлено на рисунке 23.


```

<div class="categories">
<? include(PATH . 'views/categories.php'); ?>
</div>
<div class="col-sm-5 col-sm-offset-2">
<div class="features_items">
<? if(UserModel::is_admin()): ?>
<h2 class="title text-center">Добавление категории</h2>
<div class="signup-form">
<div class="form-category">
<font color="Red"><ruby id="errors"></ruby></font>
<table><tr><td><font color="Cyan"><ruby
id="success"></ruby></font></td></tr></table>
    <input                                type="text"                                id="name_category"
placeholder="Название категории" title="Название категории"/>

<input type="checkbox" class="checkbox" id="is_visible"
checked><label>Видна покупателям</label>

<br><br>
<input type="button" class="btn btn-default btn-add-category"
title="Добавить категорию" value="Добавить категорию"/>
</div>
</div>
<? else: ?>
<h2 class="title text-center">Нет доступа</h2>
<? endif; ?>
</div>
</div>
</div>
</div>
</section>

```

При нажатии на кнопку «Добавить категорию», представленную на рисунке 25, срабатывает обработчик события нажатия на кнопку, который считывает введенное пользователем значение названия категории и

переключателя, значение которого определяет, будет ли видна созданная категория обычным пользователям. На стороне клиента данные проверяются на корректность (Поле «Название категории» не должно быть пустым), после чего данные передаются на сервер методом *add_category_ajax* контроллера *cart*. В методе происходит добавление новой категории в базу данных и возвращение нового списка категорий, который затем отображается на странице на стороне клиента.

Аналогичным образом, переходя по ссылке «Редактировать категорию», происходит переход на соответствующую страницу «Редактирование категории». В данном методе происходит получение списка категорий и информация о редактируемой категории из базы данных. Полученная информация подставляется на форму в представлении *views/form_edit_category*. Дальнейшая доработка и расширение функционала позволит ее сделать более универсальной.

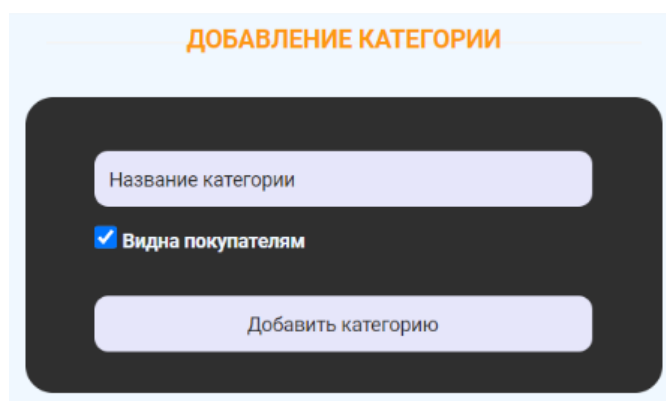


Рисунок 25 - Добавление категории

При нажатии на «Сохранить изменения» на стороне клиента происходит совершенно аналогичная проверка корректности, как и при добавлении категории. Данные передаются на сервер, где их принимает метод *edit_category_ajax* контроллера *catalog*. Данные обновляются в базе данных, а клиенту передается новый список категорий, который обновляется на странице.

При нажатии на кнопку «Удалить категорию» обработчик события отправляет на сервер идентификатор категории. На сервере метод *del_category_ajax* контроллера *catalog* принимает идентификатор категории и выполняет проверку на то, что в данной категории отсутствуют товары. Удаление категории и базы данных производится только в случае, если в категории товары отсутствуют, а в противном случае сервер возвращает сообщение о невозможности удаления. В случае успешного удаления категории на стороне клиента происходит удаление категории без перезагрузки страницы.

Покупатель (обычный пользователь) может только оформить заказ, а доставкой заказов до нужного места должны заниматься курьеры. Поэтому администратор должен иметь доступ ко всему списку заказов покупателей для организации работы курьеров. В режиме администратора можно перейти на вкладку «Список заказов». В этом случае вызывается метод *list_orders* контроллера *catalog*, в котором из базы данных формируется последний список заказов и вызывается представление для их отображения: *views/list_orders*.

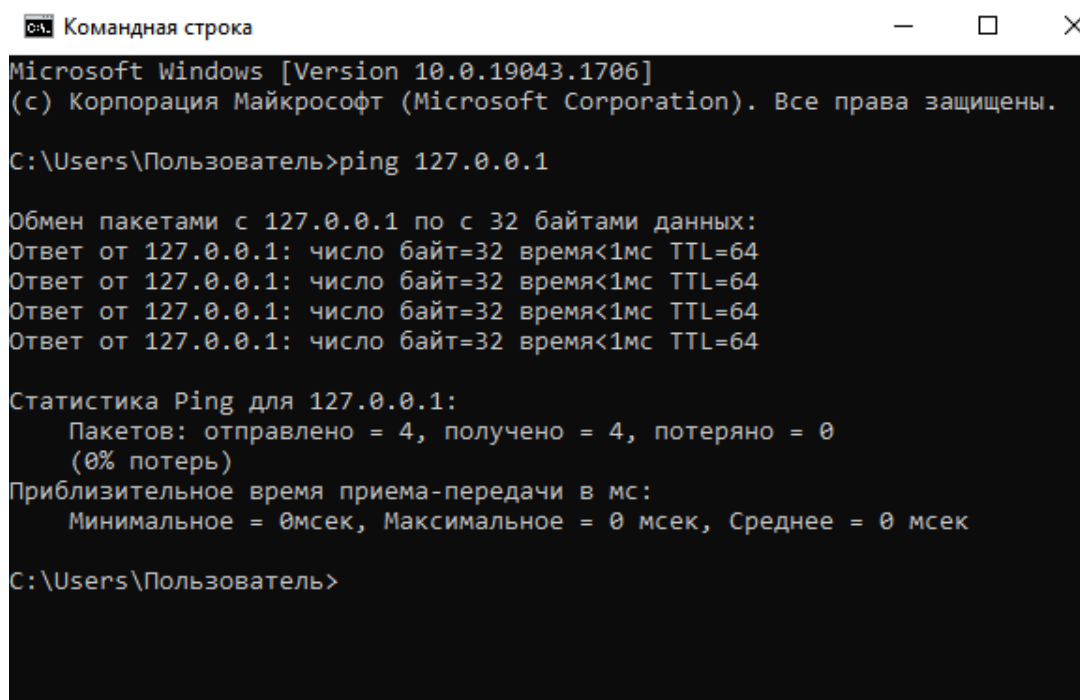
СПИСОК ЗАКАЗОВ									
ФИО	Адрес	Телефон	Почта	Заказанные товары				Дата и время	Итого
Арсентьева Ульяна Сергеевна			arsenteva.ulya@ yandex.ru	58	Шорты мужские	IMHO	x1	Р 2 170.00	16.03.22 00:47 Р 20 164.90
				63	Шорты мужские	BEZET	x1	Р 4 644.90	
				65	Шорты мужские	IMHO	x1	Р 2 480.00	
				109	Кофта мужская на молнии	PULLTONIC	x1	Р 10 870.00	
Арсентьева Ульяна Сергеевна			arsenteva.ulya@ yandex.ru	74	Шапка мужская	BALABON	x1	Р 1 675.80	16.03.22 00:47 Р 5 114.20
				79	Шапка мужская зимняя вязаная	ALDINI	x1	Р 2 079.20	
				82	Шапка TIRO WOOLIE BLACK/ WHITE	Adidas	x1	Р 1 359.20	

Рисунок 26 - Список заказов

1.4 Технология разработки

Для разработки программного обеспечения требуется установка и настройка среды разработки, СУБД и ОС. Прежде чем устанавливать программы, необходимо произвести проверку сетевых настроек и убедиться в том, что порты 80 и 3306 свободны, так как эти порты занимают *Web*-сервер *Apache* и сервер *MySQL*. Для этого следует запустить командную строку *Windows* и ввести команду *cmd*. Далее ввести в командную строку следующую команду:

ping 127.0.0.1 (рисунок 27).



```
Microsoft Windows [Version 10.0.19043.1706]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Пользователь>ping 127.0.0.1

Обмен пакетами с 127.0.0.1 по 32 байтами данных:
Ответ от 127.0.0.1: число байт=32 время<1мс TTL=64
Ответ от 127.0.0.1: число байт=32 время<1мс TTL=64
Ответ от 127.0.0.1: число байт=32 время<1мс TTL=64
Ответ от 127.0.0.1: число байт=32 время<1мс TTL=64

Статистика Ping для 127.0.0.1:
    Пакетов: отправлено = 4, получено = 4, потеряно = 0
    (0% потерь)
Приблизительное время приема-передачи в мс:
    Минимальное = 0мсек, Максимальное = 0 мсек, Среднее = 0 мсек

C:\Users\Пользователь>
```

Рисунок 27 - Командная строка

Если число потерянных пакетов больше 0, то необходимо произвести проверку сетевых настроек. В нашем же случае число потерянных пакетов равно 0.

Затем проводится проверка занятости портов 80 и 3306, в нашем случае эти порты свободны.

1.4.1 Установка XAMPP

Все необходимые программы входят в пакет XAMPP, который необходимо установить. Для этого необходимо перейти на сайт <https://www.apachefriends.org/> и загрузить последнюю версию. Затем запустить программу установки, рисунок 28, и нажать кнопку *Next*.



Рисунок 28 - Установка XAMPP

На следующем шаге производится проверка установки всех флажков, при их наличии, следует нажать кнопку *Next*, рисунок 29.

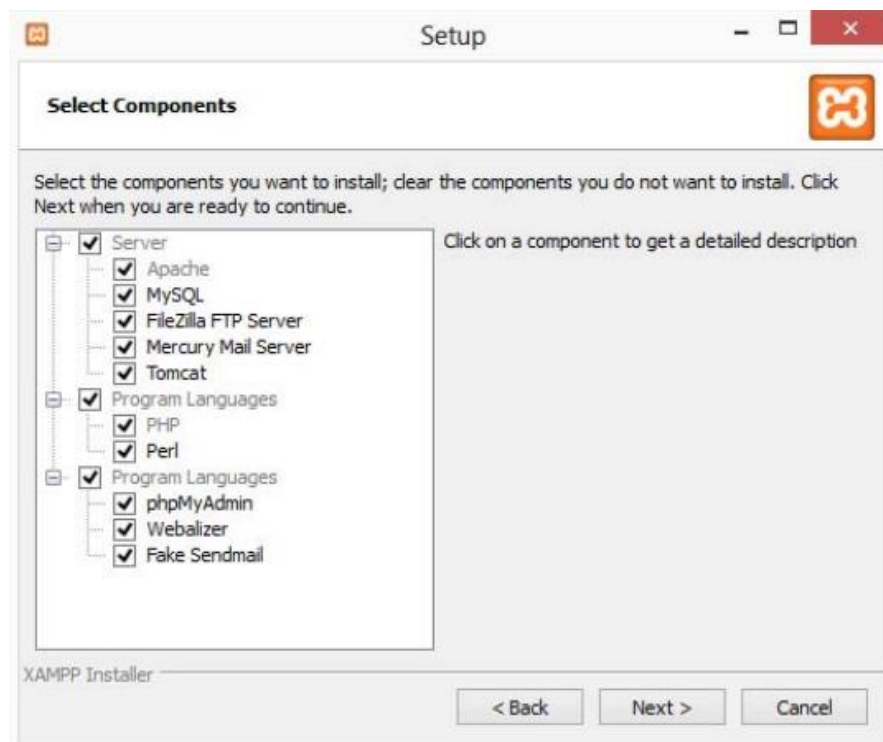


Рисунок 29 - Установка XAMPP

Требуется установить пакет в папку *C:\xampp*. На следующих двух шагах (рисунок 30 и 31) так же следует нажать кнопку *Next*.

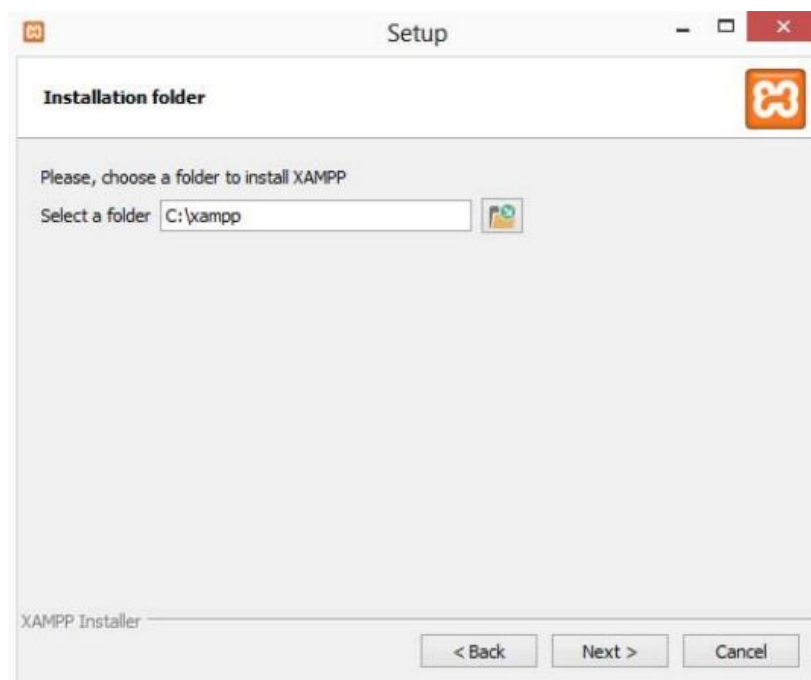


Рисунок 30 - Установка XAMPP



Рисунок 31 - Установка XAMPP

В процессе установки будет выведено окно с запросом блокировки запуска сервера *Apache*. Следует в обязательном порядке разрешить запуск, иначе он работать не будет. На последнем шаге, показанном на рисунке 32, следует нажать кнопку *Finish*.



Рисунок 32 - Установка XAMPP

Затем автоматически запускается приложение *XAMPPControlPanel*,
рисунок 33.

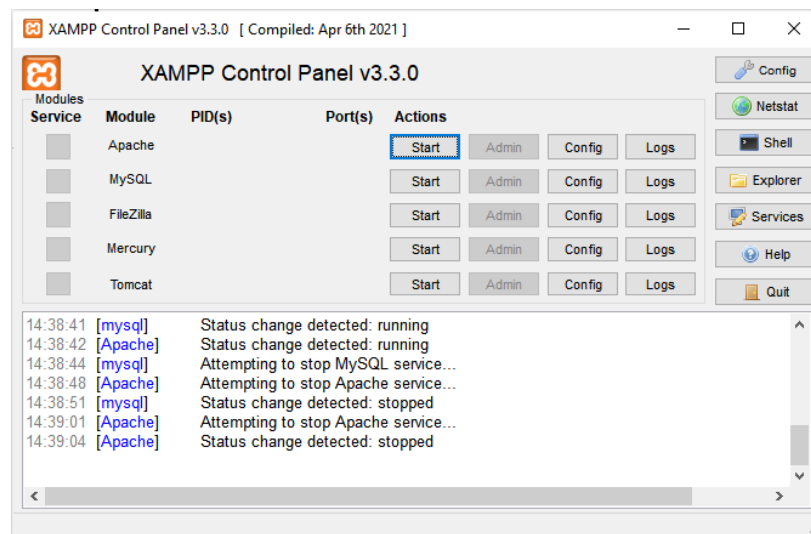


Рисунок 33 - Окно *XAMPPControlPanel*

Для запуска серверов следует нажать кнопки *Start* у пунктов *Apache* и *MySQL*, рисунок 34. При успешном запуске названия серверов будут выделены зеленым цветом, а справа отобразятся идентификаторы процессов и номера портов, на которых серверы запущены. При этом кнопки *Start* будут заменены на кнопки *Stop*, с помощью которых можно будет серверы остановить.

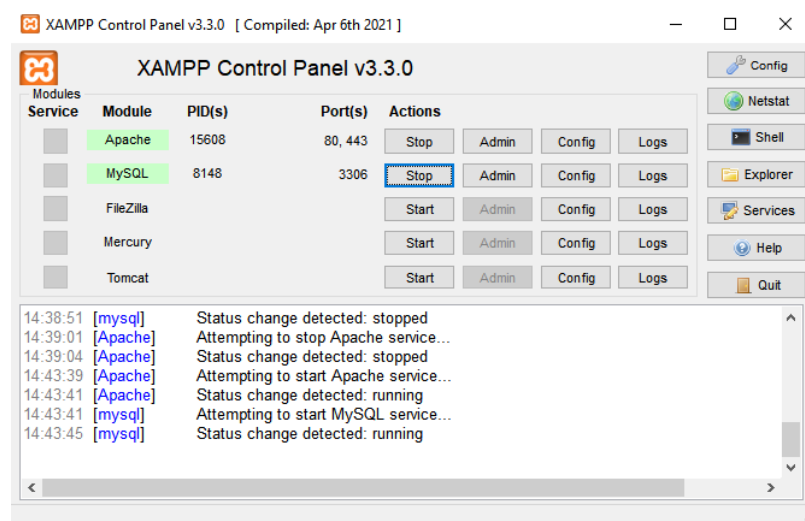


Рисунок 34 - Окно *XAMPPControlPanel*

После нажатия красной кнопки с крестиком в заголовке окна *XAMPP Control Panel* окно не закрывается, а просто скрывается. Значок приложения доступен на панели задач *Windows*. Если щелкнуть на этом значке правой кнопкой мыши, то отобразится контекстное меню (рисунок 35), с помощью которого можно открыть окно *XAMPP Control Panel* или управлять работой серверов.

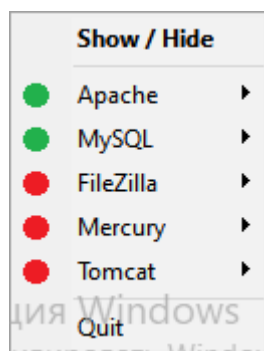


Рисунок 35 -Контекстное меню приложения *XAMPP Control Panel*

Для проверки работоспособности Web-сервера *Apache* необходимо открыть Web-браузер и в адресной строке ввести: *http://localhost/*. В результате должно отобразиться приветствие пакета *XAMPP*, рисунок 36. Чтобы проверить работоспособность сервера *MySQL* и программы *phpMyAdmin*, в адресной строке Web-браузера необходимо ввести: *http://localhost/phpmyadmin/*. Результат успешной проверки показан на рисунке 37.

Успешный запуск программы *phpMyAdmin* уже свидетельствует о работоспособности интерпретатора *PHP*.



Welcome to XAMPP for Windows 8.1.6

You have successfully installed XAMPP on this system! Now you can start using Apache, MariaDB, PHP and other components. You can find more info in the [FAQs](#) section or check the [HOW-TO Guides](#) for getting started with PHP applications.

XAMPP is meant only for development purposes. It has certain configuration settings that make it easy to develop locally but that are insecure if you want to have your installation accessible to others. If you want have your XAMPP accessible from the internet, make sure you understand the implications and you checked the [FAQs](#) to learn how to protect your site. Alternatively you can use [WAMP](#), [MAMP](#) or [LAMP](#) which are similar packages which are more suitable for production.

Start the XAMPP Control Panel to check the server status.

Рисунок 36 - Приветствие XAMPP

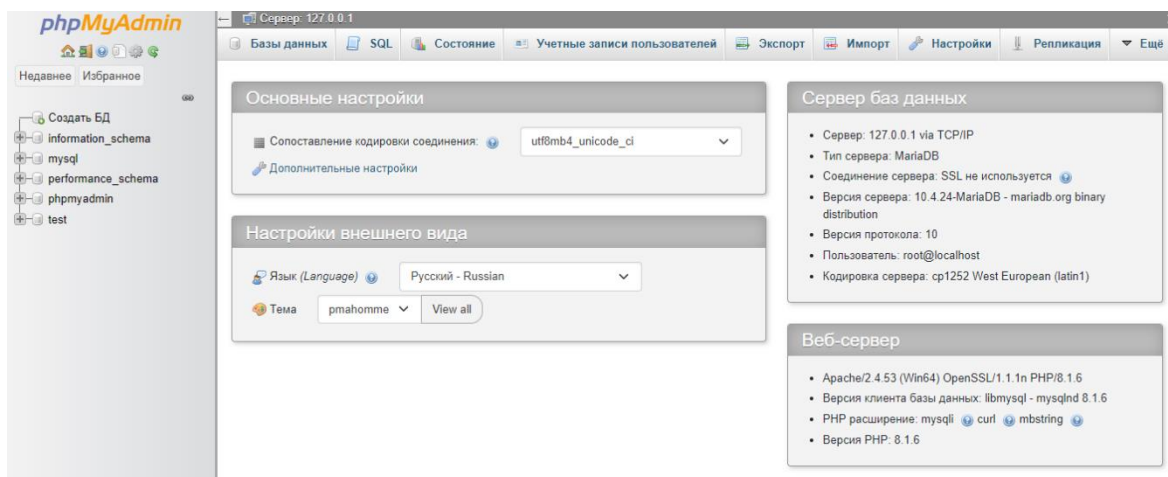


Рисунок 37 - Программа phpMyAdmin

1.4.2 Подключение MySQL к Apache

Для подключения базы данных необходимо перейти в папку MySQL и создать в ней текстовый файл *my.ini*, со следующим содержанием представленным на рисунке 38.

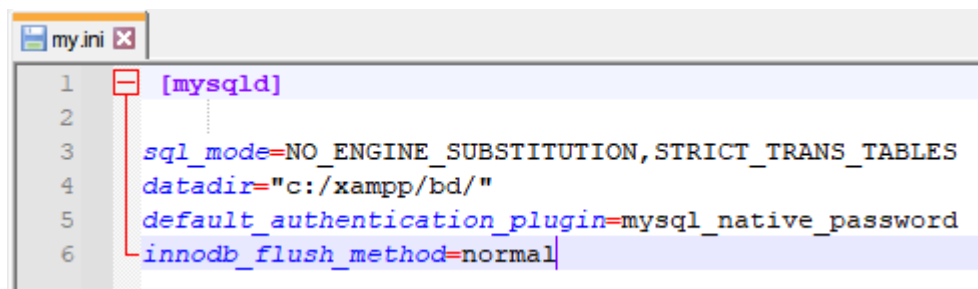


Рисунок 38 - Файл *my.ini*

В папке *bd* будут храниться базы данных.

Настройка завершена, теперь необходимо выполнить инициализацию и установку, рисунок 39. Для этого необходимо открыть командную строку от имени администратора и ввести следующие строки:

```

C:\Server\bin\mysql-8.0\bin\mysqld --initialize-insecure --
user=root
C:\Server\bin\mysql-8.0\bin\mysqld --install
net start mysql
  
```

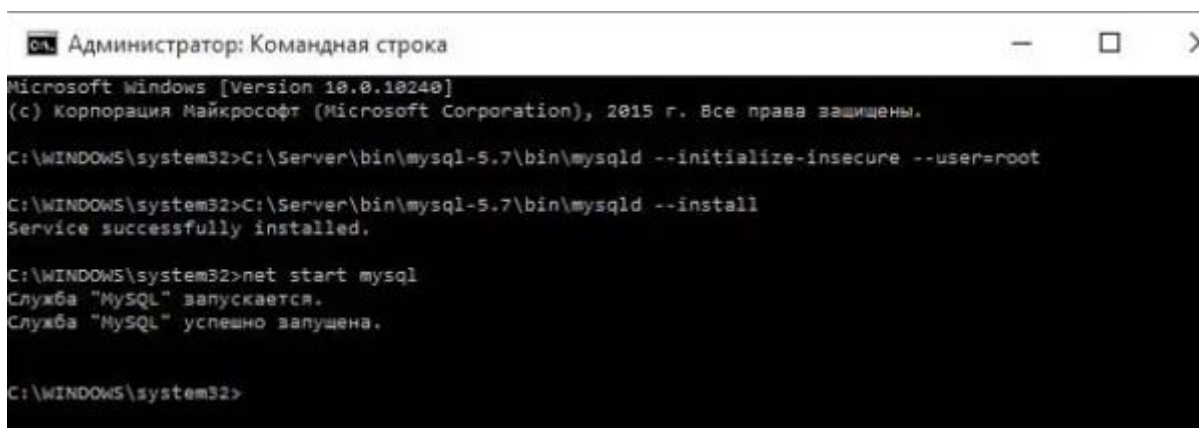


Рисунок 39 - Командная строка

1.5 Выводы

В данном разделе была определена предметная область, определены задачи и функции, реализуемые интернет-магазином, определен стек технологий. В качестве языка программирования выбран язык *PHP*, web-сервером определен *Apache*. Приложение будет реализовываться в формате *MVC* приложения. Был разработан web-портал интернет-магазина. Данный

портал представляет собой две независимые друг от друга части. За работу администратора отвечает первая часть, а для работы клиента в режимах: гость и/или зарегистрированный пользователь, отвечает вторая часть. Был описан процесс установки и настройки инструментария для разработки web-приложения.

					ПГУ1.090301.03.001ПЗ	Лист
Изм.	Лист	№докум.	Подпись	Дата		67

2 Безопасность жизнедеятельности

Нет сомнения в том, что жизнь современного человека тесно связана с информационными технологиями. Несмотря на большое количество достоинств использования компьютера, существуют также определенные недостатки. Наиболее значительным из них является его влияние на здоровье человека. Многочасовая работа на компьютере может стать причиной заболеваний, связанных с воздействием электромагнитного излучения, с повышенной нагрузкой на органы зрения, органы дыхания, а также с сидячим образом жизни, который негативно влияет на мышцы и суставы. Выделяют следующие факторы риска:

- из-за повышения температуры некоторых компонентов техники происходит повышение температура в помещении;
- происходит снижение уровня влажности воздуха в зоне выполнения работ;
- из-за недостаточной освещенности рабочей зоны пользователь получает высокие нагрузки на зрение;
- существует риск поражения электрическим током и статическим электричеством;
- происходит переутомление из-за монотонности работы.

Для снижения рисков предъявляются следующие требования к организации рабочего места:

- площадь рабочего места пользователя, использующего персональный плоский монитор, должна быть не меньше 4,5 квадратных метров;
- рабочее место должно быть оборудовано специальной мебелью, которая должна подлежать регулировке;

– для внутренней отделки помещений необходимо применять безопасные материалы;

– для достижения необходимого уровня освещения в помещении необходимо иметь источники естественного света, установленные на север и северо-запад, во избежание солнечных бликов на экранах компьютеров, затрудняющих получение информации. Регулировка солнечного света должна быть обязательной;

– необходима ежедневная влажная уборка;

– должен проводиться контроль уровня шума и вибрации, интенсивности электромагнитных полей.

					ПГУ1.090301.03.001ПЗ	Лист
Изм.	Лист	№докум.	Подпись	Дата		69

Заключение

В ходе выполнения выпускной квалификационной работы была разработана структура базы данных для последующего создания интернет-магазина. Эта структура выполнена в формате реляционной СУБД.

В разработанном *web*-портале интернет-магазина была устранена часть недостатков аналогичных систем и выполнены все заявленные в техническом задании функциональные требования.

Выполненные отладка и тестирование доказали работоспособность и корректность работы приложения.

Был проведен анализ предметной области, который показал, что у уже существующих *web*-порталов имеются некоторые недостатки. В ходе выполнения отладки и тестирования было доказано, что приложение работоспособно.

Дальнейшая доработка и расширение функционала позволит ее сделать более универсальной.

					ПГУ1.090301.03.001ПЗ	Лист
Изм.	Лист	№докум.	Подпись	Дата		70

Список используемых источников

1. Герберт Шилдт. Полный справочник по C#: Пер. с англ. — М. : Издательский дом "Вильямс", 2004 — 752 с.
2. Простое приложение ASP.NET Web Forms 4.5 [Электронный ресурс]
URL: https://professorweb.ru/my/ASP_NET/webforms_4_5/level1/1_0.php
3. Руководство по веб-формам ASP.NET [Электронный ресурс]. – URL: <https://docs.microsoft.com/ru-ru/aspnet/web-forms/overview/>
4. Знакомство с отладчиком Visual Studio [Электронный ресурс]. – URL: <https://docs.microsoft.com/ru-ru/visualstudio/debugger/debugger-feature-tour?view=vs-2019>
5. Методы и средства отладки, помогающие писать более качественный код [Электронный ресурс]. – URL: <https://docs.microsoft.com/ru-ru/visualstudio/debugger/write-better-code-with-visual-studio?view=vs-2019>
6. Техника безопасности при работе с компьютером [Электронный ресурс]. - URL: <https://helmt.ru/bez-rubriki/tehnika-bezopasnosti-pri-rabote-s-kompyuterom/>

ПриложениеА

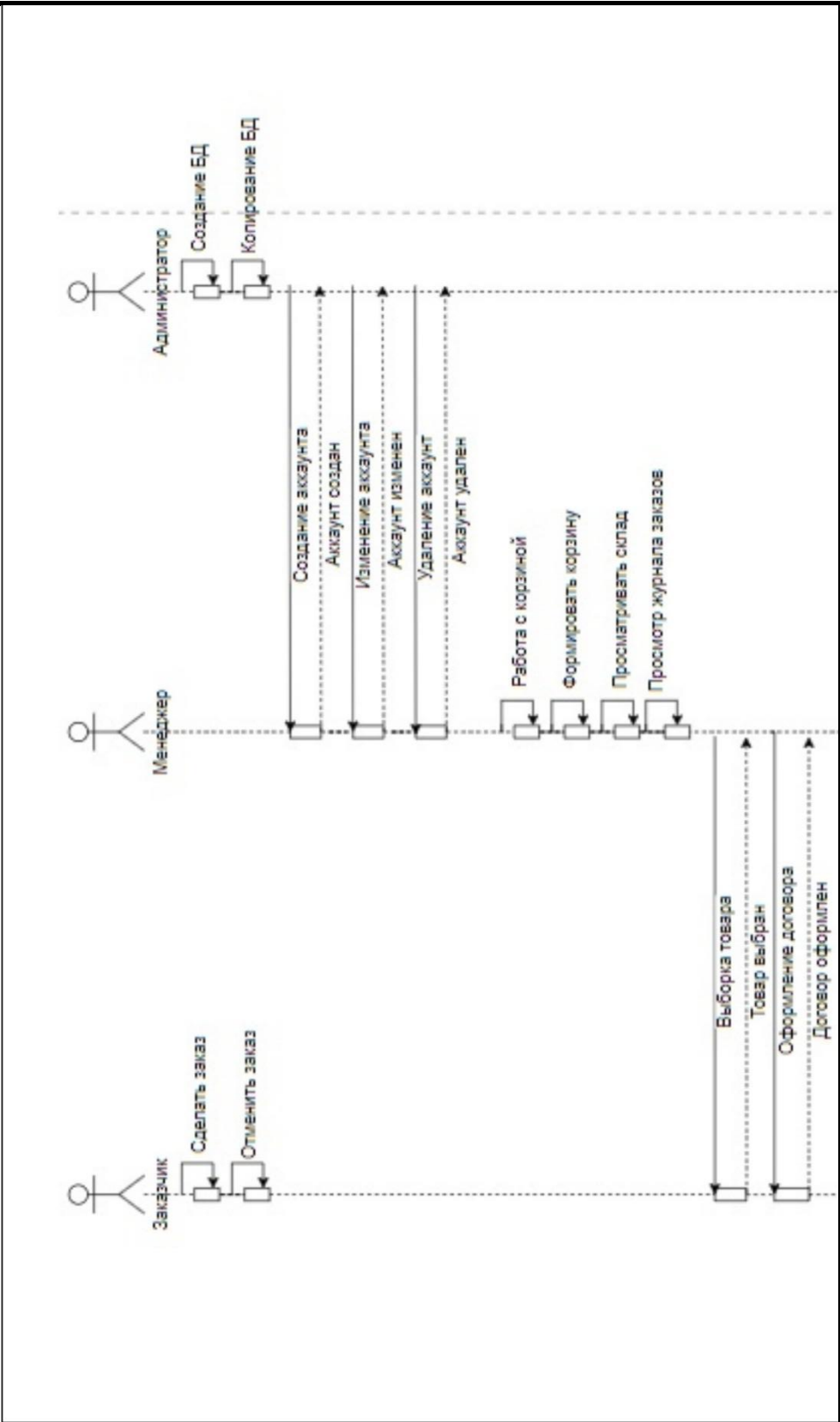
(Обязательное)

Web-портал интернет-магазина

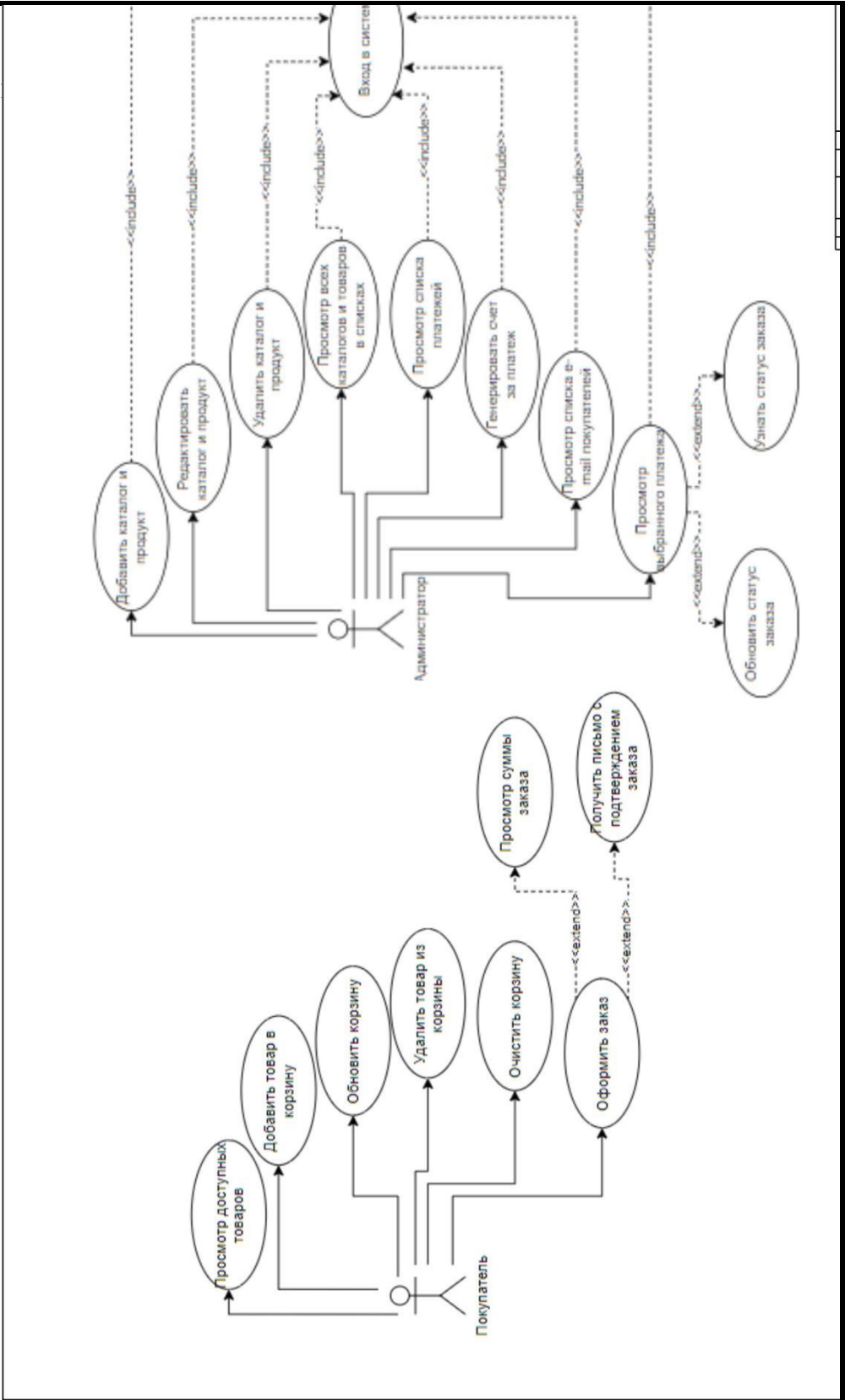
UML-диаграммы

					ПГУ1.090301.03.001ПЗ	Лист
						72
Изм.	Лист	№докум.	Подпись	Дата		

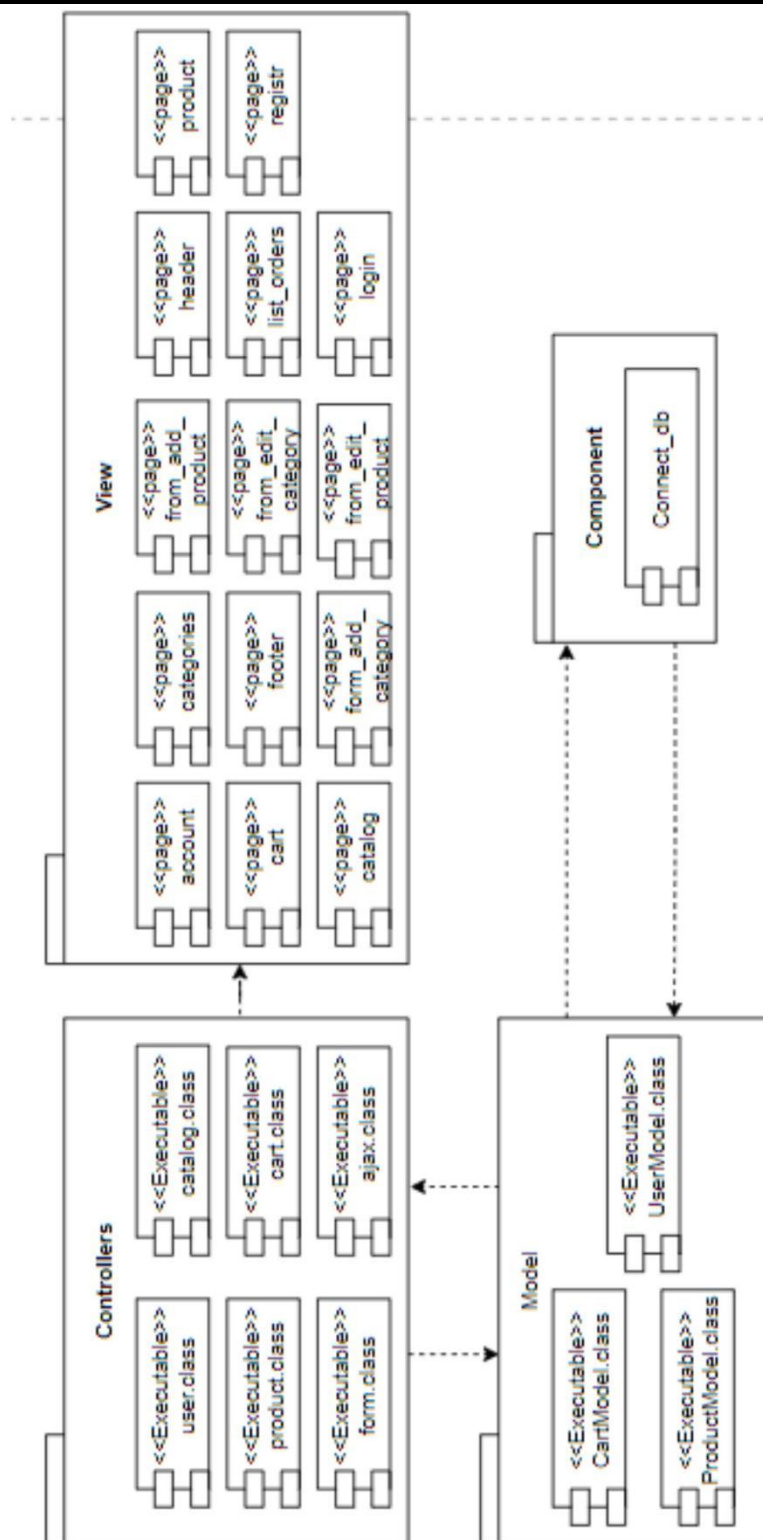
Приложение А.1



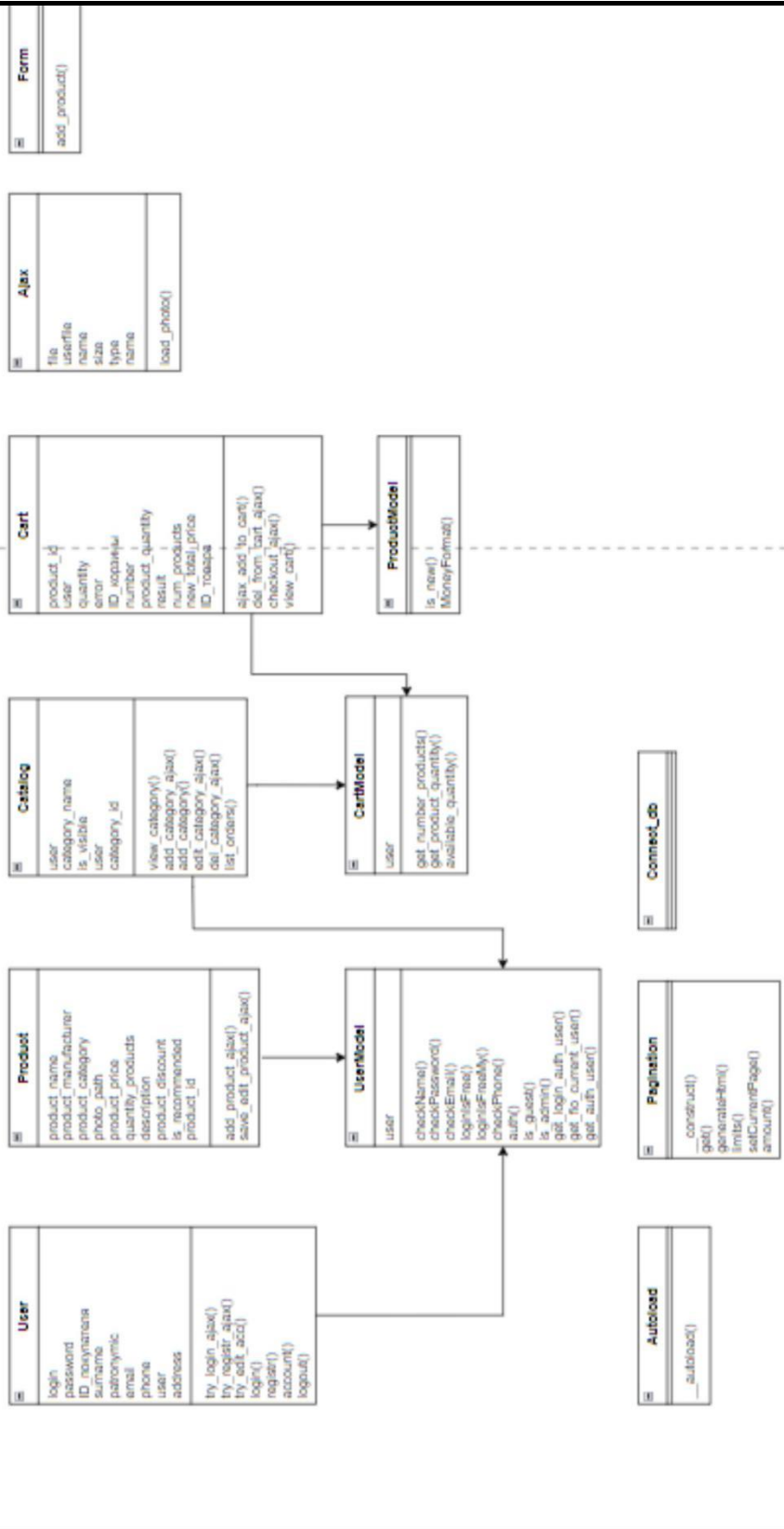
Приложение А.2 - Д



Приложение А.



Приложение А.



ПриложениеБ

(Обязательное)

Web-портал интернет-магазина

Листингпрограммы

					ПГУ1.090301.03.001ПЗ	Лист
Изм.	Лист	№докум.	Подпись	Дата		77

Файл«*Pagination.php*»

```

<?

// Класс для генерации постраничной навигации
class Pagination
{
    // @var Максимальное число ссылок навигации на странице
    private $max = 10;

    // @var Ключ для GET, в который пишется номер страницы
    private $index = 'page-';

    // @var Текущая страница
    private $current_page;

    // @var Общее количество записей
    private $total;

    // @var Записей на странице
    private $limit;

    /**
     * Запуск необходимых данных для навигации
     * @param integer $total - общее количество записей
     * @param integer $limit - количество записей на
    страницу
     *
     * @return
     */

    public function __construct($total, $currentPage,
    $limit, $index)
    {

        # Устанавливаем общее количество записей
        $this->total = $total;

        # Устанавливаем количество записей на страницу
        $this->limit = $limit;
    }
}

```

```

$this->index = $index;

        # Устанавливаем количество страниц
        $this->amount = $this->amount();

        # Устанавливаем номер текущей страницы
$this->setCurrentPage($currentPage);

    }

    /**
     * Для вывода ссылок
    *
     * @return HTML-код со ссылками навигации
    */

    public function get()
    {
        if ($this->amount > 1)
        {
            # Для записи ссылок
            $links = null;

            # Получаем ограничения для цикла
            $limits = $this->limits();

            $html = '<ul class="pagination">';

            # Генерируем ссылки
            for ($page = $limits[0]; $page <= $limits[1];
            $page++)
            {
                # Если текущая это текущая страница, ссылки
                нет и добавляется класс active
                if ($page == $this->current_page)
                {
                    $links .= '<li class="active"><a
                    href="">' . $page . '</a></li>';
                }
            }
        }
    }

```

```

else
{
    # Иначе генерируем ссылку
    $links .= $this->generateHtml($page);
}
}

# Если ссылки создались
if (!is_null($links))
{
    # Если текущая страница не первая
    if ($this->current_page > 1)
        # Создаёмссылку"Напервую"
        $links = $this->generateHtml(1, '&lt;');
    . $links;

    # Еслитекущаястраницанепервая
    if ($this->current_page < $this->amount)
        # Создаём ссылку "На последнюю"
        $links .= $this->generateHtml($this-
        >amount, '&gt;');
    }
    $html .= $links . '</ul>';
    # Возвращаем html
    return $html;
}

elsereturn;
}

/**
 * Для генерации HTML-кода ссылки
 * @param integer $page - номер страницы
 * @return

```

```

        # Конец - общее количество страниц

        $end = $this->amount;

        # Начало - минус $this->max от конца
        $start = $this->amount - $this->max > 0 ? $this->amount - $this->max : 1;

    }

    # Возвращаем
    return array($start, $end);
}

// Устанавливает текущую страницу
private function setCurrentPage($currentPage)
{
    # Получаем номер страницы
    $this->current_page = $currentPage;

    # Если текущая страница более нуля
    if ($this->current_page > 0)
    {
        # Если текущая страница меньше общего количества страниц
        if ($this->current_page > $this->amount)
        # Устанавливаем страницу на последнюю
            $this->current_page = $this->amount;
    }

    # Устанавливаем страницу на первую
    else $this->current_page = 1;
}

// Возвращает общее число страниц
private function amount()
{
    # Делим и возвращаем
    return ceil($this->total / $this->limit);
}

```

Файл«Autoload.php»

```
<?

function __autoload($class_name)
{
    # List all the class directories in the array.
    $array_paths = array('models/', 'components/');
    foreach ($array_paths as $path)
    {
        $path = PATH . $path . $class_name . '.php';
        if (is_file($path))
        {
            include_once $path;
        }
    }
}

?>
```

Файл«Connect_db.php»

```
<?

header('Content-Type: text/html; charset=utf-8');

$host = 'localhost';
$user = 'arsentp3_shop';
$database = 'arsentp3_shop';
$db_password = '1234q!';

$db = new mysqli($host, $user, $db_password, $database);

?>
```

Файл«Ajax.php»

```
<?

class ajax
{
    public static function load_photo()
    {
        if(isset($_POST['file']))
        {
            define(PATH, dirname(__FILE__));

            $uploadDir = PATH . "/images/product_photos/";

            $types = array("image/gif",
"image/png","image/jpeg","image/pjpeg", "image/p-png");

            $size = 2097152; //2 мб

            $file = $_FILES['userfile']['name'];
            if(!isset($file))
            {
                $res = array("answer"=>"Ошибка! Возможно
файл слишком большой");
                exit(json_encode($res));
            }

            if($_FILES['userfile']['size'] > $size ||
$_FILES['userfile']['size'] == 0)
            {
                $res = array("answer"=>"Ошибка! максимальный
вес 2 мб");
                exit(json_encode($res));
                exit("file");
            }

            if(!in_array($_FILES['userfile']['type'],
$types))
            {
```

```

        $res = array("answer" =>"Ошибка! Допустимые расширение- .gif,
        .png, .jpg");
    }

    if(move_uploaded_file($_FILES['userfile']['tmp_name'],
    $uploadDir . $file))
    {
        $res = array("answer" =>
    $_FILES['userfile']['name'], "file" => $file);
        exit(json_encode($res));
    }
    else
    {
        $res = array("answer" =>"Ошибказагрузки",
    "file" => $file);
        exit(json_encode($res));
    }
    }
}
?>

```

Файл«Cart.php»

```

<?
class cart
{
    public static function ajax_add_to_cart()
    {
        global $db;
        $product_id = $_POST['product_id'];
        $user_id = $_SESSION['user'];
        $quantity = $_POST['quantity'];

        $result = ['number' => '', 'error' => '',
    'product_quantity' => ''];

        if ($quantity < 0)
        {

```

```

        $result['error'] = "Значение не может быть
отрицательным!";
    else if ($quantity > ($available =
    CartModel::available_quantity($product_id)))
    {
        if ($available == 0) $result['error'] =
        "Недостаточно товаров на складе! У вас в корзине находятся все
        единицы данного товара со склада.";
        else $result['error'] = "Недостаточно товаров
        на складе! Оставшееся количество данного товара на складе:
        $available.";
    }
    else
    {
        //Проверка на существование товара в корзине
        пользователя
        $exist = $db -> query("SELECT * FROM ShoppingCart WHERE
        ID_покупателя='$user_id' AND ID_товара='$product_id'");
        if ($exist -> num_rows)
        {
            //Если уже существует, то увеличиваем
            количество товаров в корзине
            $id_cart = $exist -> fetch_array()['ID_корзины'];
            $db -> query("UPDATE ShoppingCart SET
            Количество = Количество + '$quantity' WHERE
            ID_корзины='$id_cart'");
        }
        else
        {
            //Если не существует, то добавляем новый
            товар в корзину
            $db -> query("INSERT INTO ShoppingCart (ID_покупателя,
            ID_товара, Количество) VALUES('$user_id', '$product_id',
            '$quantity')");
        }
        $result['number'] =
        '('.CartModel::get_number_products($user_id).')';
        $result['product_quantity'] =
        CartModel::get_product_quantity($product_id);
        exit(json_encode($result));
    }

    //Выполняет удаление товара из корзины (Возвращает
    результат выполнения запроса на удаление, новую суммарную
    строимость всех товаров в корзине, число товаров в корзине)
    public static function del_from_cart_ajax($cart_id)
    {
        global $db;
    }

```

```

        $user_id = $_SESSION['user'];
        $result = $db -> query("DELETE FROM ShoppingCart
WHERE ID_корзины=$cart_id");
        $total_sum = $db -> query("SELECT SUM(Количество * (1 -
Скидка / 100) * Цена) FROM Products INNER JOIN
ShoppingCart ON Products.ID_товара=ShoppingCart.ID_товара WHERE
ID_покупателя=$user_id")->fetch_row()[0];
        $total_sum = ProductModel::MoneyFormat($total_sum);
        $num_products =
CartModel::get_number_products($user_id);
        exit(json_encode(['result' => $result, 'total' =>"₽
$total_sum", 'num_products' =>"($num_products)"]));
    }

    //Выполняет проверку на введенное количество товара в
корзине и обновляет это количество в базе данных
    //Выдает ошибку, если введенное количество товара
превышает количество товаров на складе
    //В случае, если введенное значение корректно,
обновляются значения стоимости и товаров в корзине
    public static function test_quantity_product_cart($cart_id,
$quantity)
    {
        global $db;
        //Ошибка, новое количество выбранного товара, общее
число товаров в корзине, новая итоговая стоимость товара, новая
суммарная стоимость всех товаров
        $result = ['error' => '', 'new_quantity' => '', 'num_products'
=> '', 'new_price' => '', 'new_total_price' => ''];
        $product_id = $db -> query("SELECT ID_товара FROM
ShoppingCart WHERE ID_корзины='$cart_id'") -> fetch_row()[0];
        //Получение идентификатора выбранного товара
        $quantity_in_stock = $db -> query("SELECT На_складе
FROM Products WHERE ID_товара='$product_id'") ->
fetch_row()[0]; //Количество данного товара на складе
        if ($quantity > $quantity_in_stock)
        {
            $result['error'] = 'Введенное количество
товаров превышает количество товаров на складе!';
            $quantity = $result['new_quantity'] = $quantity_in_stock;
        }

        else if ($quantity < 0)
        {
            $result['error'] = 'Значение не должно быть
отрицательным!';
            $quantity = $result['new_quantity'] = 0;
        }
        $db -> query("UPDATE ShoppingCart SET
Количество=$quantity WHERE ID_корзины='$cart_id'");
        $user_id = $_SESSION['user'];

```

```

        $result['num_products'] = '(' .
CartModel::get_number_products($user_id) . ')';
$new_product_price = $db -> query("SELECT Количество * (1 -
Скидка / 100) * Цена FROM Products INNER JOIN ShoppingCart ON
Products.ID_товара=ShoppingCart.ID_товара WHERE
ID_корзины='$cart_id'");
        $result['new_price'] = '₽ ' .
ProductModel::MoneyFormat($new_product_price ->
fetch_row()[0]);
        $new_total_price = $db -> query("SELECT
SUM(Количество * (1 - Скидка / 100) * Цена) FROM Products INNER
JOIN ShoppingCart ON Products.ID_товара=ShoppingCart.ID_товара
WHERE ID_покупателя='$user_id'");
        $result['new_total_price'] = '₽ ' .
ProductModel::MoneyFormat($new_total_price -> fetch_row()[0]);
exit(json_encode($result));
    }
    //Выполняет всинхронно оформление заказа (сохранение
информации о заказе в базу данных и очищение корзины
пользователя)
    public static function checkout_ajax()
    {
        global $db;
        $user_id = $_SESSION['user'];
        $cart = $db -> query("SELECT * FROM ShoppingCart
WHERE ID_покупателя='$user_id'");
        $cart_products = [];
        while($product = $cart -> fetch_array())
            $cart_products[$product['ID_товара']] =
$product['Количество'];
        $order = json_encode($cart_products);
        if ($cart_products != [])
        {
            $db -> query("INSERT INTO Orders
(ID_покупателя, Товары, Дата_время_заказа) VALUES ('$user_id',
'$order', CURRENT_TIMESTAMP())");
            $db -> query("DELETE FROM ShoppingCart WHERE
ID_покупателя='$user_id'");

            // Уменьшает число товаров на складе у каждого купленного
товара
            foreach($cart_products as $product_id => $quantity)
                $db -> query("UPDATE Products SET
На_складе=На_складе-'$quantity' WHERE
ID_товара='$product_id'");

            exit('1');
        }
        else exit('Для оформления заказа необходимо
добавить товары в корзину!');
    }

```

```

        public static function view_cart()
        {
            global $controller, $db;
            $user_id = $_SESSION['user'];
            $cart_products = $db -> query("SELECT *, (1 -
Скидка / 100) * Цена AS Сумма_со_скидкой, Количество * (1 -
Скидка / 100) * Цена AS Итоговая_сумма FROM Products INNER JOIN
ShoppingCart ON Products.ID_товара=ShoppingCart.ID_товара WHERE
ID_покупателя='$user_id'");

            require_once(PATH . 'views/cart.php');
            return true;
        }
    }
?>

```

Файл«Catalog.php»

```

<?
class catalog
{
    const SHOW_LIMIT = 9;

    //Просмотртоваровкатегории
    public static function view_category($id_category, $page)
    {
        if(!isset($_SESSION['user']))
            header("Location: /login");

        global $db;
        global $controller;
        if(UserModel::is_admin())
            $categories = $db -> query('SELECT * FROM
Categories');
        else $categories = $db -> query('SELECT * FROM
Categories WHERE IsVisible=1');

        $limit = self::SHOW_LIMIT;
        $offset = ($page - 1) * $limit;
        $products = $db -> query("SELECT * FROM Products WHERE
ID_категории='$id_category' ORDER BY ID_товара LIMIT $limit OFFSET
$offset");

        $total = $db -> query("SELECT COUNT(ID_категории) FROM
Products WHERE ID_категории='$id_category'") -> fetch_row()[0];
        $pagination = new Pagination($total, $page, $limit,
'page-');

        require_once(PATH . 'views/catalog.php');
        return true;
    }
}

```

```

public static function view_product($id_category, $id_product)
{
    if(!isset($_SESSION['user']))
        header("Location: /login");
    global $db;
    global $controller;
    if(UserModel::is_admin())
        $categories = $db -> query('SELECT * FROM
Categories');
    else $categories = $db -> query('SELECT * FROM
Categories WHERE isVisible=1');
    $product = $db -> query("SELECT * FROM Products WHERE
ID_товара=$id_product AND ID_категории='$id_category'") ->
fetch_array();
    CartModel::available_quantity($product['ID_товара']);
    require_once(PATH . 'views/product.php');
    return true;
}
public static function add_category_ajax()
{
    if(UserModel::is_admin())
    {
        global $db;
        $category_name = $_POST['category_name'];
        $is_visible = $_POST['is_visible'] == 'true';
        $r = $db -> query("INSERT INTO Categories
(Название, isVisible) VALUES ('$category_name', '$is_visible')");

        $action = 'add_category';
        if(UserModel::is_admin())
            $categories = $db -> query('SELECT * FROM
Categories');
        else $categories = $db -> query('SELECT * FROM
Categories WHERE isVisible=1');
        include(PATH . 'views/categories.php');
        return true;
    }
}
public static function add_category()
{
    if(!isset($_SESSION['user']))
        header("Location: /login");
    global $db, $controller, $action;
    if(UserModel::is_admin())
        $categories = $db -> query('SELECT * FROM
Categories');
    else $categories = $db -> query('SELECT * FROM
Categories WHERE isVisible=1');
    require_once(PATH . 'views/form_add_category.php');
    return true;
}

```

```

        public static function edit_category($id_category)
        {
            if(!isset($_SESSION['user']))
                header("Location: /login");

            global $db, $controller;
            if(UserModel::is_admin())
                $categories = $db -> query('SELECT * FROM
Categories');
            else $categories = $db -> query('SELECT * FROM
Categories WHERE IsVisible=1');
            $category = $db -> query("SELECT * FROM Categories
WHERE ID_категории='$id_category'") -> fetch_array();

            require_once(PATH . 'views/form_edit_category.php');
            return true;
        }

        public static function edit_category_ajax()
        {
            if(UserModel::is_admin())
            {
                global $db;
                $id_category = $_POST['category_id'];
                $category_name = $_POST['category_name'];
                $is_visible = $_POST['is_visible'] == 'true';
                $r = $db -> query("UPDATE Categories SET
Название='$category_name', IsVisible='$is_visible' WHERE
ID_категории='$id_category'");

                $categories = $db -> query('SELECT * FROM
Categories');

                include(PATH . 'views/categories.php');
                return true;
            }
        }

        public static function del_category_ajax($id_category)
        {
            if(UserModel::is_admin())
            {
                global $db;
                //Проверка, что категория не содержит товаров
                $is_empty = $db -> query("SELECT COUNT(*) FROM Products WHERE
ID_категории='$id_category'") -> fetch_row()[0] == 0;
                if ($is_empty)
                {
                    $result = $db -> query("DELETE FROM Categories
WHERE ID_категории='$id_category'");
                    exit($result);
                }
                else exit('Категория содержит товары! Перед
удалением категории необходимо удалить все товары категории.');
```

```

    }
}
public static function list_orders()
{
    if(UserModel::is_admin())
    {
        global $db, $action;
        $users_orders = $db ->query("SELECT * FROM Orders
INNER JOIN Customers ON
Orders.ID_покупателя=Customers.ID_покупателя ORDER BY
Дата_время_заказа DESC");
        include(PATH . 'views/list_orders.php');
    }
}
}
?>

```

Файл«Form.php»

```

<?
class form
{
    public static function add_product()
    {
        global $db, $action;
        $categories = $db -> query('SELECT * FROM Categories');
        require_once(PATH . 'views/form_add_product.php');
        return true;
    }
    public static function edit_product($product_id)
    {
        global $db;
        $categories = $db -> query('SELECT * FROM Categories');
        $product = $db -> query("SELECT * FROM Products WHERE
ID_товара='$product_id'") -> fetch_array();
        require_once(PATH . 'views/form_edit_product.php');
        return true;
    }
}
?>

```

Файл«Product.php»

```

<?
class product
{
    //Добавление товара в БД при нажатии на кнопку "Добавить
товар"
    public static function add_product_ajax()

```

					ПГУ1.090301.03.001ПЗ	Лист
						91
Изм.	Лист	№докум.	Подпись	Дата		

```

        if (UserModel::is_admin())
        {
            global $db;
            $product_name = $_POST['product_name'];
            $product_manufacturer =
$_POST['product_manufacturer'];
            $product_category = $_POST['product_category'];
            $photo_path = $_POST['photo_path'];
            $product_price = $_POST['product_price'];
            $quantity_products = $_POST['quantity_products'];
            $description = $_POST['description'];
            $product_discount = $_POST['product_discount'];
            $is_recommended = $_POST['is_recommended'] ==
'true';
            echo $db -> query("INSERT INTO Products (Название, Производитель,
ID_категории, Фото, Цена, На_складе, Описание, Скидка,
IsRecommended, Дата_добавления)
VALUES ('$product_name', '$product_manufacturer',
'$product_category', '$photo_path', '$product_price',
'$quantity_products', '$description', '$product_discount',
'$is_recommended', CURRENT_DATE())");
        }
        // Сохранение изменений в БД при нажатии на кнопку
        "Сохранить изменения" (Редактирование товара)
        public static function save_edit_product_ajax()
        {
            if (UserModel::is_admin())
            {
                global $db;
                $product_id = $_POST['product_id'];
                $product_name = $_POST['product_name'];
                $product_manufacturer =
$_POST['product_manufacturer'];
                $product_category = $_POST['product_category'];
                $photo_path = $_POST['photo_path'];
                $product_price = $_POST['product_price'];
                $quantity_products = $_POST['quantity_products'];
                $description = $_POST['description'];
                $product_discount = $_POST['product_discount'];
                $is_recommended = $_POST['is_recommended'] ==
'true';
                echo $db -> query("UPDATE Products SET
Название='$product_name', Производитель='$product_manufacturer',
ID_категории='$product_category', Фото='$photo_path',
Цена='$product_price', На_складе='$quantity_products',
Описание='$description', Скидка='$product_discount',
IsRecommended='$is_recommended' WHERE ID_товара='$product_id'");
            }
        }
    }
}

```

```

        // Удаление товара из БД при нажатии на кнопку
        "Удалить товар" (Каталог в режиме администратора)
        public static function del_product_ajax($product_id)
        {
            if (UserModel::is_admin())
            {
                global $db;
                echo $db->query("DELETE FROM Products WHERE
ID_товара='$product_id'");
            }
        }
    }
?>

```

Файл«User.php»

```

<?
class user
{
    //Выполняетпопыткувойтинасайт
    public static function try_login_ajax()
    {
        global $db;
        $errors = ['login' => '', 'password' => ''];
        $login = $_POST['login'];
        $password = $_POST['password'];
        if (!UserModel::checkPassword($password))
        $errors['password'] = 'Парольдолженсодержатьне менее 4-хсимволов!';
        if (!UserModel::checkName($login)) $errors['login'] = 'Логин должен
быть не короче 2-х символов!';
        else if (UserModel::loginIsFree($login))
        {
            $errors['login'] = 'Пользователя с введенным логином не
существует!';
        }
        else
        {
            $user = $db->query("SELECT * FROM Customers WHERE
Логин='$login' AND Пароль='$password'");
            if ($user->num_rows)
            {
                $user = $user->fetch_array();
                UserModel::auth($user['ID_покупателя']);
            }
            else
            {
                $errors['password'] = 'Введен неверный пароль!';
            }
        }
    }
}

```

```

        exit(json_encode($errors));
    }
    //Выполняет попытку выполнить регистрацию
public static function try_registr_ajax()
{
    global $db;
    $errors = ['surname' => '', 'name' => '', 'patronymic' =>
'', 'login' => '', 'password' => '', 'email' => ''];
    $surname = $_POST['surname'];
    $name = $_POST['name'];
    $patronymic = $_POST['patronymic'];
    $login = $_POST['login'];
    $password = $_POST['password'];
    $email = $_POST['email'];

    if (!UserModel::checkName($surname)) $errors['surname'] = 'Фамилия
должна быть не короче 2-х символов!';
    if (!UserModel::checkName($name)) $errors['name'] = 'Имя
должно быть не короче 2-х символов!';
    if (!UserModel::checkName($patronymic))
$errors['patronymic'] = 'Отчество должно быть не короче 2-х символов!';

    if (!UserModel::checkName($login)) $errors['login'] =
'Логин должен быть не короче 2-х символов!';
    if (!UserModel::checkEmail($email)) $errors['email'] =
'Введеннекорректный e-mail!';
    if (!UserModel::checkPassword($password))
$errors['password'] = 'Пароль должен содержать не менее 4-х символов!';

    if (array_values($errors) == ['', '', '', '', '', ''])
    {

        if (UserModel::loginIsFree($login))
        {
            $date = date("Y-m-d H:i:s");
            $db->query("INSERT INTO Customers (Фамилия, Имя,
Отчество, Логин, Пароль, Почта, Дата_Регистрации) Values('$surname',
'$name', '$patronymic', '$login', '$password', '$email', '$date')");
        }

        else
        {
            $errors['login'] = 'Введенный логин уже занят!';
        }
    }
    exit(json_encode($errors));
}

//Выполняет попытку сохранить изменения в аккаунте
public static function try_edit_acc()
{

```

```

        global $db;
        $errors = ['surname' => '', 'name' => '', 'patronymic' =>
'', 'login' => '', 'password' => '', 'email' => '', 'phone' => ''];

        $id = $_SESSION['user'];
        $original_login = UserModel::get_login_auth_user();
        $surname = $_POST['surname'];
        $name = $_POST['name'];
        $patronymic = $_POST['patronymic'];
        $login = $_POST['login'];
        $password = $_POST['password'];
        $email = $_POST['email'];
        $phone = $_POST['phone'];
        $adress = $_POST['address'];

if (!UserModel::checkName($surname)) $errors['surname'] = 'Фамилия
должна быть не короче 2-х символов!';
        if (!UserModel::checkName($name)) $errors['name'] = 'Имя
должно быть не короче 2-х символов!';
        if (!UserModel::checkName($patronymic))
$errors['patronymic'] = 'Отчество должно быть не короче 2-х символов!';
        if (!UserModel::checkName($login)) $errors['login'] =
'Логин должен быть не короче 2-х символов!';
if (!UserModel::checkEmail($email)) $errors['email'] =
'Введеннекорректный e-mail!';
        if (!UserModel::checkPassword($password))
$errors['password'] = 'Парольдолженсодержатьне менее 4-хсимволов!';
        if (!empty($phone) && !UserModel::checkPhone($phone))
$errors['phone'] = 'Некорректноуказанномертелефона!';

        if (array_values($errors) == ['', '', '', '', '', '', ''])
        {
            if (UserModel::loginIsFreeMy($login, $original_login))
            {
                $db->query("UPDATE Customers SET
Фамилия='$surname', Имя='$name', Отчество='$patronymic',
Логин='$login', Пароль='$password', Почта='$email', Телефон='$phone',
Адрес='$adress' WHERE ID_покупателя='$id'");
            }

            else
            {
                $errors['login'] = 'Введенный логин уже занят!';
            }
        }
        exit(json_encode($errors));
    }

    public static function login()
    {
        if(isset($_SESSION['user']))
            header("Location: /catalog/1/page-1");
    }

```

```

        global $action;
        require_once(PATH . 'views/login.php');
        return true;
    }
    public static function registr()
    {
        if(isset($_SESSION['user']))
            header("Location: /catalog/1/page-1");

        global $action;
        require_once(PATH . 'views/registr.php');
        return true;
    }
    public static function account()
    {
        if(!isset($_SESSION['user']))
            header("Location: /login");
        $user = UserModel::get_auth_user();
        global $action;
        require_once(PATH . 'views/account.php');
        return true;
    }
    //Выход
    public static function logout()
    {
        unset($_SESSION['user']);
        header("Location: /");
        return true;
    }
}
?>

```

Файл«Main.css»

```

@import
url(http://fonts.googleapis.com/css?family=Roboto:400,300,400italic
,500,700,100&subset=latin,cyrillic-ext,cyrillic);
@import
url(http://fonts.googleapis.com/css?family=Open+Sans:400,800,300,60
0,700);
@import url(http://fonts.googleapis.com/css?family=Abel);
body {

    font-family: 'Roboto', sans-serif;
    position: relative;
    font-weight:400px;
    background: aliceblue;
}
.center {

```

```

        text-align: center;
        font-size: 14px;
    }
    #errors{
        text-align: center;
    }
    ul li {
        list-style: none;
    }
    a:hover {
        outline: none;
        text-decoration:none;
    }
    a:focus {
        outline:none;
        outline-offset: 0;
    }
    a {
        -webkit-transition: 300ms;
        -moz-transition: 300ms;
        -o-transition: 300ms;
        transition: 300ms;
    }
    h1, h2, h3, h4, h5, h6 {
        font-family: 'Roboto', sans-serif;
    }
    .btn:hover,
    .btn:focus{
        outline: none;
        box-shadow: none;
    }
    /*****Header CSS*****/
    /*****Header CSS*****/
    .header_top {
        background: none repeat scroll 0 0 #2f2f2f;
    }
    .contactinfo ul li:first-child{
        margin-left: -15px;
    }
    .contactinfo ul li a{
        font-size: 12px;
        color: lightskyblue;
        font-family: 'Roboto', sans-serif;
    }
    .form-category{
        overflow: hidden;
        padding-bottom: 19px;
        padding-left: 50px;
        padding-top: 35px;
        position: relative;
        background: #2f2f2f;

```

```

        color: aliceblue;
        padding-right: 50px;
    }
    .contactinfo p
    {
        padding-top: 10px;
        padding-left: 15px;
        font-size: 14px;
        color: lightskyblue;
        font-family: 'Roboto', sans-serif;
        margin-left: 15px;
    }
    .admin-elements{
        white-space: nowrap;
    }
    .admin-elements a:hover{
        cursor: pointer;
    }
    .contactinfo ul li a:hover{
        background:inherit;
    }
    .header-middle .container .row {
        border-bottom: 1px solid #f5f5f5;
        margin-left: 0;
        margin-right: 0;
        padding-bottom: 50px;
        padding-top: 20px;
    }
    .header-middle .container .row .col-sm-4{
        padding-left: 0;
    }
    .header-middle .container .row .col-sm-8 {
        padding-right:0;
    }
    .shop-menu ul li {
        display:inline-block;
        padding-left: 15px;
        padding-right: 15px;
    }
    .shop-menu ul li:last-child {
        padding-right: 0;
    }

    .shop-menu ul li a {
        color: #696763;
        font-family: 'Roboto', sans-serif;
        font-size: 14px;
        font-weight: 300;
        padding:0;
        padding-right: 0;
        margin-top: 10px;
    }

```

```

margin-right:3px;
}
.shop-menu ul li a:hover {
color:#fe980f;
background: transparent;
}
.mainmenu ul li a:hover, .mainmenu ul li a.active, .shop-menu ul
li a.active{
background:none;
color:#ff8a00;
}
/* Dropdown menu*/

.nav.navbar-nav > li:hover > ul.sub-menu{
display: block;
-webkit-animation: fadeInUp 400ms;
-moz-animation: fadeInUp 400ms;
-ms-animation: fadeInUp 400ms;
-o-animation: fadeInUp 400ms;
animation: fadeInUp 400ms;
}
.pull-center {
padding-right: 10px;
padding-left: 10px;
}
.pull-center img{
max-width: 100%;
border-radius: 20px;
padding: 5px;
}
.pull-center img:hover{
opacity: .75;
}
.not-hover img:hover{
opacity: 1;
}
.signup-row{
padding-top: 10px;
border-radius: 10px;
background: #2f2f2f;
}
.signup-col{
background: #2f2f2f;
padding: 15px;
border-radius: 20px;
}
.view-product img {
max-width: 100%;
border: 1px solid #F7F7F0;
}
.pull-center img, .view-product img, .pull-center input, .pull-
center select, .pull-center textarea{ margin: auto;

```

```

        display: block;
        border-radius: 20px;
        z-index: 3;}
.border_img{
    background: #2f2f2f;
    padding: 5px;
}
.pull-center input, .pull-center select, .pull-center textarea{
    border-radius: 10px;
    color: #2f2f2f;
}
.navbar-nav li ul.sub-menu li{
    padding: 10px 20px 0 ;
}
.navbar-nav li ul.sub-menu li:last-child{
    padding-bottom: 20px;
}
.navbar-nav li ul.sub-menu li a{
    color: #fff;
}
.navbar-nav li ul.sub-menu li a:hover{
    color: #FDB45E;
}
@-webkit-keyframes fadeInUp {
    0% {
        opacity: 0;
        -webkit-transform: translateY(20px);
        transform: translateY(20px);
    }
    100% {
        opacity: 1;
        -webkit-transform: translateY(0);
        transform: translateY(0);
    }
}
/*****Footer CSS*****/
*****/
#footer {
    position:fixed;
    left:0;
    bottom:0;
    width:100%;
    background: #F0F0E9;
    z-index: 1000;
}
.footer-bottom {
    background: #2f2f2f;
    padding-top: 10px;
}
.footer-bottom p {
    color: lightskyblue;

```

```

    font-family: 'Roboto', sans-serif;
    font-size: 14px;
    margin-left: 15px;
}
.footer-bottom p span a {
    color: #FE980F;
    font-style: italic;
    text-decoration: underline;
}
/***** Home *****/
/*****/
.right {
    right: 0;
}
.category-products {
    border: 1px solid #F7F7F0;
    margin-bottom: 35px;
    padding-bottom: 20px;
    padding-top: 15px;
    background: #2f2f2f;
    border-radius: 20px;
}
.left-sidebar h2, .signup-form h2, .cart_form{
    color: #FE980F;
    font-family: 'Roboto', sans-serif;
    font-size: 18px;
    font-weight: 700;
    margin: 10px auto 28px;
    text-align: center;
    text-transform: uppercase;
    position: relative;
    z-index: 3;
}
.left-sidebar h2:after, h2.title:after{
    content: "";
    position: absolute;
    border: 1px solid #f5f5f5;
    bottom: 8px;
    left: 0;
    width: 100%;
    height: 0;
    z-index: -2;
}
.left-sidebar h2:before{
    content: "";
    position: absolute;
    bottom: -6px;
    width: 130px;
    height: 30px;
    z-index: -1;
    left: 50%;
    margin-left: -65px;
}

```

```

}
h2.title:before{
  content: "";
  position: absolute;
  bottom: -6px;
  width: 220px;
  height: 30px;
  z-index: -1;
  left: 50%;
  margin-left: -110px;
}
.category-products .panel {
  color: #f8db98;
  background-color: transparent;
  border: 0px;
  border-radius: 0px;
  box-shadow: none;
  margin-bottom: 0px;
}
.category-products .panel-default .panel-heading {
  background-color: transparent;
  border: 0 none;
  color: #FFFFFF;
  padding: 5px 20px;
}
.category-products .panel-default .panel-heading .panel-title a {
  color: aliceblue;
  font-family: 'Roboto', sans-serif;
  font-size: 14px;
  text-decoration: none;
  text-transform: uppercase;
  padding-left: 5px;
}
.category-products .panel-default .panel-heading .panel-title
a.active,
.category-products .panel-default .panel-heading .panel-title
a:hover{
  color: #FE980F;
}
.panel-group .panel-heading + .panel-collapse .panel-body {
  border-top: 0 none;
}
.padding-right {
  padding-right: 0;
}
.features_items{
  overflow: hidden;
}
h2.title {
  color: #FE980F;
  font-family: 'Roboto', sans-serif;
  font-size: 18px;

```

```

margin: 10px 15px;
text-transform: uppercase;
margin-bottom: 30px;
position: relative;
}
.product-image-wrapper{
overflow: hidden;
margin-bottom:30px;
}
.single-products {
position: relative;
}
.productinfo h2{
color: #FE980F;
font-family: 'Roboto', sans-serif;
font-size: 24px;
font-weight: 700;
margin-top: 10px;
margin-bottom: 5px;
}
.productinfo p{
font-family: 'Roboto', sans-serif;
font-size: 14px;
font-weight: 400;
color: aliceblue;
margin: 0 0 5px;
}
.productinfo{
background: #2F2F2F;
border-radius: 20px;
}
.single-products:hover {
display:block;
height:100%;
}
.admin-elem i{
margin-right: 0;
width: 20px;
}
.admin-elem{
margin-left: 7px;
margin-right: 7px;
}
.add-to-cart, .edit-product, .del-product {
background: aliceblue;
border: 0 none;
border-radius: 10px;
color: #2f2f2f;
font-family: 'Roboto', sans-serif;
font-size: 15px;
margin-bottom: 25px;

```

```

}
.add-to-cart:hover, .cart:hover {
    border: 0 none;
    border-radius: 10px;
    color: Black;
    opacity: .75;
}
.add-to-cart i{
    margin-right:5px;
}
/*****Shop CSS*****/
*****/

.pagination {
    display: inline-block;
    margin-bottom: 25px;
    margin-top: 0;
    padding: 15px;
    background: #2f2f2f;
    border-radius: 10px;
    margin-left: 10px;
}
.pagination li:first-child a, .pagination li:first-child span,
.pagination li:last-child a, .pagination li:last-child span{
    border-radius: 10px;
    margin-left: 0;
}
.pagination .active a, .pagination .active span, .pagination
.active a:hover, .pagination .active span:hover, .pagination
.active a:focus, .pagination .active span:focus {
    background-color: #FE980F;
    color: White;
    cursor: default;
    z-index: 2;
}
.pagination li a:hover, .pagination li span:hover{
    background-color: #FE980F;
    color: White;
}
.pagination li a, .pagination li span {
    background-color: aliceblue;
    border: 0;
    float: left;
    line-height: 1.42857;
    margin-left: -1px;
    padding: 6px 12px;
    position: relative;
    text-decoration: none;
    margin-right: 5px;
}

```

```

/*****
*****Product Details CSS*****
*****/
.product-details {
    margin-bottom: 40px;
    overflow: hidden;
}
.view-product {
    position: relative;
}
.product-information {
    overflow: hidden;
    padding-left: 60px;
    padding-top: 35px;
    position: relative;
    background: #2f2f2f;
    border-radius: 20px;
    color: aliceblue;
}
.newarrival{
    position:absolute;
    top: 4px;
    left: 4px;
    border-radius: 10px;
}
.recommended{
    position:absolute;
    top: 4px;
    right: 4px;
    width: 35px;
    height: 35px;
}
.product-information h2 {
    color: aliceblue;
    font-family: 'Roboto', sans-serif;
    font-size: 20px;
    margin-top: 0;
}
.product-information p {
    color: aliceblue;
    font-family: 'Roboto', sans-serif;
    margin-bottom: 5px;
}
.product-information span {
    display: inline-block;
    margin-bottom: 8px;
    margin-top: 18px;
}
.product-information span span {
    color: #FE980F;
    float: left;
    font-family: 'Roboto', sans-serif;

```

```

        font-size: 30px;
        font-weight: 700;
        margin-right: 20px;
        margin-top: 0px;
    }
    .product-information span input {
        border: 0;
        color: #2f2f2f;
        font-family: 'Roboto', sans-serif;
        font-size: 14px;
        font-weight: 700;
        outline: medium none;
        text-align: center;
        width: 55px;
    }
    label{
        color: aliceblue;
    }
    .product-information span label {
        color: aliceblue;
        font-family: 'Roboto', sans-serif;
        font-weight: 700;
        margin-right: 5px;
    }
    .cart {
        background: aliceblue;
        border: 0 none;
        color: #2f2f2f;
        font-family: 'Roboto', sans-serif;
        font-size: 15px;
        margin-bottom: 10px;
        margin-left: 20px;
        border-radius: 10px;
    }
    /*****login page CSS*****/
    /*****/
    #form {
        display: block;
        margin-bottom: 185px;
        margin-top: 185px;
        overflow: hidden;
    }
    .signup-form textarea{
        margin-bottom: 10px;
        height: 150px;
        padding-left: 8px;
        max-width: 600px;
        background: White;
        color: #2f2f2f;
        font-size: 15px; }

```

```

background: lavender;
border: medium none;
color: #2f2f2f;
display: block;
font-family: 'Roboto', sans-serif;
height: 40px;
margin-bottom: 10px;
margin-top: 2px;
padding-left: 10px;
width: 100%;
border-radius: 10px;
}
.signup-form input:hover, .signup-form select:hover{
    opacity: .9;
}
.signup-form input::placeholder,
.signup-form select::placeholder, .signup-form
textarea::placeholder{
    color: #2f2f2f;
}
.signup-form select {
    padding-left: 6px;
}
.signup-form img{
    margin-bottom: 10px;
    height: 300px;
}
table {
    width: 250px; /* Ширинатаблицы */
margin: auto; /* Выравниваем таблицу по центру окна */
}
td {
    text-align: center;
}
.long-center {
    width: 600px;
}
.left {
    width: 70%; /* Ширинатаблицы */
margin: 0;
}
.left td {
    text-align: left;
padding-left: 15px;
}
.left td:first-child {
padding-left: 0;
}
.signup-form .checkbox {
height: 16px;
width: 16px;
display: inline;

```

```

        background: #F0F0E9;
    }
    .signup-form button {
        background: #FE980F;
        border: medium none;
        border-radius: 0;
        color: #FFFFFF;
        display: block;
        font-family: 'Roboto', sans-serif;
        padding: 6px 25px;
    }
    .or{
        background: #FE980F;
        border-radius: 40px;
        color: #FFFFFF;
        font-family: 'Roboto', sans-serif;
        height: 50px;
        line-height: 50px;
        margin-top: 75px;
        text-align: center;
        width: 50px;
    }
/*****
*****Cart CSS*****
*****/
#cart_items .cart_info {
    border: 1px solid #E6E4DF;
    margin-bottom: 50px;
    background: #2f2f2f;
    border-radius: 20px;
}
.cart_info p{
    color: aliceblue;
}
#cart_items .cart_info .cart_menu {
    background: #FE980F;
    color: #fff;
    font-family: 'Roboto', sans-serif;
    font-weight: normal;
}
.table-condensed td{
    text-align: left;
}
.uniform-5{
    width: auto;
}
.uniform-5 tr td{
    width: 20%;
}
.wrap{
word-break: break-all;

```

```

}
.uniform{
    width: 10%;
}
.uniform-5 tr .center{
    text-align: center;
}
tr td .green-color{
    color: springgreen;
}
.table-condensed tr td .money{
    color: #FE980F;
}
#cart_items .cart_info .table.table-condensed thead tr {
    height: 51px;
}
#cart_items .cart_info .table.table-condensed tr{
    border: 1px solid White;
}
.cart_info table tr td {
    border-top: 0 none;
    vertical-align: inherit;
}
.table-condensed tbody tr .img{
    width: 30px;
    padding: 5px;}
#cart_items .cart_info .cart_description a {
    color: darkturquoise;
    font-family: 'Roboto', sans-serif;
    font-size: 16px;
    font-weight: normal;
}
#cart_items .cart_info .cart_total_price {
    color: #FE980F;
    font-size: 18px;
}
.cart_delete {
    margin-right: -12px;
    overflow: hidden;
}

.cart_delete a {
    background: #F0F0E9;
    color: #363432;
    padding: 5px 8px;
    font-size: 16px;
    border-radius: 10px;
}
.cart_delete a:hover {
    background: #FE980F
}

```

```
.user_info input, select, textarea {
    background: #F0F0E9;
    border:0;
    color: #696763;
    padding: 5px;
    width: 100%;
    border-radius: 0;
    resize: none
}
```

Файл«Main.js»

```
//Возвращает true, если строка пуста или состоит только из
пробелов
String.prototype.isEmpty = function()
{
    return (this.length === 0 || !this.trim());
};
$(document).ready(function()
{
    //При нажатии на кнопку "Вход" (Вход на сайт)
    $(".btn-login").click(function()
    {
        var params =
        {
            login: $("#user_login").val(),
            password: $("#user_password").val()
        };
        $.post("/try_login", params, function(data)
        {
            var errors = (JSON.parse(data));
            $("#error_login").html(errors.login);
            $("#error_password").html(errors.password);
            if ((errors.login + errors.password).isEmpty())
            {
                window.location.href = "/catalog/1/page-1";
            }
        });
    });
    //При нажатии на кнопку "Регистрация" (Регистрация на сайте)
    $(".btn-registr").click(function()
    {
        $("#success").html("");

        var params =
        {
            surname: $("#user_surname").val(),
            name: $("#user_name").val(),
```

```

        patronymic: $("#user_patronymic").val(),
        login: $("#user_login").val(),
        password: $("#user_password").val(),
        email: $("#user_email").val()
    };

$.post("/try_registr", params, function(data)
{
    var errors = (JSON.parse(data));
    $("#error_surname").html(errors.surname);
    $("#error_name").html(errors.name);
    $("#error_patronymic").html(errors.patronymic);
    $("#error_login").html(errors.login);
    $("#error_password").html(errors.password);
    $("#error_email").html(errors.email);
    if ((errors.surname + errors.name + errors.patronymic +
errors.login + errors.password + errors.email).isEmpty())
    {
        $("#success").html("Регистрация успешно
завершена");
    }

    });
});

//При нажатии на кнопку "Сохранить изменения" (Мой аккаунт)
$("#btn-edit-acc").click(function()
{
    $("#success").html("");

    var params =
    {
        surname: $("#user_surname").val(),
        name: $("#user_name").val(),
        patronymic: $("#user_patronymic").val(),
        login: $("#user_login").val(),
        password: $("#user_password").val(),
        email: $("#user_email").val(),
        adsress: $("#user_adsress").val(),
        phone: $("#user_phone").val(),
    };

    $.post("/edit_acc", params, function(data)
    {
        var errors = (JSON.parse(data));
        $("#error_surname").html(errors.surname);
        $("#error_name").html(errors.name);
        $("#error_patronymic").html(errors.

```

```

patronymic);
    $("#error_login").html(errors.login);
    $("#error_password").html(errors.password);
    $("#error_email").html(errors.email);
    $("#error_phone").html(errors.phone);

    if ((errors.surname + errors.name + errors.patronymic +
errors.login + errors.password + errors.email +
errors.phone).isEmpty())
    {
        $("#success").html("Изменения успешно сохранены");
    }
    });
});

//При нажатии на кнопку "Добавить товар"
$(".btn-add-product").click(function()
{
    $("#success").html("");
    $("#errors").html("");

    if ($("#product_name").val().isEmpty())
    {
        $("#errors").html("Необходимо указать название
товара");
    }
    else if ($("#product_manufacturer").val().isEmpty())
    {
        $("#errors").html("Необходимо указать производителя
товара");
    }
    else if ($("#product_category").val() == -
1)
    {
        $("#errors").html("Необходимо выбрать категорию
товара");
    }
    else if ($("#product_price").val().isEmpty())
    {
        $("#errors").html("Необходимо указать цену товара");
    }
    else if ($("#quantity_products").val().isEmpty())
    {
        $("#errors").html("Необходимо указать количество
товаров на складе");
    }
    else if ($("#product_discount").val().isEmpty())
    {
        $("#errors").html("Необходимо указать скидку товара");
    }

```

```

        }
        else if ($("#product_discount").val() > 100 ||
$("#product_discount").val() < 0)
{
            $("#errors").html("Скидка должна быть от 0% до 100%");
        }
        else
        {
            var photo_path = "";
            if ($("#btnUpload").val() != "Загрузка"&&
$("#btnUpload").val() != "Загрузить фото")
                photo_path = "/images/product_photos/" +
$("#btnUpload").val();

            var params =
            {
                product_name: ($("#product_name").val(),
                product_manufacturer:
$("#product_manufacturer").val(),
                product_category: ($("#product_category").val(),
                product_price: ($("#product_price").val(),
                quantity_products: ($("#quantity_products").val(),
                product_discount: ($("#product_discount").val(),
                description: ($("#description").val(),
                photo_path: photo_path,
                is_recommended:
$("#is_recommended").prop('checked'),
            };

            $.post("/add_product", params, function(data)
            {
                if (data == 1) $("#success").html('Успешно');
if (data == 0) $("#errors").html('Возникла неизвестная ошибка');
            });
        });

        //При нажатии на кнопку "Сохранить изменения" (Редактирование
товара)
$("#btn-save-edit-product").click(function()
{
    $("#success").html("");
    $("#errors").html("");

    if ($("#product_name").val().isEmpty())
    {
        $("#errors").html("Необходимо указать название
товара");
    }
}

```

```

    else if ($("#product_manufacturer").val().isEmpty())
    {
        $("#errors").html("Необходимо указать производителя
товара");
    }
    else if ($("#product_category").val() == -1)
    {
        $("#errors").html("Необходимо выбрать категорию
товара");
    }
    else if ($("#product_price").val().isEmpty())
    {
        $("#errors").html("Необходимо указать цену товара");
    }
    else if ($("#quantity_products").val().isEmpty())
    {
        $("#errors").html("Необходимо указать количество
товаров на складе");
    }
    else if ($("#product_discount").val().isEmpty())
    {
        $("#errors").html("Необходимо указать скидку товара");
    }
    else if ($("#product_discount").val() > 100 ||
$("#product_discount").val() < 0)
    {
        $("#errors").html("Скидка должна быть от 0% до 100%");
    }
    else
    {
        var photo_path = "";
        if ($("#btnUpload").val() != "Загрузка"&&
$("#btnUpload").val() != "Загрузить фото")
            photo_path = "/images/product_photos/" +
$("#btnUpload").val();

        var params =
        {
            product_id: $(this).attr('data_id'),
            product_name: $("#product_name").val(),
            product_manufacturer:
$("#product_manufacturer").val(),
            product_category: $("#product_category").val(),
            product_price: $("#product_price").val(),
            quantity_products: $("#quantity_products").val(),
            product_discount: $("#product_discount").val(),
            description: $("#description").val(),
            photo_path: photo_path,

```

```

$("#is_recommended").prop('checked'),
    });

    $.post("/save_edit_product", params, function(data)
    {
        if (data == 1) $("#success").html('Успешно');
if (data == 0) $("#errors").html('Возникла неизвестная ошибка');
    });
    });
    //При нажатии на кнопку "В корзину" (Каталог)
$("#.add-to-cart").click(function()
{
    var params =
    {
        product_id: $(this).attr('data_id'),
        quantity: 1
    };

    $.post("/add_to_cart/", params, function(data)
    {
        response = JSON.parse(data);
        $("#span-count").html(response.number);
        if (!response.error.isEmpty())
        {
            alert(response.error);
        }
    });
    });
    //При нажатии на кнопку "В корзину" (Информация о товаре)
    (Учитывается количество)
$("#.add-to-cart-quantity").click(function()
{
    var params =
    {
        product_id: $(this).attr('data_id'),
        quantity: $("#quantity-to-cart").val()
    };

    $.post("/add_to_cart/", params, function(data)
    {
        response = JSON.parse(data);
        $("#span-count").html(response.number);
        $("#num-in-cart").html(response.product_quantity);
        if (!response.error.isEmpty())
        {
            alert(response.error);
        }
    });
    });
    });

```

```

        //При нажатии на кнопку "x" (Удаление товара из корзины)
(Корзина)
$(".cart_quantity_delete").click(function()
{
    var id = $(this).attr('data_id');

    $.post("/del_from_cart/" + id, {}, function(data)
    {
        response = JSON.parse(data);
        if (response.result == 1)
        {
            $(".cart-product-" + id).remove();
            if(response.num_products != "()") $("#span-
count").html(response.num_products);
            else $("#span-count").html("");
            $(".cart_total_sum").html(response.total);
        }
    });
});
//При подтверждении количества определенного товара в корзине
(Нажатие на галочку в разделе "Корзина")
$(".apply_cart_quantity").click(function()
{
    var id = $(this).attr('data_id');
    var input = $('.cart_quantity_input-' + id);
    var quantity = input.val();

    $.post('/test_quantity_product_cart/' + id + '/' +
quantity, {}, function(data)
    {
        response = JSON.parse(data);

        if (!response.error.isEmpty())
        {
            alert(response.error);
            input.val(response.new_quantity);
            input.select();
        }
        if(response.num_products != "()") $("#span-
count").html(response.num_products);
        else $("#span-count").html("");
        $(".cart_price-" + id).html(response.new_price);
        $(".cart_total_sum").html(response.new_total_price);
    });
});
//Удаление товара при нажатии на кнопку "Удалить товар"
(Каталог, режим администратора)
$(".del-product").click(function()
{
    var id = $(this).attr('data_id');
    $.post('/del_product/' + id, {}, function(data)

```

```

        {
            if (data == 1)
            {
                $('#product_' + id).remove();
            }
        });
    });

    //При нажатии на кнопку "Оформить заказ" в разделе "Корзина"
    $(".checkout").click(function()
    {
        $("#success").html("");
        $("#error").html("");
        $.post('/checkout/', {}, function(data)
        {
            if (data == 1)
            {
                $("#span-count").html("");
                $(".cart_total_sum").html("₽ 0.00");
                $(".table-condensed").remove();
                $("#success").html("Заказ успешно оформлен!");
            }
            else
            {
                $("#error").html(data);
            }
        });
    });

    //При нажатии на кнопку "Добавить категорию" в разделе
    "Добавление категории" (Режим администратора)
    $('section').on('click', '.btn-add-category', function()
    {
        $("#success").html("");
        $("#errors").html("");

        if ($("#name_category").val().isEmpty())
        {
            $("#errors").html("Необходимо указать название
категории");
        }

        else
        {
            var params =
            {
                category_name: $("#name_category").val(),
                is_visible: $("#is_visible").prop('checked')
            };
            $.post("/add_category", params, function(data)
            {
                $('.categories').html(data);
                $("#success").html("Успешно");
            }

```

```

        });
    }
});

//При нажатии на кнопку "Сохранить изменения" в разделе
"Редактирование категории" (Режим администратора)
$("#btn-save-edit-category").click(function()
{
    $("#success").html("");
    $("#errors").html("");

    if ($("#name_category").val().isEmpty())
    {
        $("#errors").html("Необходимо указать название
категории");
    }
    else
    {
        var params =
        {
            category_id: $(this).attr('data_id'),
            category_name: $("#name_category").val(),
            is_visible: $("#is_visible").prop('checked')
        };

        $.post("/edit_category", params, function(data)
        {
            $('.categories').html(data);
            $("#success").html("Успешно");
        });
    }
});

//При нажатии на кнопку "Сохранить изменения" в разделе
"Редактирование категории" (Режим администратора)
$('section').on('click', '.del-category', function()
{
    var id = $(this).attr('data_id');
    $.post("/del_category/" + id, {}, function(data)
    {
        if (data == '1')
            $('#category_' + id).remove();
        else alert(data);
    });
});

//При нажатии на кнопку загрузки фото
var button = $("#btnUpload"), file;
new AjaxUpload(button,
{
    action: "/ajax/load_photo",
    data: {file: file},

```

```

        name: "userfile",
        onSubmit: function(file, ext)
        { //расширение файла
            if(!(ext &&/^(jpg|png|Jpeg|gif|)$/i.test(ext)))
        {
            alert("Ошибка допустимые разрешение! jpg, png, jpeg,
gif");
return false;
        }
        button.val("Загрузка");
        this.disable();
    },
    onComplete: function(file, response)
    { //2 параметра - имя файла и результат от сервера
response = JSON.parse(response);
        $("#imgUpload").attr('src',
'/images/product_photos/' + response.file);
        this.enable();
        button.val(response.answer);
    }
    });
});

```

Файл«CartModel.php»

```

<?
class CartModel
{
//Возвращает количество товаров в корзине пользователя
public static function get_number_products($user_id)
{
    global $db;
    $number_products = $db -> query("SELECT
SUM(Количество) FROM ShoppingCart WHERE
ID_покупателя='$user_id'");
    return $number_products -> fetch_row()[0];
}

//Возвращает количество определенного товара в корзине
авторизованного пользователя
public static function get_product_quantity($product_id)
{
    global $db;
    $user_id = $_SESSION['user'];
    $quantity = $db -> query("SELECT Количество FROM
ShoppingCart WHERE ID_покупателя='$user_id' AND
ID_товара='$product_id'");
    if ($quantity -> num_rows)
        return $quantity -> fetch_row()[0];
else return 0;
}
}

```

```

        //Возвращает доступное количество товара (Разность между
общим числом товаров на складе и уже находящимся в корзине)
        //Уменьшает количество товара в корзине, если на складе
его осталось меньше, чем в корзине
public static function available_quantity($product_id)
{
    global $db;
    $quantity_in_stock = $db -> query("SELECT На_складе
FROM Products WHERE ID_товара='$product_id'") -> fetch_row()[0];
    // Количество товара на складе
    $quantity_in_cart =
CartModel::get_product_quantity($product_id); //
Количество товара в корзине;
    $available_quantity = $quantity_in_stock -
$quantity_in_cart;
    if($available_quantity < 0)
    {
        $user_id = $_SESSION['user'];
        $db -> query("UPDATE ShoppingCart SET
Количество=Количество+'$available_quantity' WHERE
ID_покупателя='$user_id' AND ID_товара='$product_id'");
        return 0;
    }
    else return $available_quantity;
}
}
?>

```

Файл «ProductModul.php»

```

<?
class ProductModel
{
    //Определяет, является ли товар новым (Перестает быть им по
истечению 14 дней со дня добавления на сайте)
    public static function is_new($id_product)
    {
        global $db;
        return $db -> query("SELECT COUNT(*) FROM Products
WHERE ID_товара='$id_product' AND (Дата_добавления + 14 >
CURRENT_DATE())") -> fetch_row()[0];
    }

    public static function MoneyFormat($price)
    {
        return number_format($price, 2, '.', ' ');
    }
}
?>

```

Файл«UserModel.php»

```
<?
class UserModel
{
    //Проверка введенного имени пользователя на корректность (Длина
строки дб не менее двух символов)
    public static function checkName($name)
    {
        return strlen($name) >= 2;
    }

    //Проверка введенного пароля на корректность (Длина пароля
дб не менее двух символов)
    public static function checkPassword($password)
    {
        return strlen($password) >= 4;
    }

    //Проверка введенного e-mail на корректность (Используется
встроенная функция, использующая регулярные выражения)
    public static function checkEmail($email)
    {
        return filter_var($email, FILTER_VALIDATE_EMAIL);
    }

    //Возвращает true, если введенный логин еще не занят (Если
он отсутствует в базе данных), и false, если логин занят
    public static function loginIsFree($login)
    {
        global $db;
        $result = $db->query("SELECT COUNT(*) FROM Customers
WHERE Логин='$login'");
        return $result->fetch_row()[0] == 0;
    }

    //Возвращает true, если введенный логин $login еще не
занят, не принимая во учитывая $original_login.
    //Используется при редактировании данных пользователя,
сохраняя при этом исходный логин $original_login.
    public static function loginIsFreeMy($login, $original_login)
    {
        global $db;
        $result = $db->query("SELECT COUNT(*) FROM Customers
WHERE Логин='$login' AND Логин!='$original_login'");
        $exist = $result->fetch_row()[0] > 0;
        return !$exist;
    }

    //Проверяет номер телефона на корректность
    public static function checkPhone($phone)
    {
        return preg_match("/^[0-9]{11}+$/", $phone);
    }
}
```

					ПГУ1.090301.03.001ПЗ	Лист
						121
Изм.	Лист	№докум.	Подпись	Дата		

```

    }
    //Выполняет аутентификацию пользователя (Сохраняет
идентификатор пользователя в сессию)
    public static function auth($user_id)
    {
        $_SESSION['user'] = $user_id;
    }

    //Возвращает true, если пользователь является гостем (Не
прошел аутентификацию) и false в противном случае
    public static function is_guest()
    {
        return !isset($_SESSION['user']);
    }

    //Возвращает true, если пользователь является
администратором и false в противном случае
    public static function is_admin()
    {
        return isset($_SESSION['user']) && $_SESSION['user']
== 0;
    }

    //Возвращает логин авторизованного пользователя
    public static function get_login_auth_user()
    {
        global $db;
        $id = $_SESSION['user'];
        return $db->query("SELECT Логин FROM Customers WHERE
ID_покупателя='$id'")->fetch_row()[0];
    }

    public static function get_fio_current_user()
    {
        global $db;
        $id = $_SESSION['user'];
        $user = $db->query("SELECT * FROM Customers WHERE
ID_покупателя='$id'")->fetch_array();
        return $user['Фамилия']. ' ' . $user['Имя']. ' ' .
$user['Отчество'];
    }

    //Получает информацию о текущем пользователе
    public static function get_auth_user()
    {
        global $db;
        $id = $_SESSION['user'];
        return $db->query("SELECT * FROM Customers WHERE
ID_покупателя='$id'")->fetch_array();
    }
}

?>

```

Файл«Account.php»

```
<? require_once(PATH . 'views/header.php'); ?>
<section>
<div class="container">
<div class="row pull-center signup-row">
<div class="signup-form center">
<h2>Мойаккаунт</h2>
<div class="col-sm-4">
<font color="Red"><ruby id="error_surname"></ruby></font>
<input type="text" name="surname" id="user_surname"
placeholder="Фамилия" value="<?= $user['Фамилия']; ?>"
title="Фамилия"/>

<font color="Red"><ruby id="error_email"></ruby></font>
<input type="email" name="email" id="user_email" placeholder="Е-
mail" value="<?= $user['Почта']; ?>" title="E-mail"/>

<input type="text" name="address" id="user_address"
placeholder="Адрес" value="<?= $user['Адрес']; ?>" title="Адрес"/>
</div>
<div class="col-sm-4">
<font color="Red"><ruby id="error_name"></ruby></font>
<input type="text" name="name" id="user_name" placeholder="Имя"
value="<?= $user['Имя']; ?>" title="Имя"/>

<font color="Red"><ruby id="error_login"></ruby></font>
<input type="text" name="login" id="user_login" placeholder="Логин"
value="<?= $user['Логин']; ?>"title="Логин"/>
<font color="Red"><ruby id="error_phone"></ruby></font>
<input type="number" name="phone" id="user_phone"
placeholder="Телефон" value="<?= $user['Телефон']; ?>"
title="Телефон"/>
</div>
<div class="col-sm-4">
<font color="Red"><ruby id="error_patronymic"></ruby></font>
<input type="text" name="patronymic" id="user_patronymic"
placeholder="Отчество" value="<?= $user['Отчество']; ?>"
title="Отчество"/>
<font color="Red"><ruby id="error_password"></ruby></font>
<input type="text" name="password"
```

					ПГУ1.090301.03.001ПЗ	Лист
						123
Изм.	Лист	Нодокум.	Подпись	Дата		

```

        id="user_password" placeholder="Пароль" value="<?=$user['Пароль']; ?>" title="Пароль"/>

        <input type="text" name="text" id="user_date"
        title="Дата регистрации" value="<?=$date_create($user['Дата_регистрации'])->format('d.m.Y'); ?>"
        readonly/>
    </div>
    <div class="col-sm-8 pull-center col-sm-offset-2 center">
    <br>
    <font color="Cyan"><ruby id="success"></ruby></font>
    <input type="button" class="btn btn-default btn-edit-acc"
    value="Сохранить изменения"/>
    </div>
    </div>
    </div>
    </div>
</section>

<? require_once(PATH . 'views/footer.php'); ?>

```

Файл «Cart.php»

```

<? include(PATH . 'views/header.php'); ?>
<section id="cart_items">
    <div class="container">
    <div class="table-responsive cart_info">
    <h2 class="cart_form">Корзина</h2>
    <table class="table table-condensed">
    <thead>
    <tr class="cart_menu">
    <td class="image">Фото</td>
    <td class="description">Описание товара</td>
    <td class="price">Цена</td>
    <td class="discount">Скидка</td>
    <td class="price_discount">С скидкой</td>
    <td class="quantity">Количество</td>
    <td class="total">Итого</td>
    <td class="delete"></td>
    <td></td>
    </tr>
    </thead>
    <tbody>
    <? $sum = 0; ?>
    <? while($cart_product = $cart_products -> fetch_array()): ?>
    <tr class="cart-product-<?=$

```

```

    <a href="/catalog/<?= $cart_product['ID_категории']; ?>/product-
    <?= $cart_product['ID_товара']; ?>"></a>
  </td>
  <td class="cart_description">
    <a href="/catalog/<?= $cart_product['ID_категории']; ?>/product-
    <?= $cart_product['ID_товара']; ?>"><?= $cart_product['Название'];
    ?></a>
    <p>Кодтовара: <?= $cart_product['ID_товара']; ?></p>
  </td>
  <td><p>₽ <?= ProductModel::MoneyFormat($cart_product['Цена']);
  ?></p></td>
  <td><p><?= $cart_product['Скидка']; ?> %</p></td>
  <td><p>₽ <?=
  ProductModel::MoneyFormat($cart_product['Сумма_со_скидкой']);
  ?></p></td>
  <td>
    <input class="cart_quantity_input-<?= $cart_product['ID_корзины'];
    ?>" type="number" value="<?= $cart_product['Количество']; ?>"
    min="0" max="<?= $cart_product['На_складе']; ?>" step="1">
    <button class="apply_cart_quantity" data_id="<?=
    $cart_product['ID_корзины']; ?>"><i class="fa fa-
    check"></i></button>
    <p>На складе: <?= $cart_product['На_складе']; ?></p>
  </td>
  <td><p class="cart_total_price cart_price-<?=
  $cart_product['ID_корзины']; ?>">₽ <?=
  ProductModel::MoneyFormat($cart_product['Итоговая_сумма']);
  ?></p></td>
  <td class="cart_delete"><button data_id="<?=
  $cart_product['ID_корзины']; ?>" class="cart_quantity_delete"><i
  class="fa fa-times"></i></button></td>
</tr>
<? $sum += $cart_product['Итоговая_сумма']; ?>
<? endwhile; ?>
<tr><td></td><td></td><td></td><td></td><td></td>
<td><p>Всего:</p></td>
<td><p class="cart_total_price cart_total_sum">₽ <?=
ProductModel::MoneyFormat($sum); ?></p></td>
<td></td>
</tr>

```

```

</tbody>
</table>
<div class="signup-form pull-center">
<table class="long-center">
<tr><td><font color="Cyan"><ruby
id="success"></ruby></font></td></tr>
<tr><td><font color="Red"><ruby
id="error"></ruby></font></td></tr>
</table>
<input type="button" class="btn btn-default
checkout" value="Оформить заказ"/>
</div>
</div>
</div>
</section>
<? include(PATH . 'views/footer.php'); ?>

```

Файл «Catalog.php»

```

<? include(PATH . 'views/header.php'); ?>

<section>
<div class="container">
<div class="row">
<? include(PATH . 'views/categories.php'); ?>
<div class="col-sm-9 padding-right">
<div class="features_items">
<h2 class="title text-center"><?= ($products -> num_rows == 0) ?
'Раздел пуст' : 'Последние товары' ?></h2>
<? while($product = $products -> fetch_array()): ?>
<div class="col-sm-4 pull-center product_<?=
$product['ID_товара']; ?>">
<div class="product-image-wrapper">
<div class="single-products">
<div class="productinfo text-center"><? if
(ProductModel::is_new($product['ID_товара'])): ?><?
endif; ?><? if ($product['IsRecommended']): ?><? endif; ?>
<a href="/catalog/<?=$product['ID_категории']; ?>/product-
<?=$product['ID_товара']; ?>">" alt=""
class="border_img"/></a>
<h2 title="Цена">Р <?= $product['Цена']; ?></h2>
<p title="Название

```

```

        товара"><?= $product['Название']; ?></p>
        <p title="Производитель"><?= $product['Производитель']; ?></p>
        <? if (UserModel::is_admin()): ?><a
href="/form_edit_product/<?=$product['ID_товара'];?>"
title="Редактироватьтовар" class="btn btn-default edit-product
admin-elem"><i class="fa fa-edit"></i></a>
        <? endif; ?>
        <button data_id="<?=$product['ID_товара'];?>"
title="Добавитьтоварвкорзину" class="btn btn-default add-to-
cart"><i class="fa fa-shopping-cart"></i>Вкорзину</button>
        <? if (UserModel::is_admin()): ?><button
data_id="<?=$product['ID_товара'];?>" title="Удалитьтовар"
class="btn btn-default del-product admin-elem"><i class="fa fa-
trash-o"></i></button>
        <? endif; ?>
    </div>
</div>
</div>
</div>
<? endwhile; ?>
</div>
<?= $pagination -> get(); ?>
</div>
</div>
</div>
</section>
<? include(PATH . 'views/footer.php'); ?>

```

Файл«Footer.php»

```

<br><br><br>
<footer id="footer">
<div class="footer-bottom">
<div class="container">
<div class="row">
<p class="pull-left">Copyright © 2022</p>
<p class="pull-right">Интернет-магазин</p>
</div>
</div>
</div>
</footer>
<script src="/js/jquery.js"></script>
<script src="/js/bootstrap.min.js"></script>
<script src="/js/ajaxupload.js"></script>
<script src="/js/main.js"></script>
</body>
</html>

```

Файл «Categories.php»

```

<div class="col-sm-3">
<div class="left-sidebar">
<h2>Категории товаров</h2>
<div class="panel-group category-products">
<? if(UserModel::is_admin()): ?>
<div class="panel panel-default">
<div class="panel-heading">
<h4 class="panel-title">
<i class="fa fa-plus-circle"></i><a href="/catalog/add" class="<?=
($action == 'add_category')? 'active': ''; ?>"> Добавить
категорию</a>
<br><br>
</h4>
</div>
</div>
<? endif; ?>

<? while($category = $categories -> fetch_array()): ?>
<div class="panel panel-default category_<?=
$category['ID_категории']; ?>">
<div class="panel-heading">
<h4 class="panel-title">
<? if(UserModel::is_admin()): ?>
<i class="<?= ($category['IsVisible']) ? 'fa fa-eye' : 'fa fa-eye-
slash'; ?>" title="<?= ($category['IsVisible']) ? 'Видна
покупателям' : 'Не видна покупателям'; ?>"></i><? endif; ?>
<a href="/catalog/<?= $category['ID_категории']; ?>/page-1"
class="<?= ($id_category == $category['ID_категории'])? 'active':
'; ?>"><?= $category['Название']; ?></a>
<? if(UserModel::is_admin()): ?>
<div class="col-sm-3 pull-right admin-elements"><a
href="/catalog/edit/<?= $category['ID_категории']; ?>"
title="Редактировать категорию"><i class="fa fa-edit"></i></a><a
data_id="<?= $category['ID_категории']; ?>" class="del-category"
title="Удалить категорию"><i class="fa fa-trash-o"></i></a>
</div>
<? endif; ?>
</h4>
</div>
</div>
<? endwhile; ?>
</div>
</div>
</div>

```

Файл«*Form_add_category.php*»

```

<? include(PATH . 'views/header.php'); ?>
<section>
<div class="container">
<div class="row">
<div class="categories">
<? include(PATH . 'views/categories.php'); ?>
</div>
<div class="col-sm-5 col-sm-offset-2">
<div class="features_items">
<? if(UserModel::is_admin()): ?>
<h2 class="title text-center">Добавление категории</h2>
<div class="signup-form">
<div class="form-category">
<font color="Red"><ruby id="errors"></ruby></font>
<table><tr><td><font color="Cyan"><ruby
id="success"></ruby></font></td></tr></table>
<input type="text" id="name_category" placeholder="Название
категории" title="Название категории"/>
<input type="checkbox" class="checkbox" id="is_visible"
checked><label> Видна покупателям</label>

<br><br>
<input type="button" class="btn btn-default btn-add-category"
title="Добавить категорию" value="Добавить категорию"/>
</div>
</div>
<? else: ?>
<h2 class="title text-center">Нет доступа</h2>
<? endif; ?>
</div>
</div>
</div>
</div>
</section>

<? include(PATH . 'views/footer.php'); ?>

```

Файл«Form_add_product.php»

```
<? include(PATH . 'views/header.php'); ?>

<section>
<div class="container">
<div class="signup-form">
<div class="row signup-row">
<? if(UserModel::is_admin()): ?>
<h2>Добавление товара</h2>
<div class="col-sm-3 pull-center not-hover">

<input type="button" id="btnUpload" class="btn btn-default"
value="Загрузить фото" />
<table><tr>
<td style="text-align:right"><input type="checkbox"
class="checkbox" id="is_recommended"
checked><label>Рекомендовать</label></td>
</tr></table>
</div>

<div class="col-sm-3 pull-center">
<input type="text" id="product_name" placeholder="Название товара"
title="Название товара"/>
<input type="text" id="product_manufacturer"
placeholder="Производитель товара" title="Производитель товара">
<select id="product_category" title="Категория товара">
<option value="-1">Категория товара</option>
<? while($category = $categories -> fetch_array()): ?>
<option value='<?= $category['ID_категории']; ?>'><?=
$category['Название']; ?></option>
<? endwhile; ?>
</select>
</div>

<div class="col-sm-3 pull-center">
<input type="number" id="product_price" min="0" step="1"
placeholder="Цена товара" title="Цена товара (₽)">
<input type="number" id="quantity_products" min="0" step="1"
placeholder="Количество товаров" title="Количество товаров на
складе (шт)">
<input type="number" id="product_discount" min="0" max="100"
step="1" placeholder="Скидка" title="Скидка (%)">
</div>
```

```

<div class="col-sm-6 pull-center">
  <textarea id="description" placeholder="Описание товара"
title="Описание товара"></textarea>
</div>

<div class="col-sm-4 pull-center col-sm-offset-1 center">
  <font color="Red"><ruby id="errors"></ruby></font>
  <font color="Green"><ruby id="success"></ruby></font>
  <input type="button" class="btn btn-default btn-add-product"
value="Добавить товар"/>
</div>
<? else: ?>
<h2>Нет доступа</h2>
<? endif; ?>
</div>
</div>
</div>
</section>

<? include(PATH .
'views/footer.php'); ?>

```

Файл «Form_edit_category.php»

```

<? include(PATH . 'views/header.php'); ?>
<section>
<div class="container">
<div class="row">
<div class="categories">
<? include(PATH . 'views/categories.php'); ?>
</div>
<div class="col-sm-5 col-sm-offset-2">
<div class="features_items">
<? if(UserModel::is_admin()): ?>
<h2 class="title text-center">Редактирование категории</h2>
<div class="signup-form">
<div class="form-category">
<font color="Red"><ruby id="errors"></ruby></font>
<table><tr><td><font color="Cyan"><ruby
id="success"></ruby></font></td></tr></table>
  <input type="text" id="name_category" placeholder="Название
категории" title="Название категории" value="<?=
$Category['Название']; ?>"/>
  <input type="checkbox" class="checkbox" id="is_visible"<?
if($Category['IsVisible']) echo 'checked'; ?>><label> Видна
покупателям</label>

```



```

$product['Производитель']; ?>" />
<select id="product_category" title="Категория товара">
<option value="-1">Категория товара</option>
<? while($category = $categories -> fetch_array()): ?>
<option value='<?= $category['ID_категории']; ?>' <?
if($category['ID_категории'] == $product['ID_категории']) echo
'selected'; ?>><?= $category['Название']; ?></option>
<? endwhile; ?>
</select>
</div>
<div class="col-sm-3 pull-center">
<input type="number" id="product_price" min="0" step="1"
placeholder="Цена товара" title="Цена товара (₽)" value="<?=
$product['Цена']; ?>"/>
<input type="number" id="quantity_products" min="0" step="1"
placeholder="Количество товаров" title="Количество товаров на
складе (шт)" value="<?= $product['На_складе']; ?>"/>
<input type="number" id="product_discount" min="0" max="100"
step="1" placeholder="Скидка" title="Скидка (%)" value="<?=
$product['Скидка']; ?>"/>
</div>
<div class="col-sm-6 pull-center">
<textarea id="description" placeholder="Описание товара"
title="Описание товара"><?= $product['Описание']; ?></textarea>
</div><div class="col-sm-4 pull-center col-sm-offset-1 center">
<font color="Red"><ruby id="errors"></ruby></font><font
color="Cyan"><ruby id="success"></ruby></font>
<input type="button" data_id="<?= $product['ID_товара']; ?>"
class="btn btn-default btn-save-edit-product" value="Сохранить
изменения"/>
</div>
<? else: ?>
<h2>Товар не найден</h2>
<? endif; ?>
<? else: ?>
<h2>Нет доступа</h2>
<? endif; ?>
</div>
</div>
</div>
</section>
<? include(PATH . 'views/footer.php'); ?>

```

Файл«Header.php»

```
<? require_once(PATH . 'controllers/cart.php'); ?>
<!DOCTYPE html>
<html lang='ru'>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<title>Главная</title>
<link href="/css/bootstrap.min.css" rel="stylesheet">
<link href="/css/font-awesome.min.css" rel="stylesheet">
<link href="/css/main.css" rel="stylesheet">
</head>
<body>
<header id="header"><!--header-->
<div class="header_top"><!--header_top-->
<div class="container">
<div class="row">
<div class="col-sm-6">
<div class="contactinfo">
<ul class="nav nav-pills">
<li><p><i class="fa fa-phone"></i> +7 900 111 22 33</p></li>
<li><p><i class="fa fa-envelope"></i> our_email@mail.ru</p></li>
</ul>
</div>
</div>
<div class="col-sm-6">
<div class="contactinfo pull-right">
<? if(UserModel::is_guest()): ?>
<p>Интернет-магазин</p>
<? else: ?>
<p><i class="fa fa-user"></i><?= UserModel::get_fio_current_user();
?></p>
<? endif; ?>
</div>
</div>
</div>
</div>
</div><!--/header_top-->
<div class="header-middle"><!--header-middle-->
<div class="container">
<div class="row">
<div class="col-sm-4">
<div class="logo pull-left">
<a href="/catalog/1/page-1"></a>
</div>
</div>
```

```

<div class="shop-menu pull-right">
<ul class="nav navbar-nav">
<? if (UserModel::is_guest()): ?>
<li><a href="/login" class="<?= ($action == "login") ? "active" :
""; ?>"><i class="fa fa-lock"></i> Вход</a></li>
<li><a href="/registr" class="<?= ($action == "registr") ? "active"
: ""; ?>"><i class="fa fa-users"></i> Регистрация</a></li>

<? else: ?>
<li><a href="/catalog/1/page-1" class="<?= ($controller ==
"catalog") ? "active" : ""; ?>"><i class="fa fa-home"></i>
Катерории</a></li>
<li><a href="/acc" class="
<?= ($action == "account") ? "active" : ""; ?>"><i class="fa fa-
user"></i> Аккаунт</a></li>
<li><a href="/cart" class="<?= ($controller == "cart") ? "active" :
""; ?>"><i class="fa fa-shopping-cart"></i> Корзина
<span id="span-count">
<? $count = CartModel::get_number_products($_SESSION['user']); if
($count != 0) echo "($count)"; ?></span></a></li>
<li><a href="/logout"><i class="fa fa-unlock"></i> Выход</a></li>
<? endif; ?>
</ul>
<? if (UserModel::is_admin()): ?>
<br>

<ul class="nav navbar-nav">
<li><a href="/form_add_product" class="<?= ($action ==
"add_product") ? "active" : ""; ?>"><i class="fa fa-plus-
circle"></i> Добавить товар</a></li>

<li><a href="/list_orders" class="<?= ($action == "list_orders") ?
"active" : ""; ?>"><i class="fa fa-list-alt"></i> Список
заказов</a></li>
</ul>
<? endif; ?>
</div>
</div>
</div>
</div>
</div><!--/header-middle-->

</header><!--/header-->

```

Файл «List_orders.php»

```
<? include(PATH . 'views/header.php'); ?>

<section id="cart_items">
  <div class="container">
    <div class="table-responsive cart_info">
      <h2 class="cart_form">Список заказов</h2>
      <table class="table table-condensed">
        <thead>
          <tr class="cart_menu">
            <td>ФИО</td>
            <td>Адрес</td>
            <td>Телефон</td>
            <td>Почта</td>
            <td>Заказанные товары</td>
            <td>Дата и время</td>
            <td>Итого</td>
          </tr>
        </thead>
        <tbody>
          <? while($user_order = $users_orders -> fetch_array()): ?>
            <tr>
              <td class="uniform"><p><?= $user_order['Фамилия'] . ' ' .
                $user_order['Имя'] . ' ' . $user_order['Отчество']; ?></p></td>
              <td class="uniform"><p><?= $user_order['Адрес']; ?></p></td>
              <td class="uniform"><p><?= $user_order['Телефон']; ?></p></td>
              <td class="uniform wrap"><p><?= $user_order['Почта']; ?></p></td>
              <td>
                <table class="table-condensed uniform-5">
                  <? $products = json_decode($user_order['Товары'], true);
                     $sum = 0;
                     foreach($products as
$product_id => $quantity):
                      $product = $db ->
query("SELECT *, Цена * (1 - Скидка / 100) AS Цена_со_скидкой FROM
Products WHERE ID_товара='$product_id'") -> fetch_array();
                      $product_name =
$product['Название'];
                      $category_id =
$product['ID_категории'];
                      $product_manufacturer =
$product['Производитель'];
```

```

$price_discount =
ProductModel::MoneyFormat($product['Цена_со_скидкой'] * $quantity);
$sum +=
$product['Цена_со_скидкой'] * $quantity; ?>
<tr>
<td class='center'><a title="Код товара" href="/catalog/<?=
$category_id ;?>/product-<?= $product_id ;?>"><?= $product_id
;?></a></td>
<td><ptitle="Название товара"><?= $product_name ;?></p></td>
<td><ptitle="Производитель товара"><?= $product_manufacturer
;?></p></td>
<td class="center"><p class="green-color" title="Количество
товара"><?= 'x' . $quantity ;?></p></td>
<td><p class="money" title="Цена с учетом скидки"><?= 'Р ' .
$price_discount ;?></p></td>
</tr>
<? endforeach; ?>
</table>
</td>

<td class="uniform"><p class="green-color"><?=
date_create($user_order['Дата_время_заказа'])->format('d.m.y H:i');
?></p></td>
<td class="uniform"><p class="money"><?= 'Р ' .
ProductModel::MoneyFormat($sum); ?></p></td>
</tr>
<? endwhile; ?>
</tbody>
</table>
</div>
</div>
</section>
<? include(PATH . 'views/footer.php'); ?>

```

Файл«Login.php»

```

<? require_once(PATH . 'views/header.php'); ?>

<section>
<div class="container">
<div class="row pull-center">
<div class="col-sm-4 col-sm-offset-4 signup-col">
<div class="signup-form center">
<h2>Вход на сайт</h2>
<font color="Red"><ruby id="error_login"></ruby></font>
<input type="login" id="user_login"

```

```
placeholder="Логин" title="Логин"/>
<font color="Red"><ruby id="error_password"></ruby></font>
<input type="password" id="user_password" placeholder="Пароль"
title="Пароль"/>
```

```
<br>
<input type="button" class="btn btn-default btn-login"
value="Вход"/>
</div>
</div>
</div>
</div>
</section>
```

```
<? require_once(PATH . 'views/footer.php'); ?>
```

Файл«*Product.php*»

```
<? include(PATH . 'views/header.php'); ?>
<section>
<div class="container">
<div class="row">
<? include('views/categories.php'); ?>
<div class="col-sm-9 padding-right">
<div class="product-details">
<div class="row">
<h2 class="title text-center">Информация о товаре</h2>
<div class="col-sm-4">
<div class="view-product">
" class="border_img"/>
</div>
</div>
<div class="col-sm-7">
<div class="product-information">
<? if (ProductModel::is_new($product['ID_товара'])): ?><?
endif; ?>
<? if ($product['IsRecommended']): ?><? endif; ?>
<h2><?= $product['Название']; ?></h2>
<span>
<span>Р <?= $product['Цена']; ?></span>
<label>Количество:</label>
```

```

<input type="number" id="quantity-to-cart" value="1"/><button
data_id="<?=$product['ID_товара'];?>" class="btn btn-default add-
to-cart-quantity cart">
<i class="fa fa-shopping-cart"></i> В корзину </button>
</span>

<table class="left">
<tr>
<td><p><b>Код товара:</b></p></td>
<td><p><?=$product['ID_товара']; ?></p></td>
</tr>
<tr>
<td><p><b>Производитель:</b></p></td>
<td><p><?=$product['Производитель']; ?></p></td>
</tr>
<tr>
<td><p><b>Категория:</b></p></td>
<td><p><? $category = $product['ID_категории']; echo $db ->
query("SELECT Название FROM Categories WHERE
ID_категории='$category'") -> fetch_row()[0]; ?></p></td>
</tr>
<tr>
<td><p><b>Уже в корзине:</b></p></td>
<td><p id="num-in-cart"><?=$
CartModel::get_product_quantity($product['ID_товара']); ?></p></td>
</tr>
<tr>
<td><p><b>Количество на складе:</b></p></td>
<td><p><?=$product['На_складе']; ?></p></td>
</tr>
<tr>
<td><p><b>Дата добавления:</b></p></td>
<td><p><?=$date("d.m.Y", strtotime($product['Дата_добавления']));
?></p></td>
</tr>
<tr>
<td colspan="2"><p><? if($product['Скидка'] != 0): ?>

```



```

<input type="text" name="name" id="user_name" placeholder="Имя"
title="Имя"/>

<font color="Red"><ruby id="error_password"></ruby></font>
<input type="password" name="password" id="user_password"
placeholder="Пароль" title="Пароль"/>

<font color="Red"><ruby id="error_email"></ruby></font>
<input type="email" name="email" id="user_email" placeholder="E-
mail" title="E-mail"/>
</div>
<div class="col-sm-6 pull-center col-sm-offset-3 center">
<br>
<font color="Cyan"><ruby id="success"></ruby></font>
<input type="button" class="btn btn-default btn-registr"
value="Регистрация"/>
</div>
</div>
</div>
</div>
</section>

<? require_once(PATH . 'views/footer.php'); ?>

```

ПриложениеВ

(Обязательное)

Web-портал интернет-магазина

Презентация

					ПГУ1.090301.03.001ПЗ	Лист
						142
Изм.	Лист	№докум.	Подпись	Дата		

Тема работы: «Web-портал интернет-магазина»

Автор работы: студент гр. 18ВВ1 Арсентьева У.С.
Научный руководитель: к.т.н., доцент Дубравин А.В.

Темаработы

Введение

- Сегодня происходит глобальное внедрение интернет-технологий во все сферы жизни человека, что, в свою очередь, формирует глобальную цифровую экономику;
- Важной задачей развития становится переход экономики в состояние, в котором большое внимание уделяется информатизации, массовому распространению интернет-технологий;
- В данных условиях интернет-магазины являются платформами для обслуживания экономики в целом, и в частности, цифровой ее части.

Введение

Изм.	Лист	№докум.	Подпись	Дата

ПГУ1.090301.03.001ПЗ

Лист

143

Цель и задачи

Цель работы - разработать масштабируемую, отказоустойчивую систему продажи товаров на базе паттерна модель, вид, контроллер.

Объектом разработки является программное приложение, которое обеспечивает клиентам доступ к каталогу товаров и услуг, администраторам доступ к настройкам системы.

Задачи:

- провести анализ предметной области;
- разработать и реализовать структуру базы данных;
- разработать программную часть для работы с базой данных;
- произвести отладку и тестирование разработанного приложения.

Цель и задачи

Функционал приложения

В процессе выполнения работы были решены следующие вопросы:

- Разработана СУБД для хранения каталога товаров и категорий продуктов;
- Разработаны структурированные запросы (SQL) и программный код (в рамках выбранного инструментария) для доступа к данным;
- Добавлены данные в каталог продуктов, которые определяют атрибуты продукта;
- Разработан для администратора личный раздел сайта.

Функционал приложения

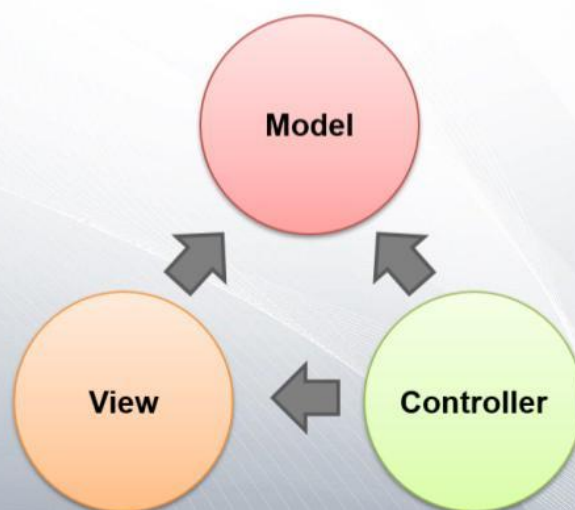
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ1.090301.03.001ПЗ

Лист

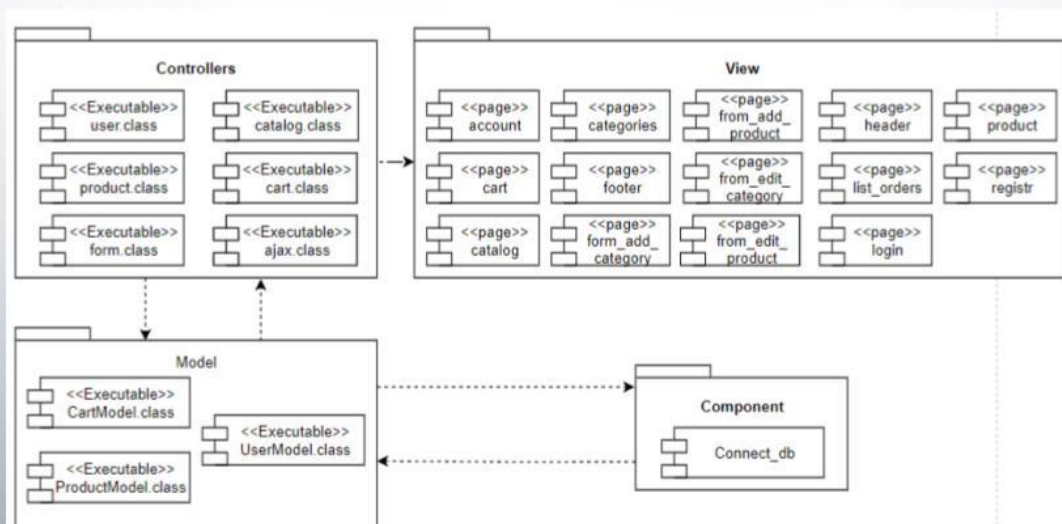
144

Архитектура приложения



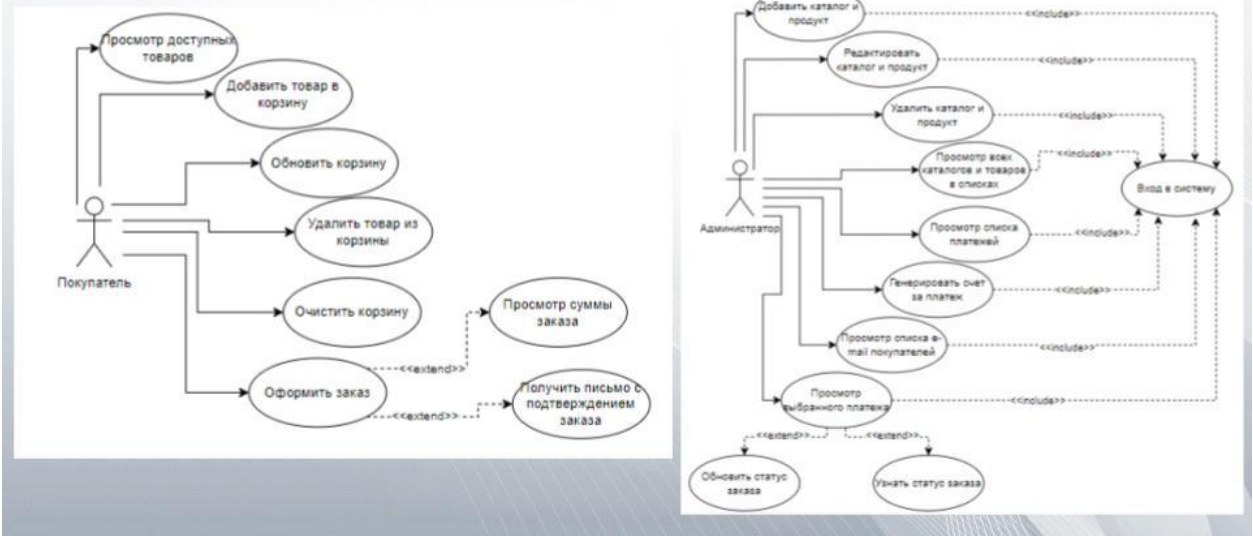
Архитектура приложения

Архитектура приложения



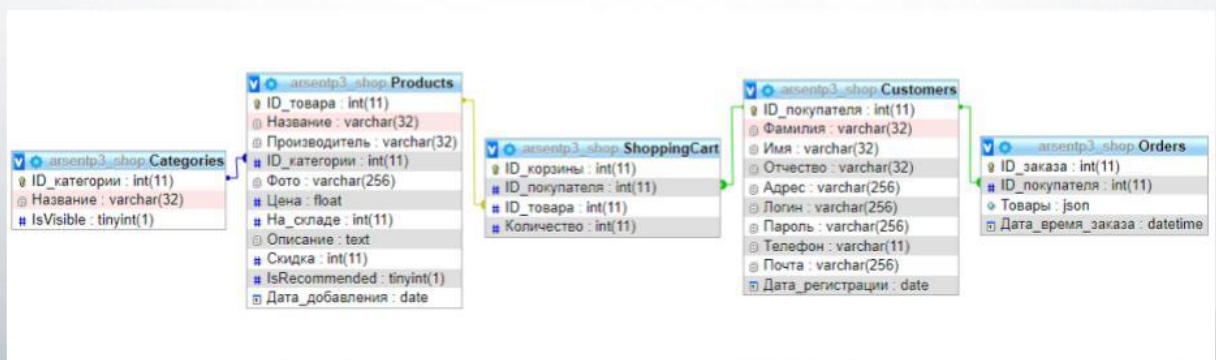
Архитектура приложения

Функционал пользователей



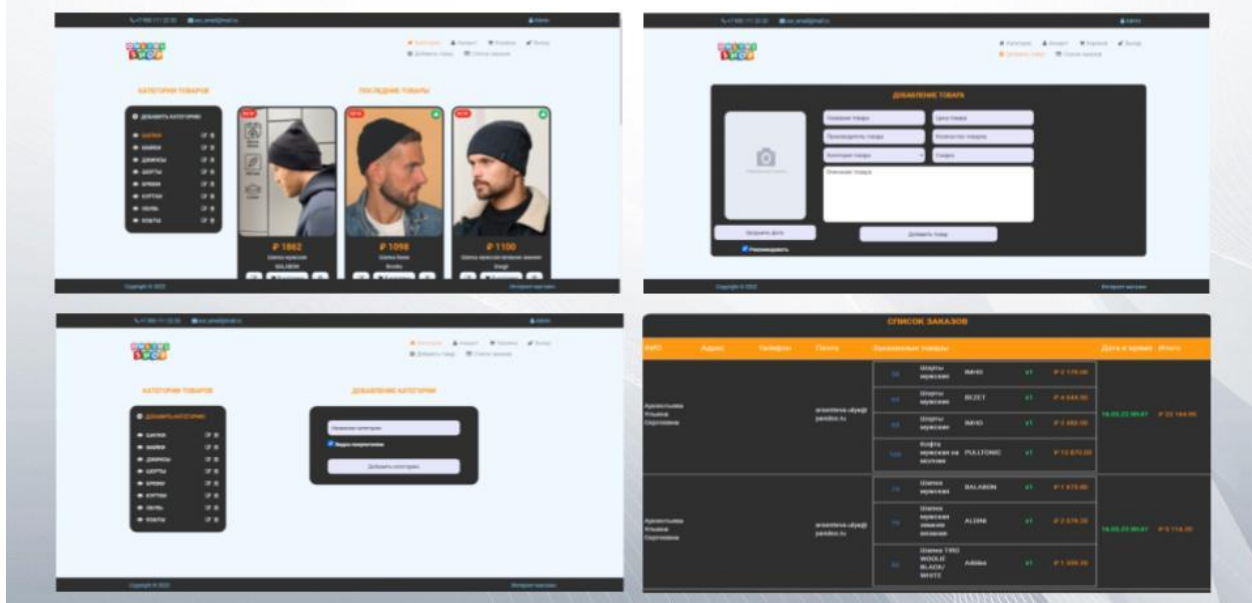
Функционал пользователей

Структура базы данных



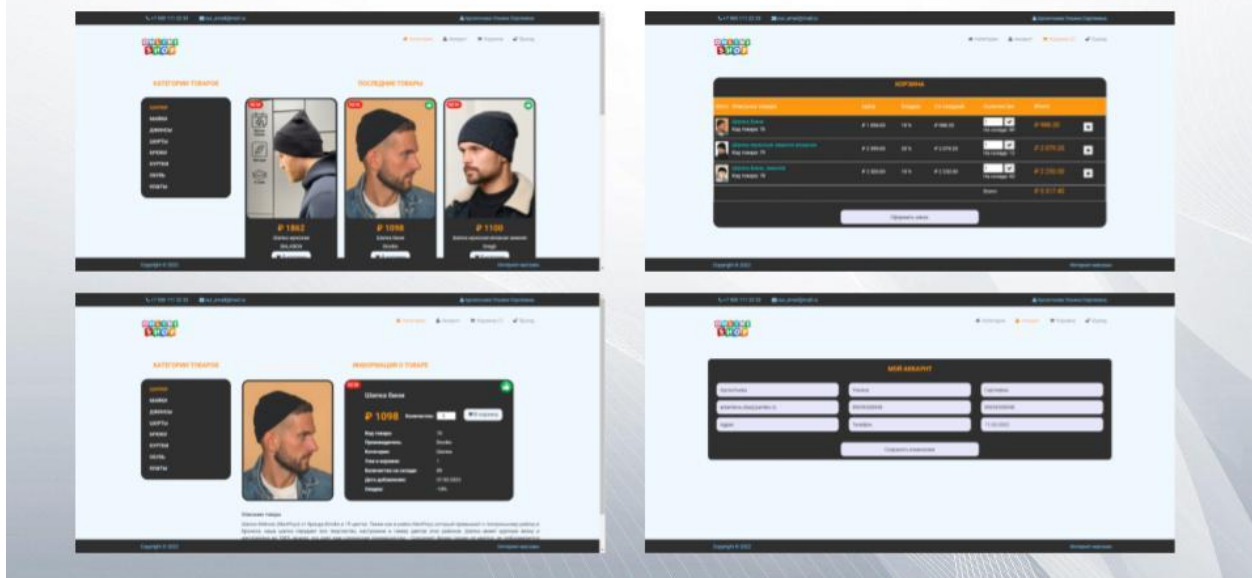
Структура базы данных

Интерфейс администратора



Интерфейс администратора

Интерфейс пользователя



Интерфейс пользователя

Заклучение

В результате проделанной работы реализован web-портал на базе классического паттерна Модель-Вид-Контроллер, что позволяет легко масштабировать приложение.

Задачи, поставленные на дипломный проект, выполнены.

А так же следует отметить, что выбран популярный инструментарий для разработки: PHP и MySQL.

Данный набор инструментов позволяет быстро собрать команду разработки, т.к. данные инструменты являются популярными среди разработчиков.

Заклучение

Спасибо за внимание!

Спасибозавнимание

					<i>ПГУ1.090301.03.001ПЗ</i>	Лист
Изм.	Лист	№докум.	Подпись	Дата		148