Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет инженерно–экономический

Кафедра экономической информатики

Дисциплина Средства и технологии анализа и разработки информационных систем

**ПОЯСНИТЕЛЬНАЯ** **ЗАПИСКА**

к курсовому проекту

на тему

**ОНЛАЙН МАГАЗИН**

Студент Пашкевич Б.В.

гр. 572301

Руководитель Хомяков П. В.

Минск 2018

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc513901445)

[1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 7](#_Toc513901446)

[1.1 Описание онлайн магазинов 7](#_Toc513901447)

[1.2 Принципы работы 7](#_Toc513901448)

[2 ОПИСАНИЕ ОСНОВНЫХ ПРОЦЕССОВ ПРЕДМЕТНОЙ ОБЛАСТИ 9](#_Toc513901449)

[3 СПЕЦИФИКАЦИЯ ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 13](#_Toc513901450)

[4 ИНФОРМАЦИОННАЯ МОДЕЛЬ СИСТЕМЫ И ЕЁ ОПИСАНИЕ 15](#_Toc513901451)

[5 ОБОСНОВАНИЕ ВЫБОРА КОМПОНЕНТОВ И ТЕХНОЛОГИЙ ДЛЯ РЕАЛИЗАЦИИ КУРСОВОГО ПРОЕКТА 18](#_Toc513901452)

[5.1 Symfony 18](#_Toc513901453)

[5.2 Doctrine 18](#_Toc513901454)

[5.3 jQuery 19](#_Toc513901455)

[5.4 Twig 19](#_Toc513901456)

[5.5 Composer 19](#_Toc513901457)

[6 МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ 20](#_Toc513901458)

[6.1 Диаграмма последовательности создания продукта 20](#_Toc513901459)

[6.2 Диаграмма состояний авторизации пользователя 20](#_Toc513901460)

[6.3 Диаграмма компонентов 21](#_Toc513901461)

[6.4 Диаграмма развертывания 21](#_Toc513901462)

[6.5 Диаграмма классов 21](#_Toc513901463)

[6.6 Блок схемы алгоритмов, реализующих бизнес-логику системы 21](#_Toc513901464)

[7 ОПИСАНИЕ ПРИМЕНЕНИЯ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ 23](#_Toc513901465)

[8 РУКОВОДСТВО ПО РАЗВЕРТЫВАНИЮ СИСТЕМЫ 24](#_Toc513901466)

[9 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ РАЗРАБОТАННОЙ СИСТЕМЫ И ОЦЕНКА ВЫПОЛНЕНИЯ ЗАДАЧ 25](#_Toc513901467)

[ЗАКЛЮЧЕНИЕ 45](#_Toc513901468)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 46](#_Toc513901469)

[ПРИЛОЖЕНИЕ А 47](#_Toc513901470)

[ПРИЛОЖЕНИЕ Б 51](#_Toc513901471)

[ПРИЛОЖЕНИЕ В 53](#_Toc513901472)

[ПРИЛОЖЕНИЕ Г 66](#_Toc513901473)

ВВЕДЕНИЕ

В текущее время интернет стремительно становиться более развитой средой для осуществления различного вида коммуникаций с потребителями. В тоже время, является существенным и тот факт, что интернет становиться достаточно удобной и дешевой «торговой площадкой». Все большее количество различных фирм старается выставить свою продукцию в on-line среде. При этом это представление не ограничивается лишь созданием промо-сайтов и размещением рекламных баннеров, статей на сайтах и информационных порталах. С ростом развития интернет-среды развивается и само предложение. Сейчас люди могут не только получать интересующую их информацию, но и совершать необходимые покупки. Так же с помощью интернет-магазинов можно приобретать товары из совершенно разных категорий, как простые потребительские товары, так и высокотехнологичные.

Такое положение вещей обусловлено непрерывным стабильным ростом пользователей глобальной сети.

Большую часть аудитории интернет сети составляют люди, работающие в офисах или проводящие большую часть своего времени за компьютером. Как правило, их образ жизни не может позволить им тратить большое количество времени на походы по обычным магазинам в поисках необходимых товаров. Возможность заказать интересующий их продукт в интернете является действительно хорошим выходом из сложившейся ситуации. Интернет существенно ограничил возможности представления некоторых товаров, поэтому в большинстве случаев подходит только для совершения повторной покупки, например приобретения парфюмерии.

Цель данного курсового проекта – существенно упростить покупку различных товаров через интернет путем улучшения и упрощения интерфейса при помощи создания программного продукта. Для достижения поставленной цели необходимо выполнить ряд задач:

* Детально изучить принцип работы электронных магазинов;
* определить список функций необходимых для эффективной работы программного продукта;
* разработать алгоритм работы программы;
* спроектировать базу данных хранимой информации;
* реализовать серверную часть приложения, которая будет реализовывать бизнес логику, и будет выполнять работу с базой данных;
* реализовать клиентскую часть приложения, с удобным и интуитивно понятным интерфейсом;
* протестировать полученное программное средство, и убедиться в его корректной работе.

1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

* 1. Описание онлайн магазинов

Начиная с 00-х годов во всем мире произошел резкий рост активности в области онлайновой торговли, можно сказать, что зародилась онлайн торговля. Вслед за крупнейшими компаниями, производящими дорогостоящее компьютерное оборудование в интернет стали выходить достаточно мелкие торговцы различными товарами. Появилось огромное количество специализированных книжных магазинов, магазины дисков и видеокассет, а так же магазины торгующие всякой мелочью. Сейчас практически любые товары можно купить не выходя из дома.

Стремительное развитие IT-технологий породили не менее стремительное развитие электронной коммерции. Один из самых популярных on-line бизнесов это интернет-магазин. Даже на фоне мирового финансового кризиса, по различным исследовательским данным, рост составил в среднем в 2014 году от 20 до 35 %. Так же количество покупок, совершаемых через Интернет, ежегодно возрастает примерно на 20 %. Маркетинговые исследования предсказывают, что к 2020 году количество покупок, которые будут совершаться через Интернет, вырастет более чем на 55 %. Таким образом, интернет-магазин является очень перспективным направлением бизнеса. Однако этот рынок отличается высокой степенью конкуренции и далеко не все проекты могут бороться и тем более выжить. Это происходит в следствии того, что зачастую предприниматели, которые начинают свой бизнес в интернете, не имеют четкого представления об этой сфере, в частности о интернет-рынке и о тех трудностях, с которыми они должны будут столкнутся в ближайшем будущем. Как показывают исследования за 2017 год, пока ещё не сформированы конкретные общепринятые методы и техника корректного управления интернет-магазином. Поэтому у каждого предпринимателя начинающего свой интернет-бизнес возникает вопрос: как правильно создать онлайн магазин, как корректно им управлять, с чего начинать и чем руководствоваться?

* 1. Принципы работы

Онлайн магазин функционально похож на обычный: это сайт, где каждый платежеспособный желающий в состоянии заказать себе товар. Онлайн магазин, как правило, разделен на несколько тематических разделов (категорий), согласно содержанию товаров.

Онлайн магазин предоставляет пользователям следующие виды услуг:

* просмотреть список товаров;
* отфильтровать по категории;
* найти товар;
* подробно рассмотреть товар;
* выбрать характеристики настраиваемого продукта;
* добавить в корзину;
* заказать товар.

Существуют специальные веб-сервисы, работающие с онлайн магазинами. Также существуют и специализированные онлайн магазины, администрация которых придерживается строго специализированных товаров по одной или нескольким отраслям промышленности, видам товаров или услуг.

Онлайн магазины можно разделить по видам продаж на:

1. B2B (би-ту-би или «бизнес для бизнеса») — продажа или оказание услуг иным коммерческим предприятиям.
2. B2C (би-ту-си или «бизнес для клиента») — продажа или оказание услуг конечному потребителю.

В данном проекте будет рассмотрен B2C магазин.

Существуют следующие виды товаров:

1. Обычный товар – объявления содержат исключительно информацию о приобретаемом товаре, то есть его характеристики, условия покупки и т.д.;
2. Настраиваемый товар – такие товары можно сказать включают в себя информацию о нескольких похожих товарах, а именно их общее описание и различия.

Самое большое количество просмотров приходится на первую страницу сайта, поэтому можно заработать еще и на рекламе. Услуга за размещение рекламы может быть платная, но результативная.

В списках товаров используется их краткое описание для максимального информирования потребителя и экономии места.

А составляя название, необходимо писать его кратко, что бы при запросе поисковым системам было легко найти продукт.

Автоматизация помогает, как и сказано выше, сэкономить время и деньги, так как весь процесс производится на одном месте без каких-либо лишних вложений и действий.

2 ОПИСАНИЕ ОСНОВНЫХ ПРОЦЕССОВ ПРЕДМЕТНОЙ ОБЛАСТИ

Взаимодействие системы с окружающей средой описывается в терминах входа («Данные о товарах»), выхода (основной результат процесса – «Обновленный каталог» и «Заказанный товар»), управления («Правила магазина» и «Нормативные документы») и механизмов («ПО», «Пользователь», «Администратор» – это ресурсы, необходимые для процесса заказа товара).

«Пользователи» – посетители сайта. Они просматривают каталог товаров. Делают заказы.

«Налоговый кодекс» и «Закон о защите прав потребителей» - это правила, которыми управляется процесс функционирования приложения.

В оказании услуг принимает участие «ПО» и «Администратор». Чтобы оформить заказ и получить прибыль.

Основным процессом данного курсового проекта является заказать товар. В соответствии с этим на рисунке 2.1 представлена контекстная диаграмма функциональной модели.

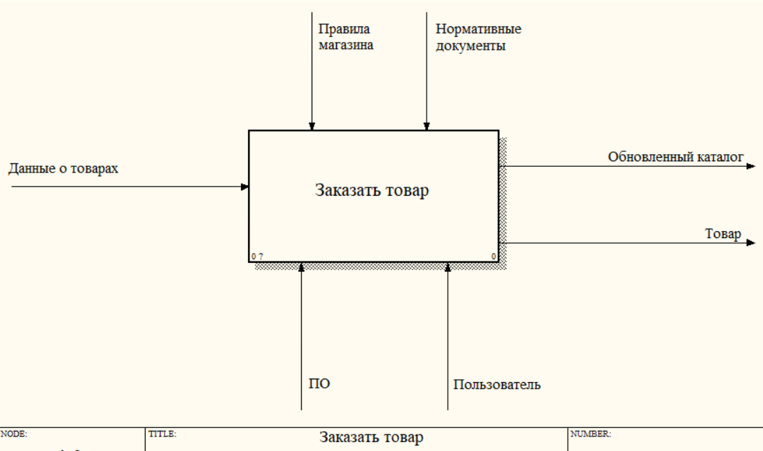


Рисунок 2.1 – Контекстная диаграмма верхнего уровня

В соответствии с рисунком 2.2 представлена диаграмма декомпозиции IDEF0, описывает разделение процесса заказа товара на следующие этапы:

- Завести учетную запись в магазине и авторизоваться;

- Выбрать товар;

- Произвести оплату;

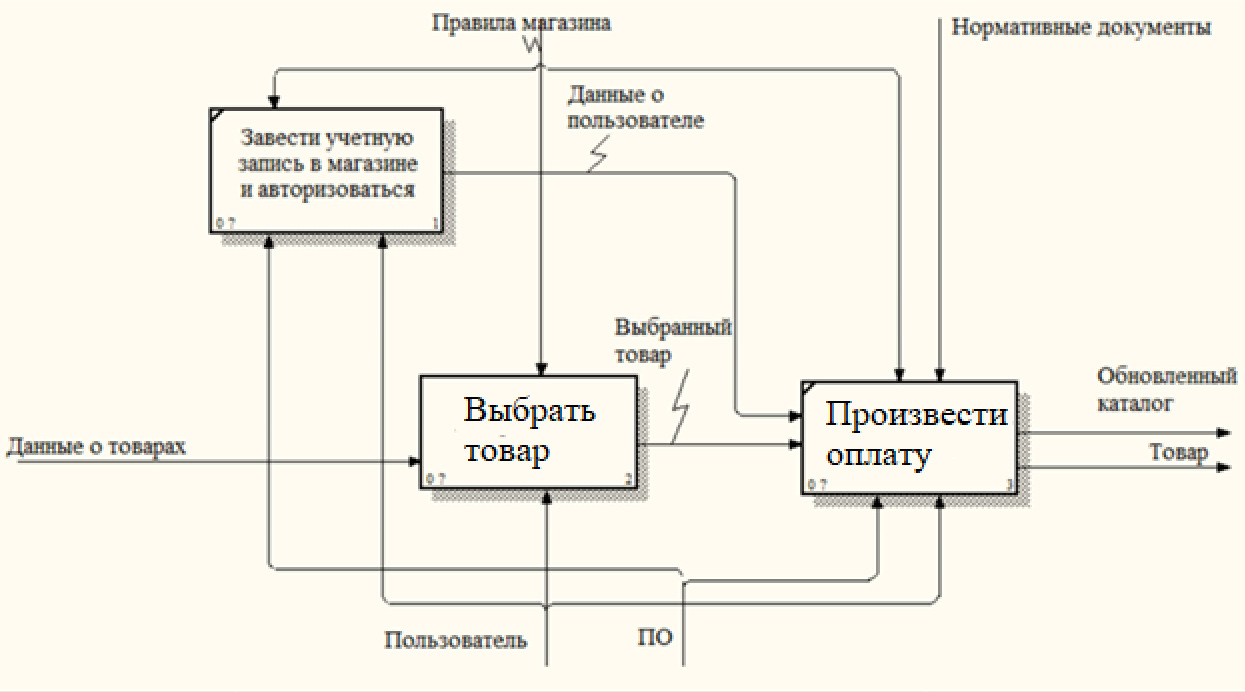


Рисунок 2.2 – Декомпозиция контекстной диаграммы

Весь процесс “Заказать товар” разбивается на:

1)           Завести учетную запись в магазине и авторизоваться – создание учетной записи, для того чтобы можно было заказать товар;

2)           Выбрать товар – представляет собой выбор товара зарегистрированным пользователем, который закрепляется за данным пользователем;

3)           Отправить заявку на заказ товара – оформление заявки;

На рисунке 2.3 представлена декомпозиция процесса «Выбрать товар». Данный процесс состоит из следующих этапов:

- Выбрать категорию;

- Просмотреть товары;

- Выбрать наиболее приемлемый.

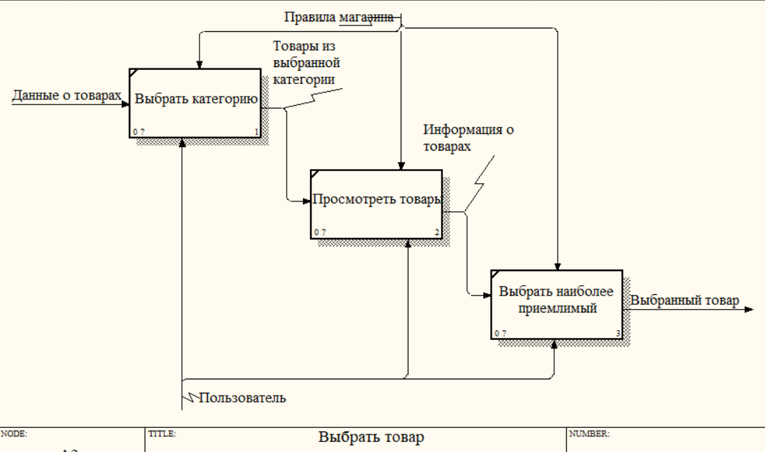


Рисунок 2.3 – Декомпозиция процесса «Выбрать товар»

На рисунке 2.4 представлена декомпозиция процесса «Оплатить товар». Данный процесс состоит из следующих этапов:

- Просмотреть корзину;

- Удалить ненужные товары из корзины;

- Оплатить.

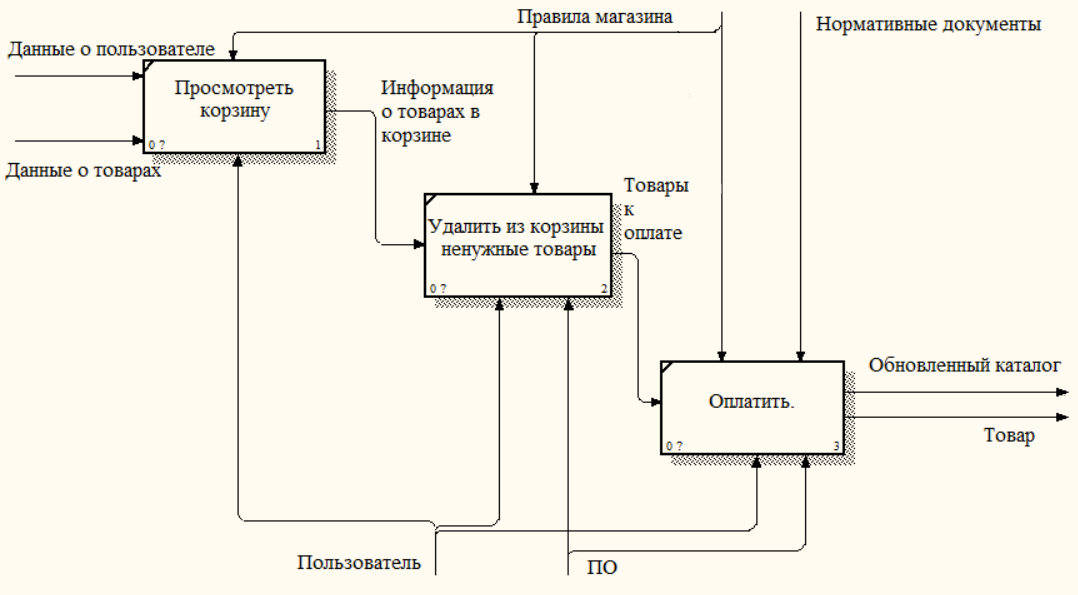


Рисунок 2.4 – Декомпозиция процесса «Произвести оплату»

С помощью данной IDEF-0 диаграммы можно подробно и тщательно рассмотреть процесс заказа товара и его подпроцессы.

3 СПЕЦИФИКАЦИЯ ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

Диаграмма вариантов использования онлайн магазина представлена ниже на рисунке 3.1.

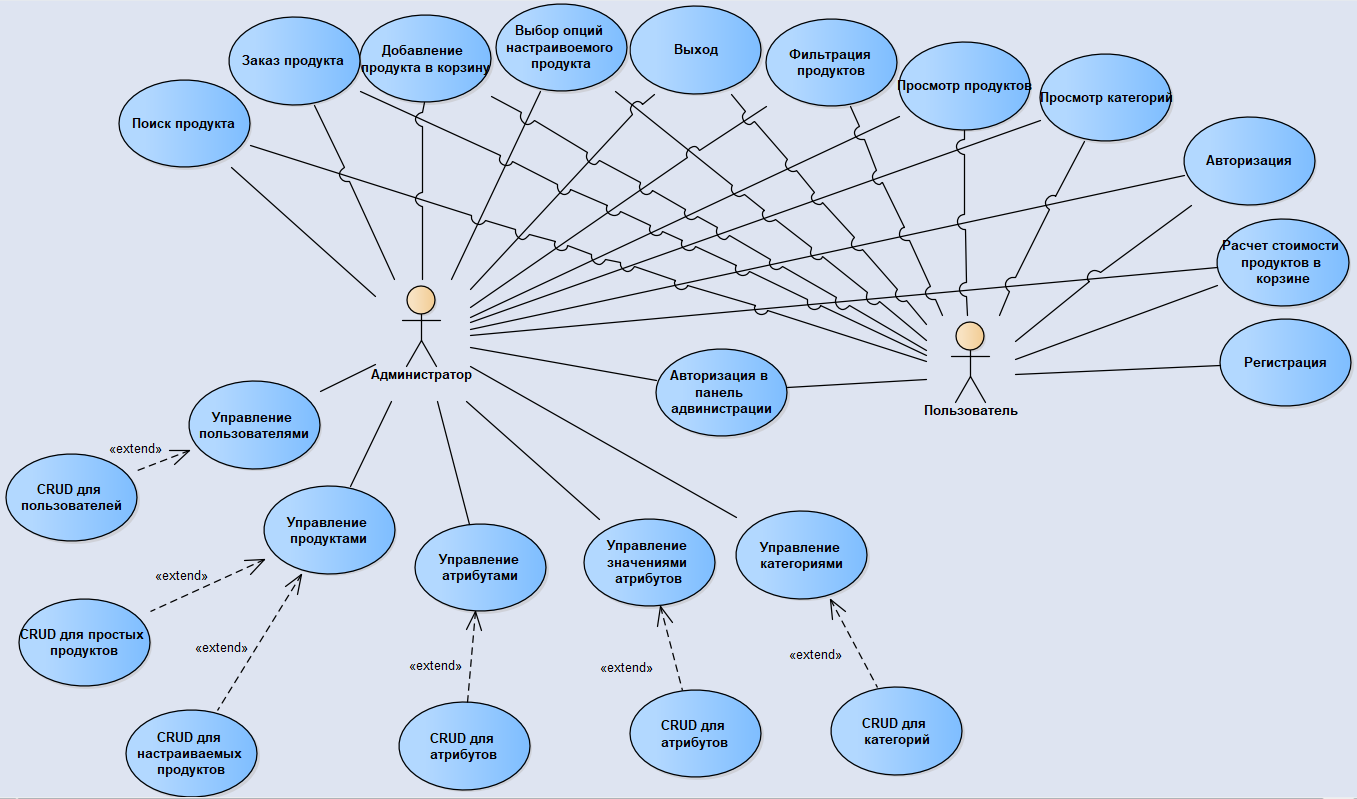


Рисунок 3.1 – Диаграмма вариантов использования

Для организации полноценной работы предоставленной системы требуются Администратор и Пользователь.

В основные функции Администратора входит не только контроль за корректной работой системы, но и такие функции как:

1. управление товарами и настройками связанными с ними сущностей, что включает их удаление, добавление и редактирование;
2. управление пользователями, а именно изменение имени, пароля, почты, номера телефона и роли пользователя;
3. управление категориями объявлений, что включает их удаление, добавление и редактирование.

К тому же в системе реализованы общие для администратора и пользователя функции, такие как:

1. авторизация, если аккаунт зарегистрирован;
2. регистрация (только обычного пользователя), если аккаунт в системе отсутствует;
3. Просмотр товаров и категорий;
4. Поиск товаров
5. Фильтрация товаров
6. Выбор характеристик настраиваемых товаров.
7. Добавления товаров в корзину
8. Оплата

4 ИНФОРМАЦИОННАЯ МОДЕЛЬ СИСТЕМЫ И ЕЁ ОПИСАНИЕ

Стадия приведения данной базы данных к нормальной форме – это часть процесса проектирования баз данных. А приводить к требуемой нормальной форме базу данных следует еще до того, как начнется непосредственно ее программная реализация, то есть, переде тем, как начнется непосредственно создание БД.

Нормальная форма, это свойство отношений в реляционной модели данных, характеризующее их с точки зрения [избыточности](http://zametkinapolyah.ru/zametki-o-mysql/proektirovanie-baz-dannyx-informacionnaya-izbytochnost-izbytochnost-dannyx-v-baze-dannyx-problemy-voznikayushhie-iz-za-informacionnoj-izbytochnosti.html), которая иногда может привести к различным ошибочным результатам выборки или изменения данных. Обычно нормальная форма должна определяться как совокупность различных требований, которым должны удовлетворять отношения.

Сам процесс преобразования базы данных к некоторой нормальной форме называется нормализацией. Нормализация предназначена для приведения структуры БД к виду, обеспечивающему минимальную логическую избыточность и читаемость, и не имеет целью уменьшение или увеличение производительности работы или же уменьшение или увеличение физического объёма базы данных. База данных обычно считается нормализованной, если все ее таблицы представлены минимум в третьей нормальной форме.

Первая нормальная форма(1НФ) – каждый атрибут отношения должен хранить некоторое атомарное значение, каждое отношение должно содержать одинаковое количество атрибутов (столбцов).

Вторая нормальная форма (2НФ) – отношение находится во второй нормальной форме, при условии, что оно находится в 1НФ, и при этом все не ключевые атрибуты должны зависеть лишь от первичного ключа.

Отношение находится в третьей нормальной форме (3НФ), если оно находится во второй нормальной форме и все не ключевые атрибуты зависит лишь от первичного ключа и не имеют зависимостей между собой.

В результате нормализации базы были выделены следующие сущности:

* Пользователь;
* Продукт;
* Категория;
* Настраиваемый продукт;
* Атрибут;
* Значение атрибута;
* Изображение;

Модель базы данных показана на рисунке 4.1. Каждая сущность в базе данных обладает своим набором атрибутов и обязательным первичным ключом.

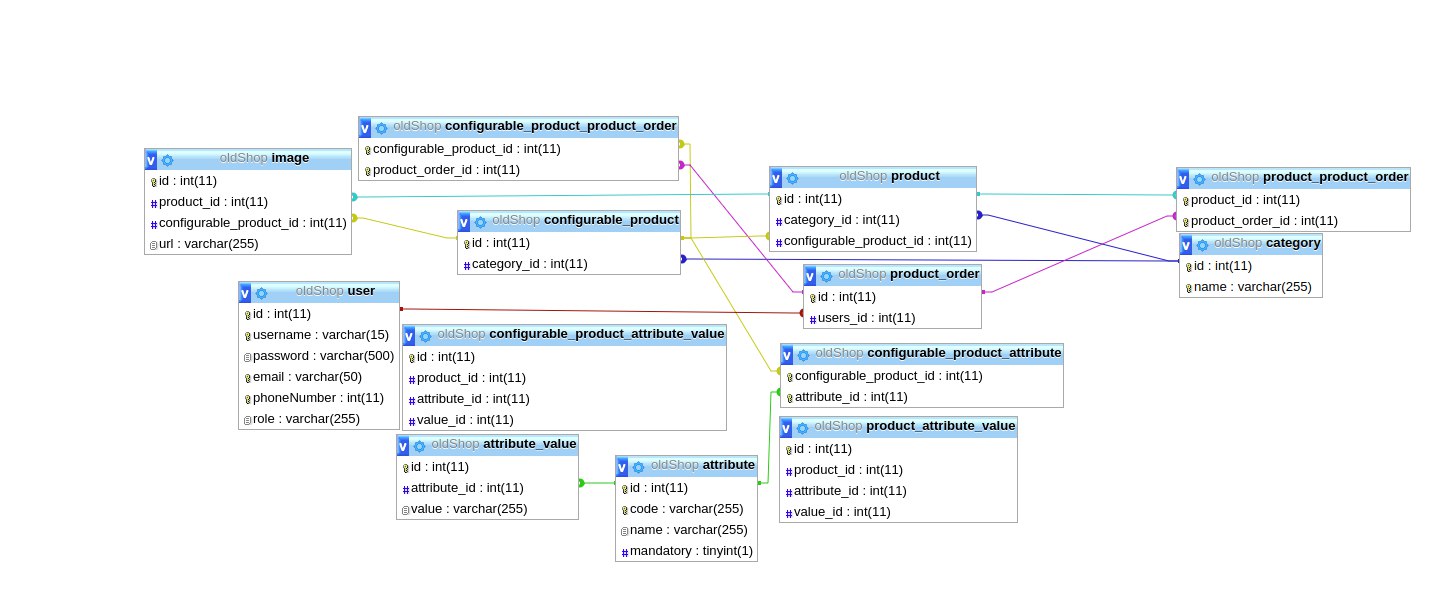


Рисунок 4.1 – Схема базы данных

Рассмотрим сущность Пользователь. Она имеет следующие поля: ИД, роль пользователя, логин, пароль, электронная почта, номер телефона. Поле электронная почта необходимо для того, чтобы администратор мог связаться с пользователем для сообщения статуса заказа. Пароль при сохранении в базу хешируется с помощью стандартного алгоритма UNIX.

Сущность Атрибут содержит имя ИД, код атрибута, название атрибута и поле “mandatory”. Название атрибута – значение которое будет отображаться при просмотре, ИД и код атрибута позволяют его идентифицировать. А поле “mandatory” указывает является ли атрибут обязательным для заполнения.

Далее сущность Значение атрибута, которая характеризуется следующими полями: ИД, ИД атрибута, значение атрибута. ИД атрибута связывает значение с атрибутом которому оно принадлежит.

Очередная сущность - сущность Продукт. Она состоит из поля ИД, ИД категории и ИД настраиваемого продукта. Поле ИД позволяет явно идентифицировать продукт, ИД категории позволяет узнать принадлежность продукта к категории и полу ИД настраиваемого продукта указывает на то принадлежит ли продукт к продуктам созданным по шаблону настраиваемого продукта.

Таблицы находятся в 1НФ, так как вся информация, находящаяся в ячейках таблиц неделима.

Так же таблицы соответствуют и 2НФ, так как все поля, содержащиеся в таблице, зависят только от первичного ключа.

Например, в таблице Пользователь поля роль пользователя, логин, пароль, электронная почта, номер телефона никак не могут быть связаны между собой, они зависят только от первичного ключа. Такое же состояние у таблицы Атрибут, Продукт и Категория с соответствующими полями, между которыми отсутствует функциональная зависимость.

Из вышеописанного можно уверенно сделать вывод, что все таблицы корректно приведены к 2НФ и 3НФ. Поэтому рассматриваемая база данных имеет корректную структуру и проектировку.

5 ОБОСНОВАНИЕ ВЫБОРА КОМПОНЕНТОВ И ТЕХНОЛОГИЙ ДЛЯ РЕАЛИЗАЦИИ КУРСОВОГО ПРОЕКТА

Во время разработки данного веб-приложения было использовано множество современных технологий, таких как Symfony Framework, Doctrine, jQuery, Composer, Bootstrap, Twig и так далее. Рассмотрим их достоинства и необходимость использования некоторых из них по отдельности.

5.1 Symfony

Symfony — свободный фреймворк, написанный на PHP, который использует паттерн Model-View-Controller. Один из самых популярных и современных PHP-фреймворков на текущий момент. Он поддерживает большое множество всевозможных баз данных, в составе фреймворка поставляется Doctrine, Twig, Composer, а также ряд других не менее известных библиотек. Главной особенностью разработки на Symfony является bundle система. Бандл - это совокупность скриптов (php, js и т. д.), а так же html, css и т. д., которые используются для достижения некоторой единой цели.

Особенность Symfony в том, что "все есть bundle" - ваше приложение, даже функционал ядра. Обычно бандлы делают полностью независимыми друг от друга, что позволяет значительно упростить разработку и поддержку проектов путем переноса полностью функционирующих частей из одного проекта в другой.

Symfony использует архитектуру модель-представление-контроллер (model-view-controller), что позволяет полностью разделять объекты, бизнес-логику и внешний вид приложения. Symfony обеспечивает полностью готовые компоненты, которые могут быть использованы (и используются) для разработки данного веб-приложений. Все эти компоненты должны быть слабо связаны между собой и при желании можно изменить, например, модель приложения, не внося существенные изменения в остальные два компонента.

5.2 Doctrine

Doctrine является инструментом объектно-реляционного отображения данных для PHP с открытым исходным кодом. Он предоставляет фреймворк для отображения объектно-ориентированной модели данных в традиционные реляционные базы данных. Doctrine не только заботится об отражении данных сущностей PHP в таблицы БД (и конвертации типов данных PHP в типы данных SQL), но также обеспечивает реализацию запросов данных и поисковые средства и может значительно сократить время разработки, которое тратится на ручное написание данного SQL кода.

Doctrine был выбран для работы с БД так как обеспечивает полностью автоматическую генерацию и обновление всех таблиц, а так же сокращение программного кода, поскольку основные запросы к базе данных обычно представлены как методы. Цель Doctrine – предоставить мощные инструменты, для достаточно простого и гибкого взаимодействия с базами данных. Doctrine создаёт свой слой абстракций над БД и позволяет связать их с концепциями объектно-ориентированного программирования. Работа с базой данных при помощи Doctrine представляет собой лишь взаимодействие с объектами. Каждая таблица – объект. При необходимости изменения СУБД код переписывать не нужно.

5.3 jQuery

jQuery - библиотека JavaScript с открытым исходным кодом для динамических web-приложений. jQuery предоставляет удобный набор инструментов для работы на стороне клиента. Облегчает работу с событиями. Библиотека jQuery содержит функциональность, полезную для максимально широкого круга задач.

5.4 Twig

Twig является современным шаблонизатором, так же написанным на языке PHP. Позволяет жёстко разграничить понятия Controller и Viewer, так как при написании шаблонов всех страниц используются только характерные для Viewer функции и синтаксис, который в свою очередь является значительно удобнее и сокращённым по сравнению с PHP.

Twig обладает собственной встроенной системой кеширования страниц и позволяет расширять свой функционал в зависимости от подробностей при разработке проекта путём написания дополнительных классов и методов, является шаблонизатором по умолчанию в Symfony.

5.5 Composer

Composer — это достаточно новый и уже относительно популярный менеджер зависимостей для языка PHP. Вы можете описать в конфигурации от каких библиотек зависит ваш проект и Composer установит все необходимые библиотеки за вас.

1. МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ

В данном курсовом проекте будут рассматриваться некоторые модели представления разрабатываемой системы, а именно:

– диаграмма последовательности;

– диаграмма состояния объектов;

– диаграмма компонентов;

– диаграмма развертывания;

– диаграмма классов

Рассмотрим каждую из них более детально.

6.1 Диаграмма последовательности создания продукта

Диаграмма последовательности создания нового продукта отображена на рисунке А.1 Приложения А.

Для начала администратор выбирает пункт «Добавить продукт» на странице продуктов в панели администратора. Далее открывается страница создания продукта, где нужно заполнить данные продукта.

После нажатия на кнопку «Сохранить», информация передается на ProductController.php, откуда данные попадают на сервер. Запрос на создание нового сообщения приходит в ProductAPIController, где обрабатывается в методе addProduct(). Этот метод в свою очередь перенаправляет запрос в

ProductService. И уже сервис обращается к репозиторию, где объявлены методы, которые далее будут обращаться непосредственно к базе данных, в которой хранится все информация. В результате имеем возможность просмотреть созданный продукт.

6.2 Диаграмма состояний авторизации пользователя

Диаграмма состояний изображена на рисунке А.2 Приложения А

Если пользователь хочет авторизоваться, то ему необходимо заполнить форму авторизации(ввести логин и пароль) и нажать кнопку Войти. Далее введенные данные сохраняются в сессию и данные отправляются на сервер для их проверки. В случае, если данные не соответствуют, форма возвращается пользователю с просьбой внести изменения во внесенную информацию, так же в возвращенной форме из сессии получается последнее введенное имя пользователя и заполняется автоматически. Если данные введены верно, то пользователь авторизовывается.

6.3 Диаграмма компонентов

Данная диаграмма представлена в приложении А на рисунке А.3.

На представленной диаграмме компонентов видно разделение работ серверов. На сервере 2 по диаграмме видно, что нету бизнес-логики, сервер 2 предназначен для получения данных с HTML страницы и отправки этих данных на сервер 1. Таким образом сервер2 является связующим звеном между сервером 1 и пользователем. Сервер 1 содержит в себе всю бизнес-логику, так же сервер 1 обращается к базе данных.

6.4 Диаграмма развертывания

Данная диаграмма показывает все присутствующие аппаратные компоненты приложения, а именно сервер 1 на одном персональном компьютере, сервер 2 на другом и база данных на третьем. Данная диаграмма необходима для корректного представления взаимодействия компонентов в распределенной среде. Данную диаграмму можно просмотреть в приложении А на рисунке А.4 .

6.5 Диаграмма классов

В приложении А на рисунках А.5, А.6 и A.7 изображены диаграммы классов для двух модулей системы сервер 1 и сервер 2.

Каждый класс, отображенный на диаграмме, каким-либо образом связан с другим. В связи с тем, что классов очень много, они были разбиты по папкам. В первых двух пакетах находятся классы описывающие бизнес-логику приложения, в пакете Controller классы являются связывающим звеном между бизнес-логикой и пользователем.

* 1. Блок схемы алгоритмов, реализующих бизнес-логику системы

**6.6.1 Схема работы всей программы**

При входе на сайт пользователь попадает на главную страницу, где он может сделать такие действия как “Выбрать категорию товаров”, “Перейти на страницу категорий”, “Выбрать продукт”, “Просмотреть корзину” (Если пользователь авторизован), “Авторизоваться” если пользователь не авторизован и “Выйти” если пользователь авторизован. При просмотре товара если это настраиваемый товар, пользователь может выбрать допустимые характеристики товара. Характеристики выбираются поочередно. При заполнении всех характеристик обновляется изображение товара и становится доступной кнопка “Добавить в корзину”, далее пользователь может “Добавить в корзину” и перейти в нее, где можно “Оплатить товары”, “Очистить корзину” или “Удалить товар”. Далее пользователь может покинуть сайт или вернутся на главную страницу.

**6.6.2 Схема алгоритма добавления продукта в корзину**

Алгоритм, представленный на рисунке Б.2 приложения Б, начинается с того, что пользователь решает добавить продукт в корзину. Сначала он выбирает нужные характеристики, то есть поочередно выбирает из доступного списка значения атрибутов. При заполнении всех характеристик обновляется изображение товара и становится доступной кнопка “Добавить в корзину”, далее пользователь может “Добавить в корзину” и перейти в нее.

7 ОПИСАНИЕ ПРИМЕНЕНИЯ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ

В ходе проектирования данного приложения были детально изучены и использованы различные паттерны проектирования.

1. Паттерн MVC (Model View Controller). Блок Модель(Model) инкапсулирует данные приложения. Модуль Представление(View) отвечает за отображение всех данных пользователю. Контроллер(Controller) отвечает за обработку входящих запросов пользователей и передачу данных через ответ модулю View для их обработки. В данном проекте моделью является весь пакет service, контроллерами – пакета Controller, а представлениями – пакета views (набор файлов расширения .html.twig).
2. Паттерн Внедрение зависимости (Dependency Injection). Предоставляет возможность внедрения любых внешних зависимостей. Объект перекладывает всю заботу о создании различных зависимостей специальному механизму. В данном проекте файлом кофигурации этого модуля является services.yml, внутри которого указаны все зависимости бандла.
3. Паттерн Сущность-Атрибут-Значение(EAV) – используется для описания различных сущностей, в которых потенциальное количество атрибутов (свойств) огромно, но при этом для конкретной сущности применяется лишь несколько выбранных из них Для этого в проекте были введены такие модели как Attribute и AttributeValue. Так же этот паттерн помогает удобно работать с настраиваемыми продуктами.

Были реализованы паттерны GoF:

1. Паттерн Фасад (Facade). Позволяет скрыть сложность данной системы путем сведения внешних вызовов сервисов к одному объекту. В проекте можно увидеть в классе Methods. Который позволяет обращаться к сторонним сервисам используя один метод.
2. Паттерн Одиночка – гарантирует создание лишь одного экземпляра данного класса. Используется в Service Container. Так же он предоставляет глобальный доступ к контейнеру.
3. Паттерн Цепочка обязонностей(Chain of responsibility) – преднозначен для организации и распределения уровней отвтственности. Используется при маршрутизации. Изначально запрос идет на главный файл routing.yml, далее при необходимости может направляется в бандл и там уже в нужный файл роутинга.

8 РУКОВОДСТВО ПО РАЗВЕРТЫВАНИЮ СИСТЕМЫ

Все исполняемые команды ориентированы на ОС Ubuntu. При развертывании на другую ОС необходимо следовать пунктам и выполнять команды соответствующие данной ОС.

Устанавливаем Apache2:

sudo apt-get update; sudo apt-get install apache2

Далее устанавливаем PHP версии 5.5.9 и выше. Для этого воспользуемся командой:

sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql

Далее необходимо перезагрузить Apache2:

sudo systemctl restart apache2

Для того, что бы подтянуть зависимости в проекте необходимо установить Composer:

sudo apt-get update

sudo apt-get install curl php5-cli git

curl -sS https://getcomposer.org/installer | sudo php -- --install-dir=/usr/local/bin --filename=composer

Затем необходимо зайти поочередно в оба проекта и прописать в терминале команду:

composer install

Далее необходимо настроить подключение к БД в файле:

app/config/**parameters.yml.dist**

**Так же необходимо сгенерировать эту базу данных при помощи скрипта или запустить автоматическую генерацию базы по сущностям, а так же ее наполнение используя фикстуры(при их наличии):**

php bin/console doctrine:schema:create

php bin/console doctrine:fixtures:load

После чего необходимо запустить сервера, воспользуемся командой введенной из в терминал из корня проекта:

bin/console server:start

Первым необходимо запустить сервера1(oldShop). После запуска этой команды должно высветится сообщение о том, что сервер запущен, так же symphony автоматически присваивает ему порт 8000. При запуске сервер2(clientOldShop) ему присвоится порт 8001.

9 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ РАЗРАБОТАННОЙ СИСТЕМЫ И ОЦЕНКА ВЫПОЛНЕНИЯ ЗАДАЧ

Не только пользователь, но и администратор, могут вводит неточные данные, либо совершить действия, приводящие к ошибкам и некорректной работе приложения. Поэтому в рассматриваемом программном продукте было осуществлено множество проверок и предупреждений об ошибках. При попытке отправить ошибочные данные система оповещает пользователя о их неверности и предлагает повторить ввод.

Пошагово рассмотрим функционал интернет магазина.

Зайдем на главную страницу (рисунок 9.1).

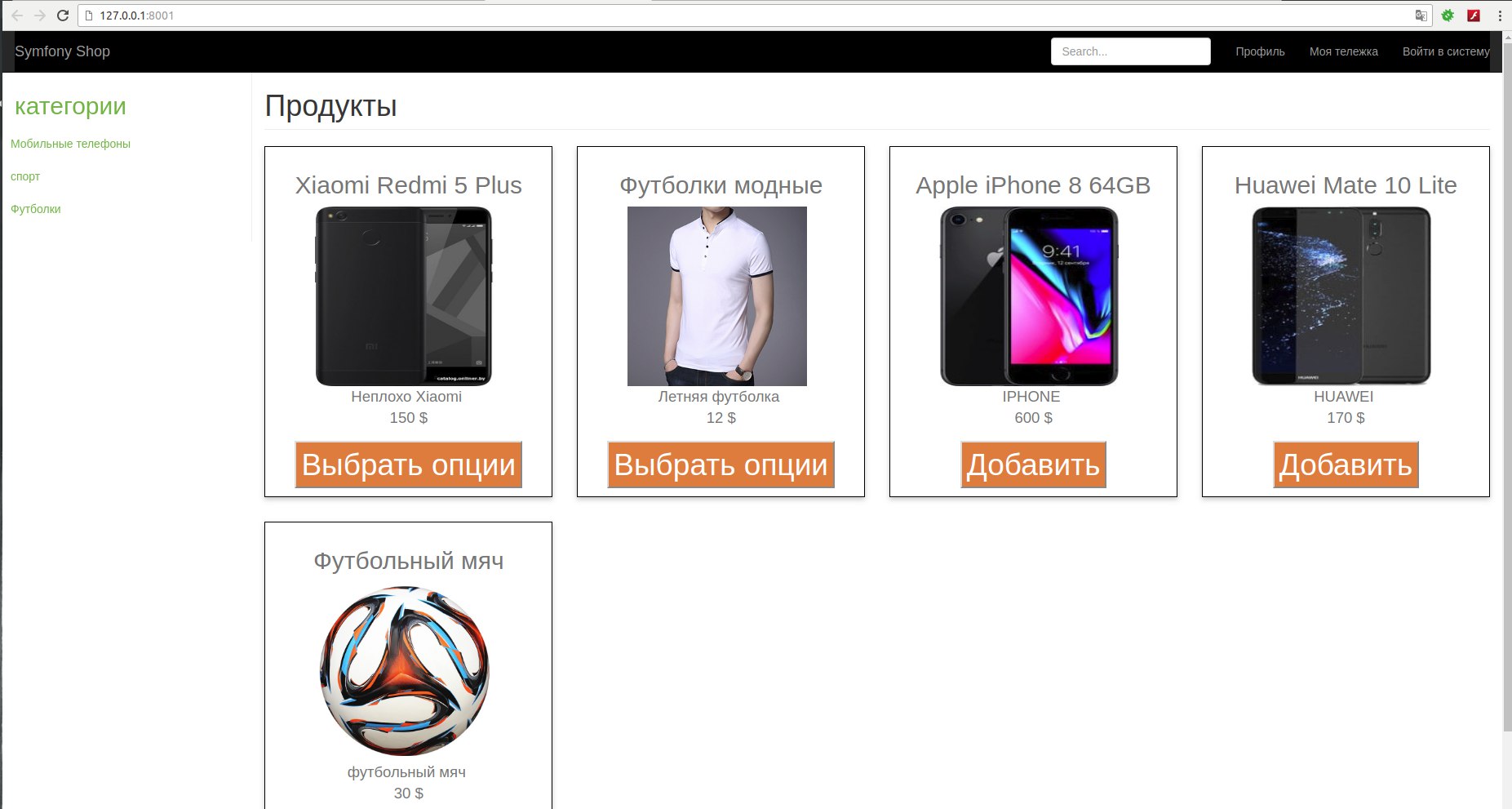


Рисунок 9.1 – Главная страница

Далее нажав кнопку “Войти в систему” выполнится переход на страницу авторизации (рисунок 9.2).

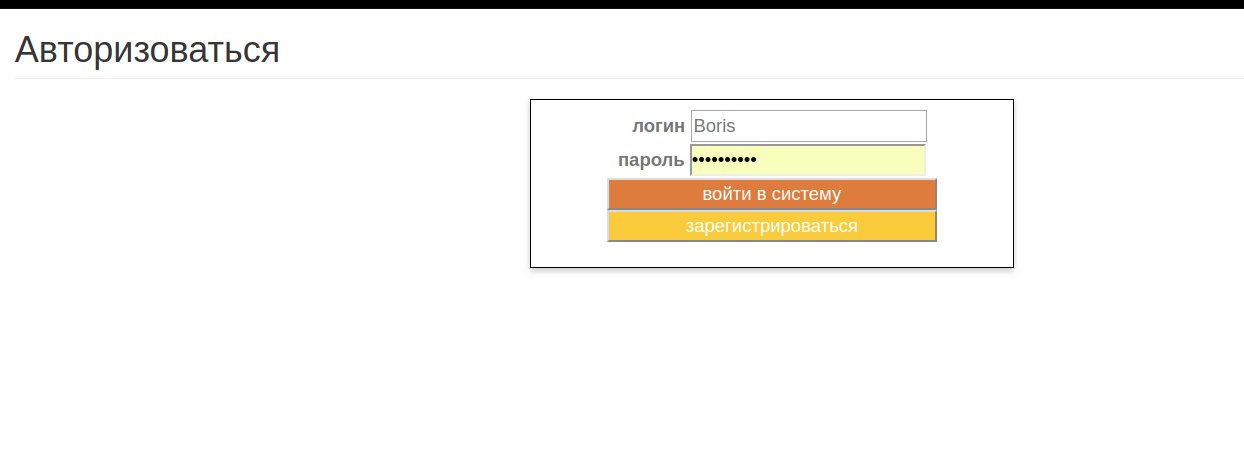


Рисунок 9.2 – Страница авторизации

При попытке ввести неверный пароль или логин пользователь получит предупреждение и сбросится введенный пароль (рисунок 9.3).

Система предусматривает проверку пустых полей. То есть, если поле является необходимым для заполнения, как например логин или пароль, то пользователь получит лаконичное и ясное предупреждение.

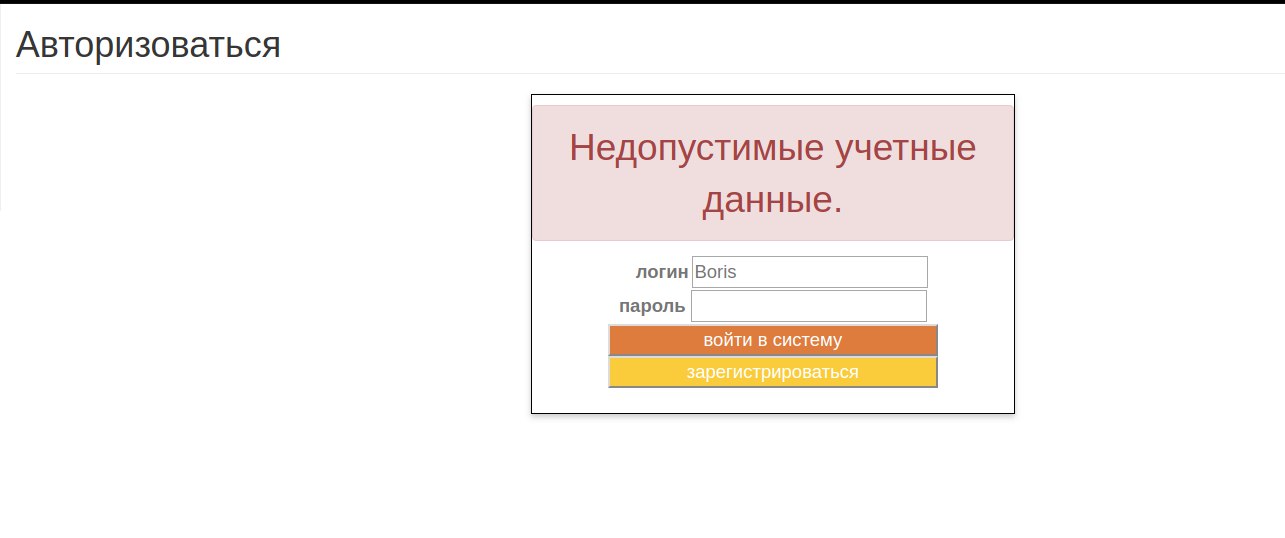


Рисунок 9.3 – Ошибка входа

Нажав кнопку “Зарегистрироваться” пользователь попадет на страницу регистрации (рисунок 9.4). Попробуем ввести неверные данные.

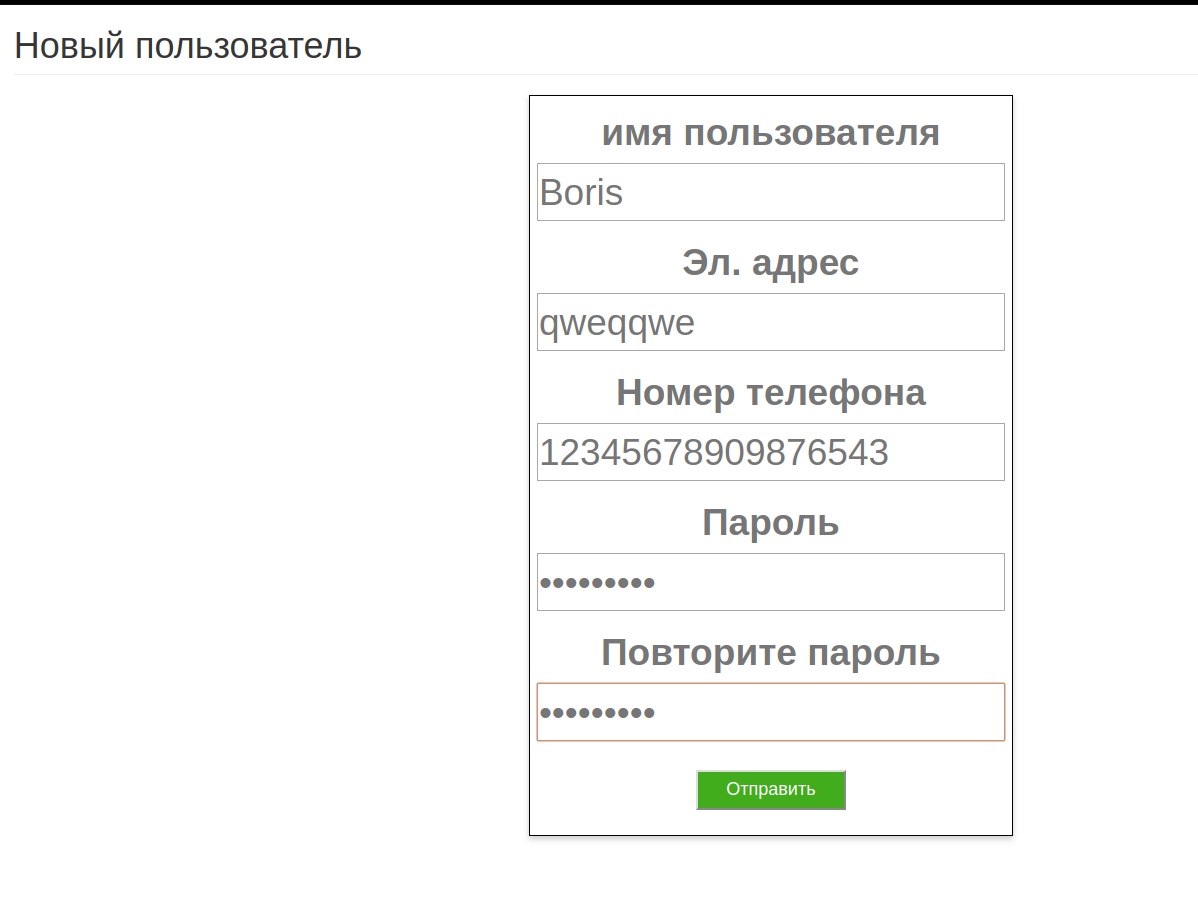


Рисунок 9.4 – Неверно заполненная форма регистрации

После того как пользователь нажмет кнопку “Отправить” он получит следующее сообщение (рисунок 9.5).

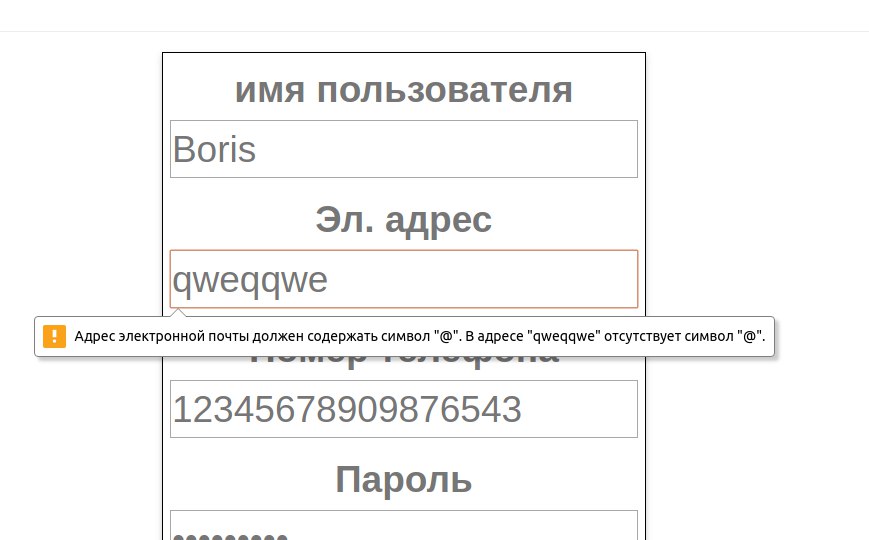


Рисунок 9.5 – Неверно заполненная почта

После изменения почты и повторной отправки формы пользователь вновь увидит, что он ошибся (рисунок 9.6).

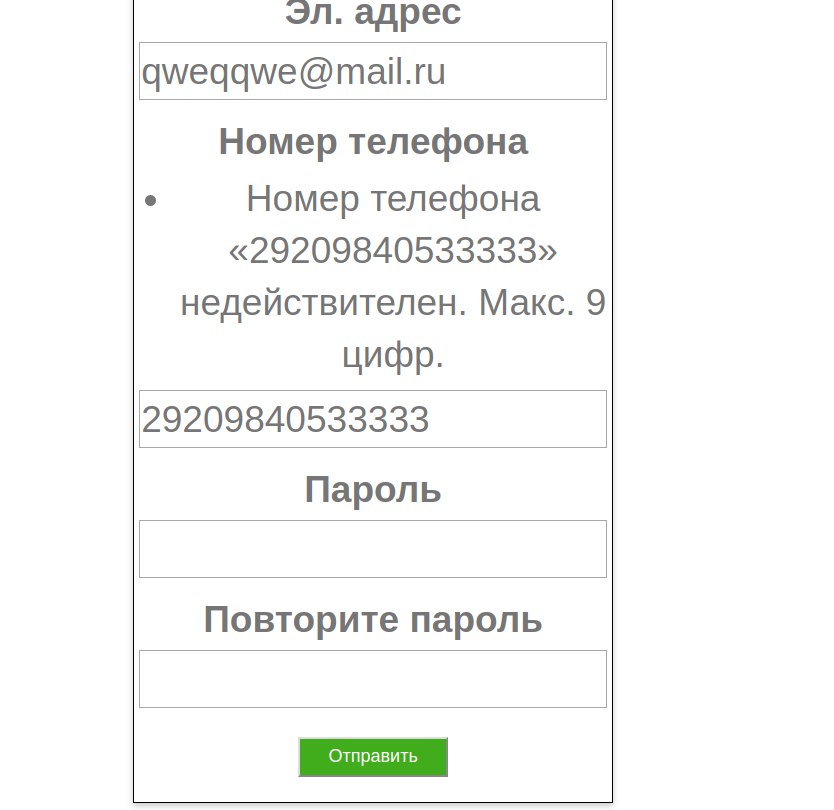


Рисунок 9.6 – Неверный телефон

При попытки ввести разные пароли пользователь получит сообщение (рисунок 9.7).

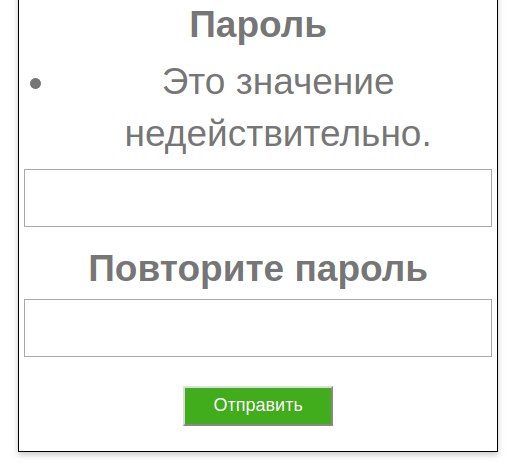


Рисунок 9.7 – Неверные пароли

При вводе верных данных пользователь успешно зарегистрируется. Далее перейдем на страницу категорий (рисунок 9.8). Здесь пользователь может наглядно увидеть все существующие категории.

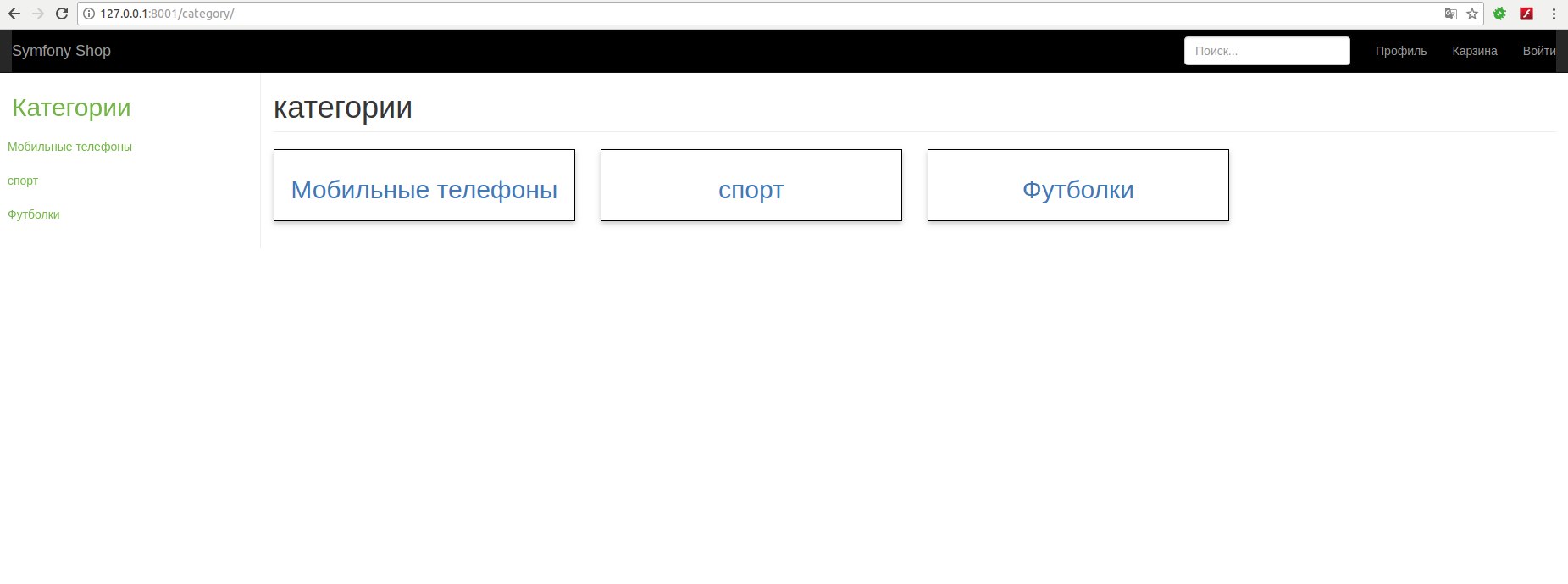


Рисунок 9.8 – Страница категорий

После выбора категории пользователь увидит товары содержащиеся в этой категории и их краткое описание (рисунок 9.9).

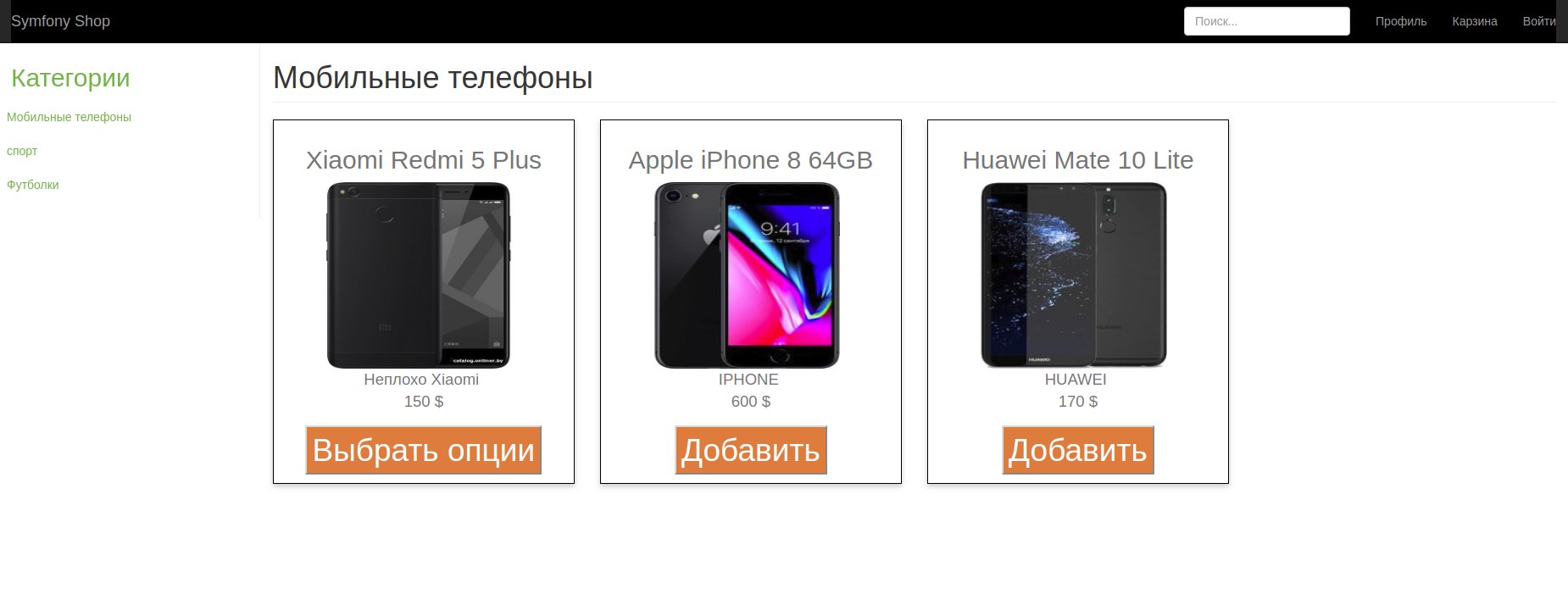


Рисунок 9.9 – Товары в категории

Пользователь может выбрать товар (простой или настраиваемый). После выбора продукта пользователь попадает на страницу с подробной информацией о товаре (рисунок 9.10).



Рисунок 9.10 – Простой продукт

При нажатии на кнопку “Добавить в корзину” продукт добавляется в корзину и на странице обновляется кнопка на “Удалить из корзины” (рисунок 9.11).

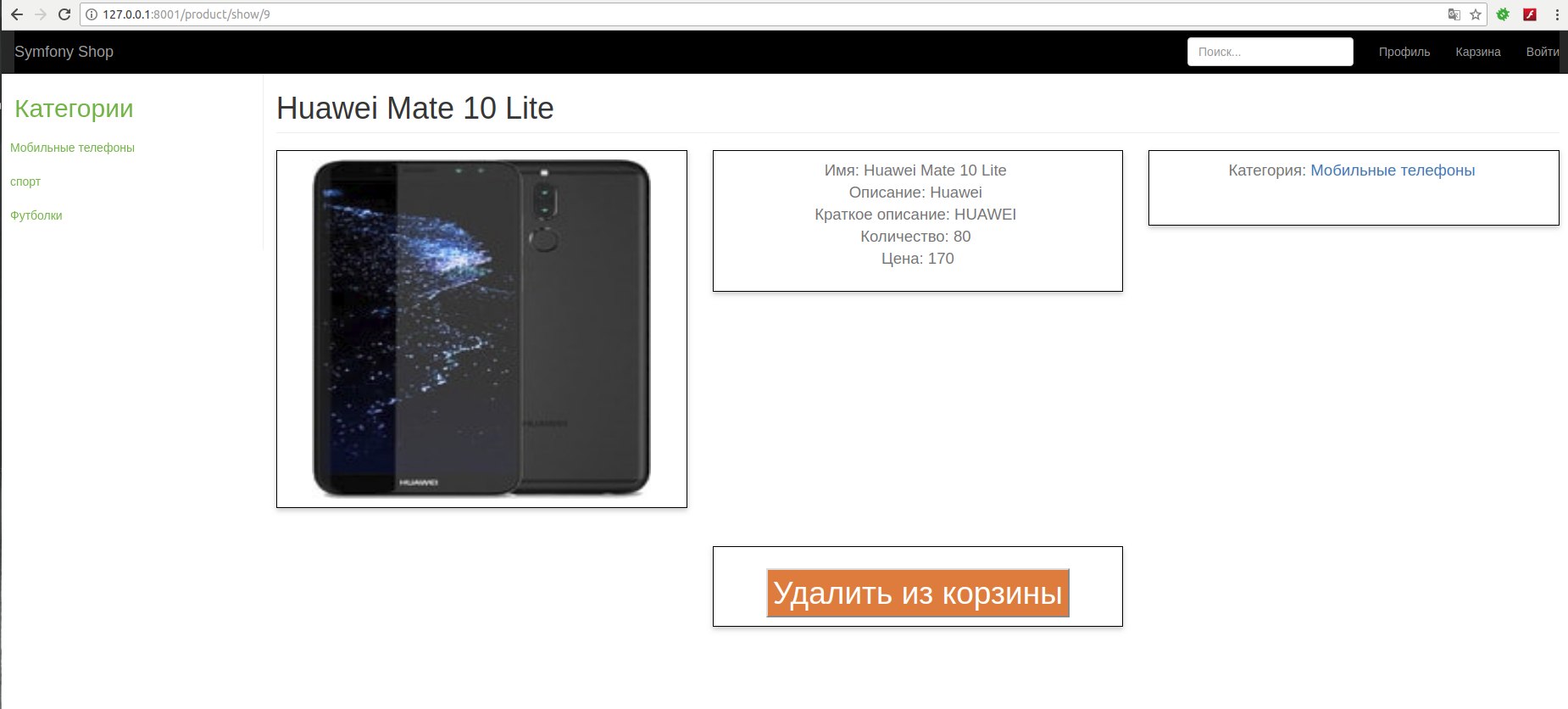


Рисунок 9.11 – Добавленный продукт

Так же пользователь может просмотреть настраиваемый товар (рисунок 9.12). Этот товар содержит в себе несколько простых отличающихся между собой лишь значениями атрибутов, например размером или цветом.

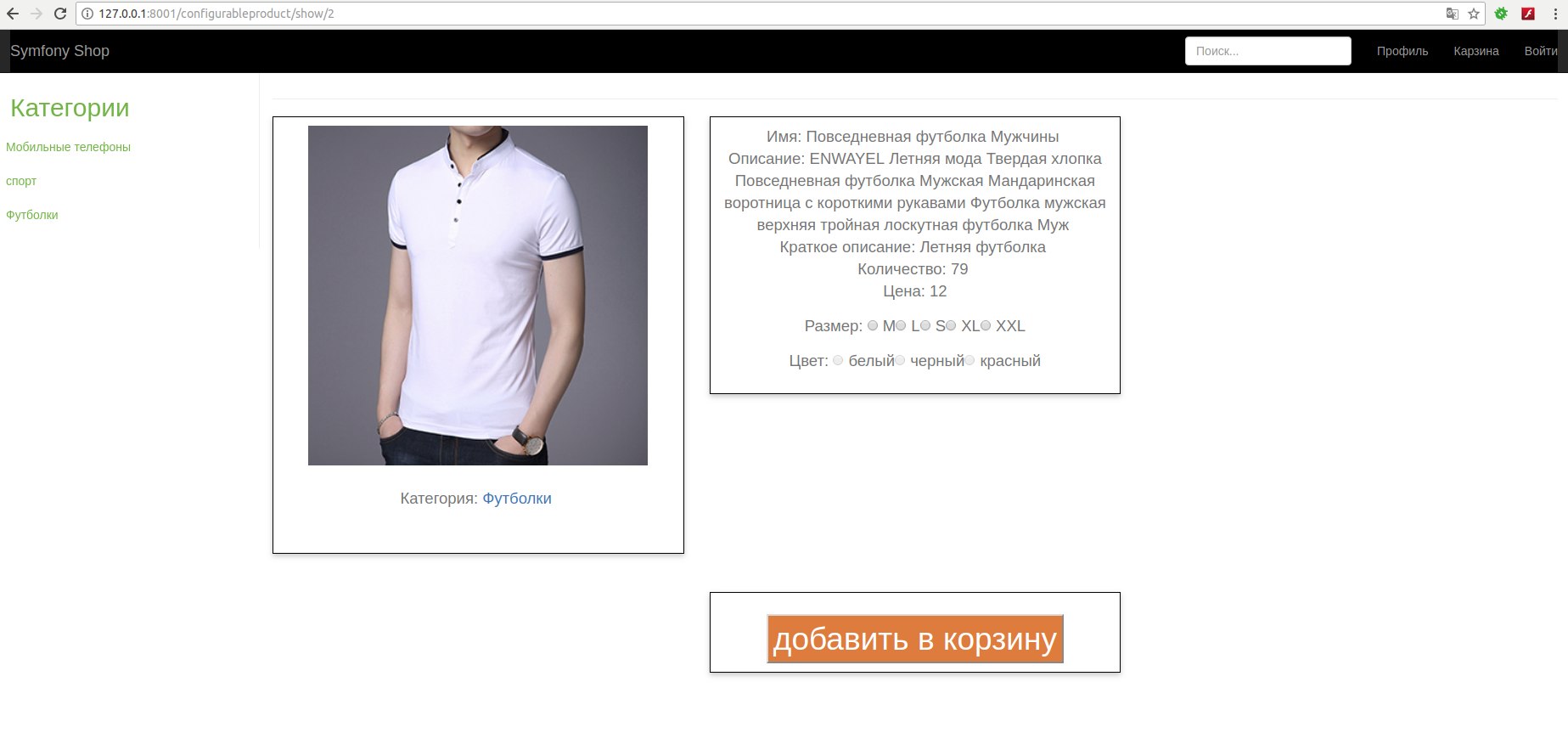


Рисунок 9.12 – Настраиваемый товар

Далее пользователь может выбирать опции для данного товара (рисунок 9.13).

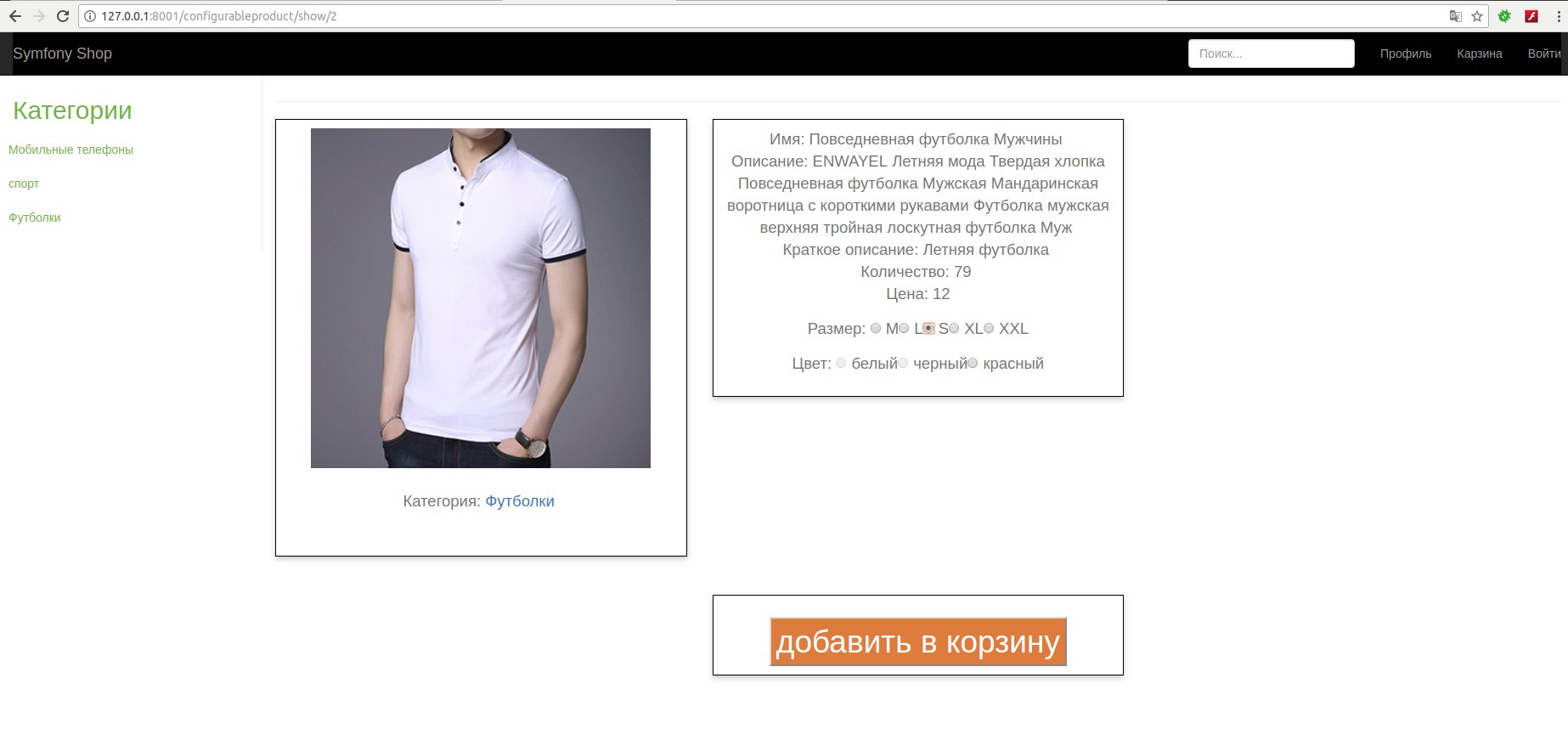


Рисунок 9.13 – Выбор опции

После выбора всех опций информация о товаре и картинка обновляются, выводится информация простого товара который соответствует данным опциям (рисунок 9.14).

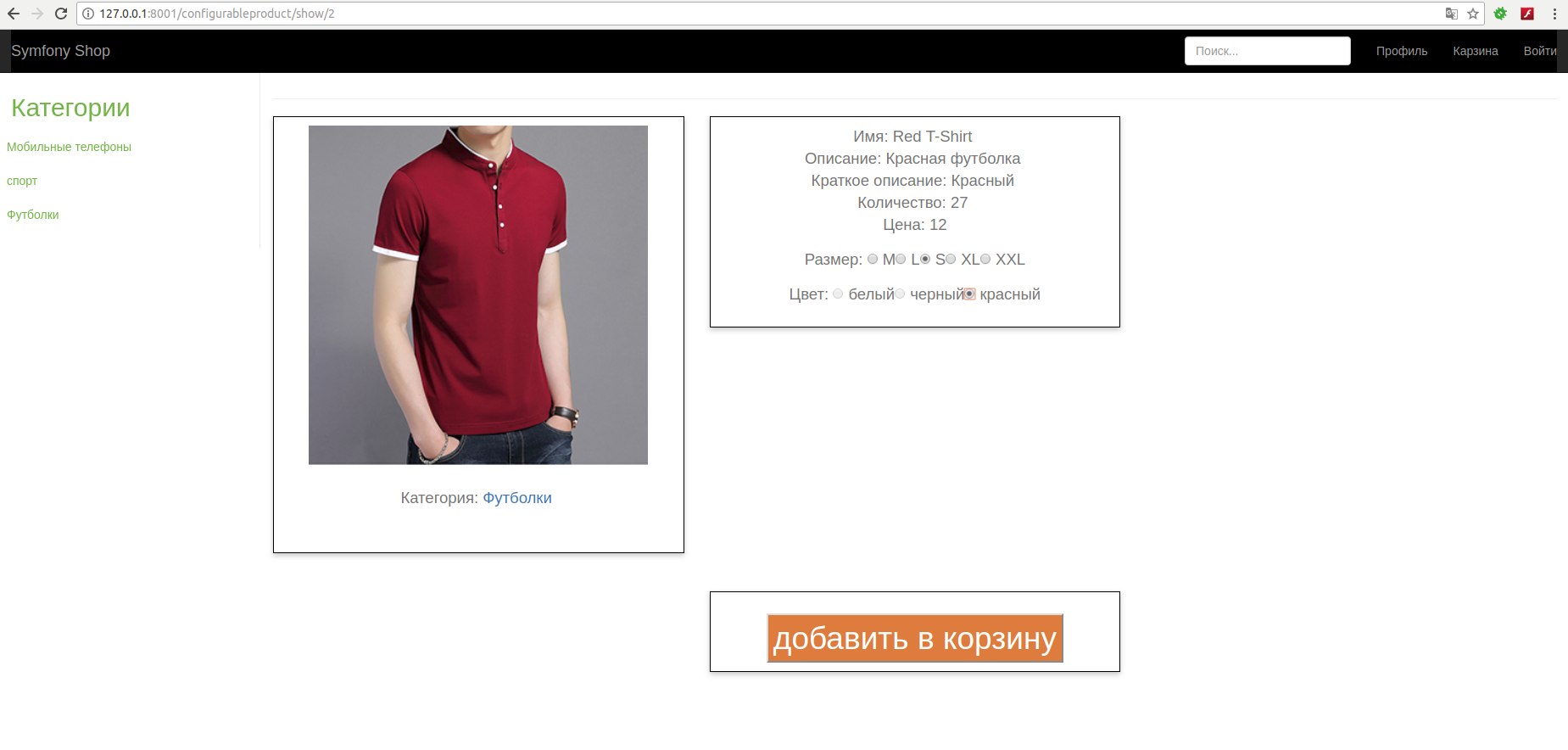


Рисунок 9.14 – Обновление товара

При добавлении продукта стоит проверка на то, что бы все опции были выбраны. В корзину добавляется простой товар и пользователя перенаправляет на выбранный им товар (рисунок 9.15).

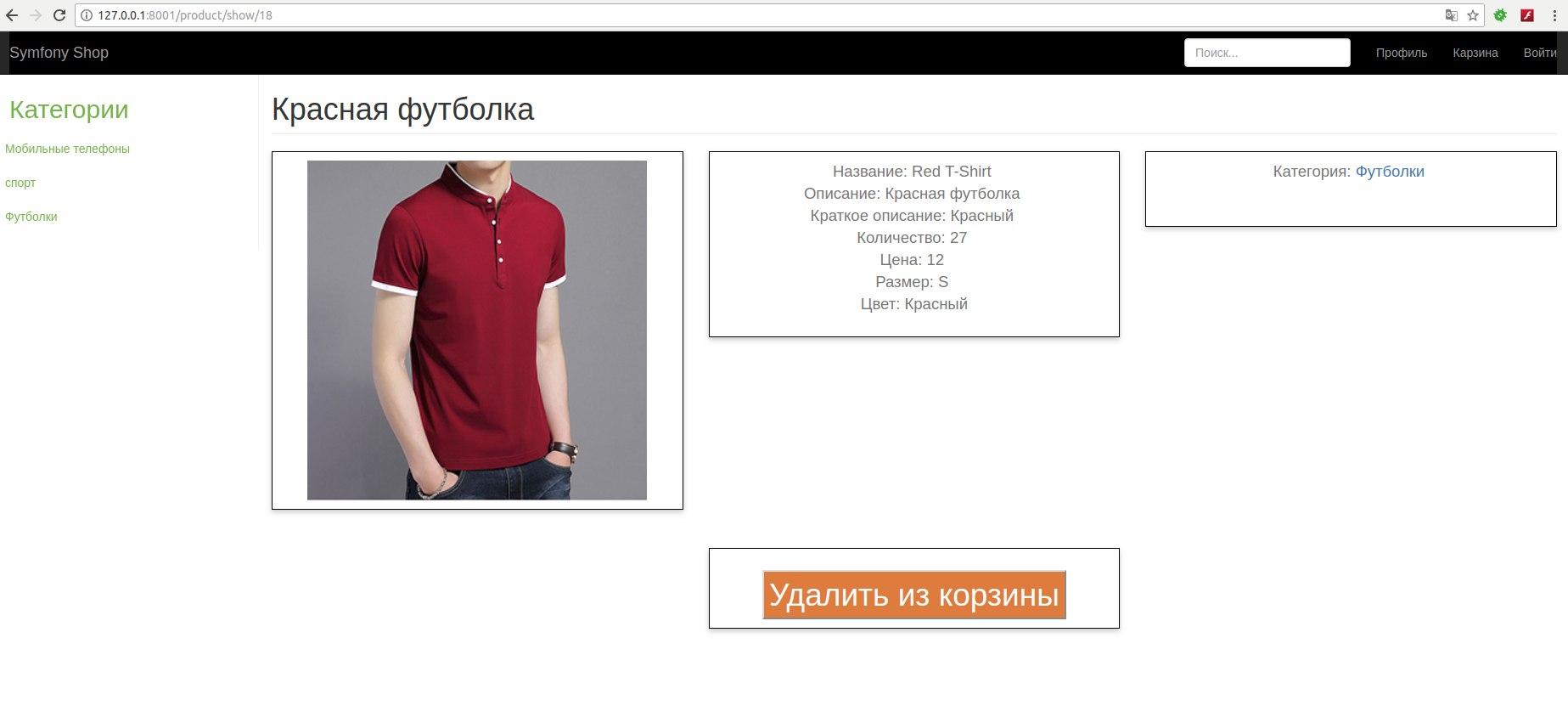


Рисунок 9.15 – Добавленный товар

Пользователь может просмотреть товары добавленные в корзину (рисунок 9.16).

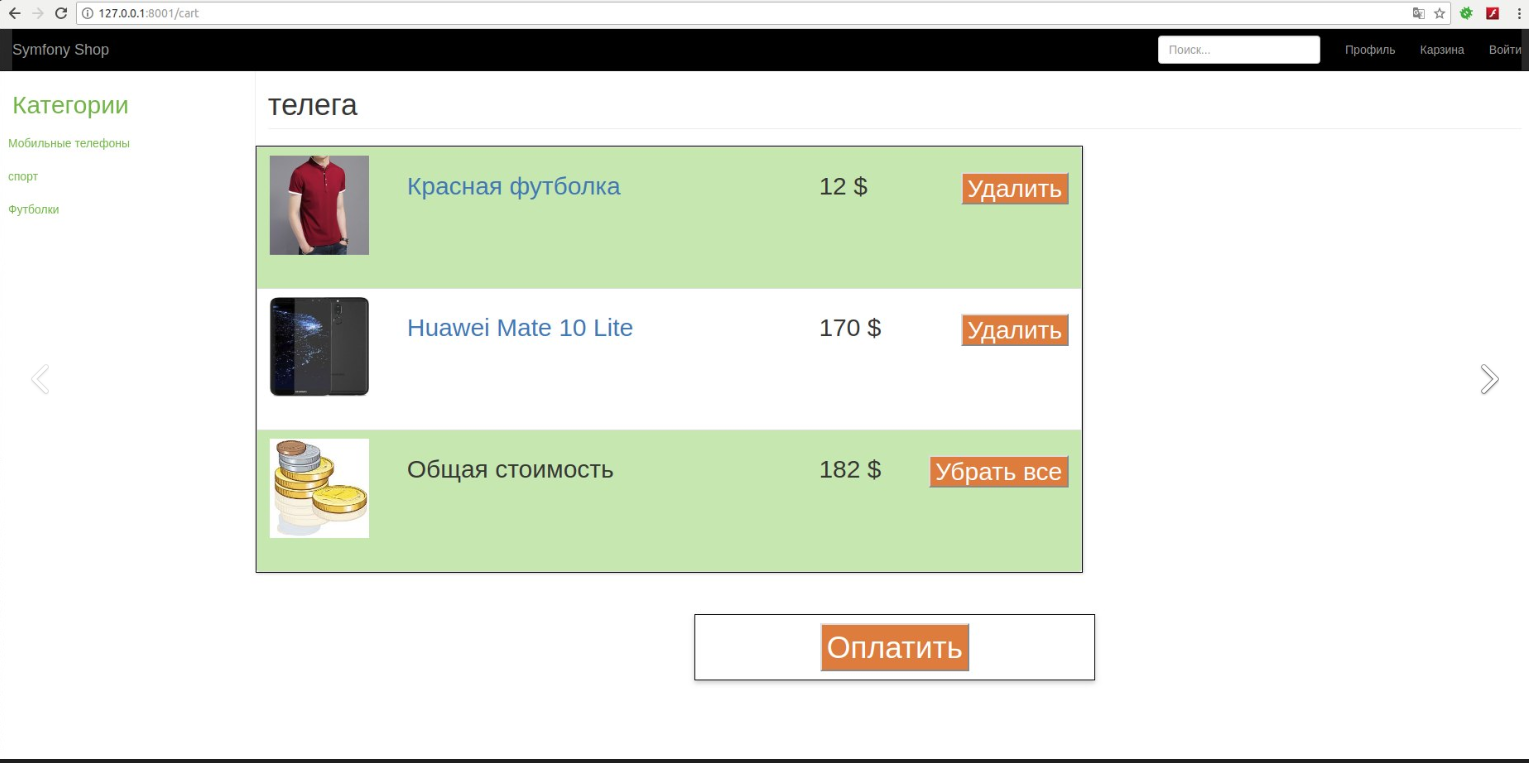


Рисунок 9.16 – Корзина

Если пользователь добавил лишний продукт в корзину, то его всегда можно удалить (рисунок 9.17).

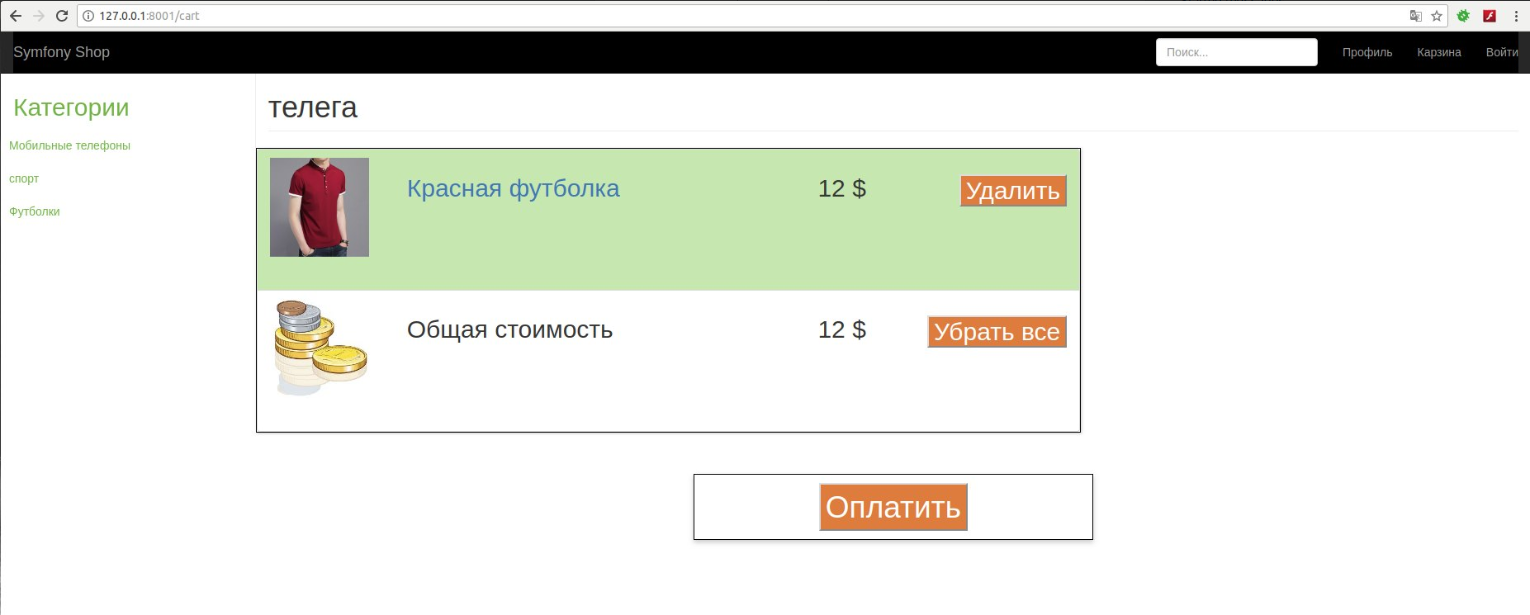


Рисунок 9.17 – Удаление продукта

Далее при желании пользователь может либо оплатить свои покупки либо очистить корзину (рисунок 9.18).

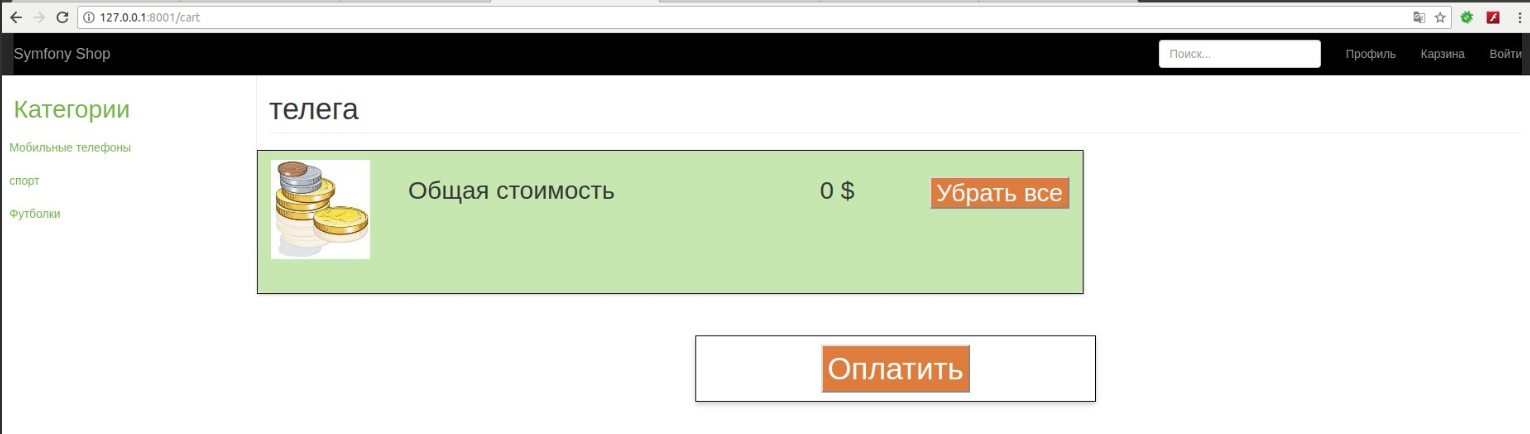


Рисунок 9.18 – Пустая корзина

Так же пользователь может воспользоваться строкой поиска, поиск ищет только по названию (рисунок 9.19).

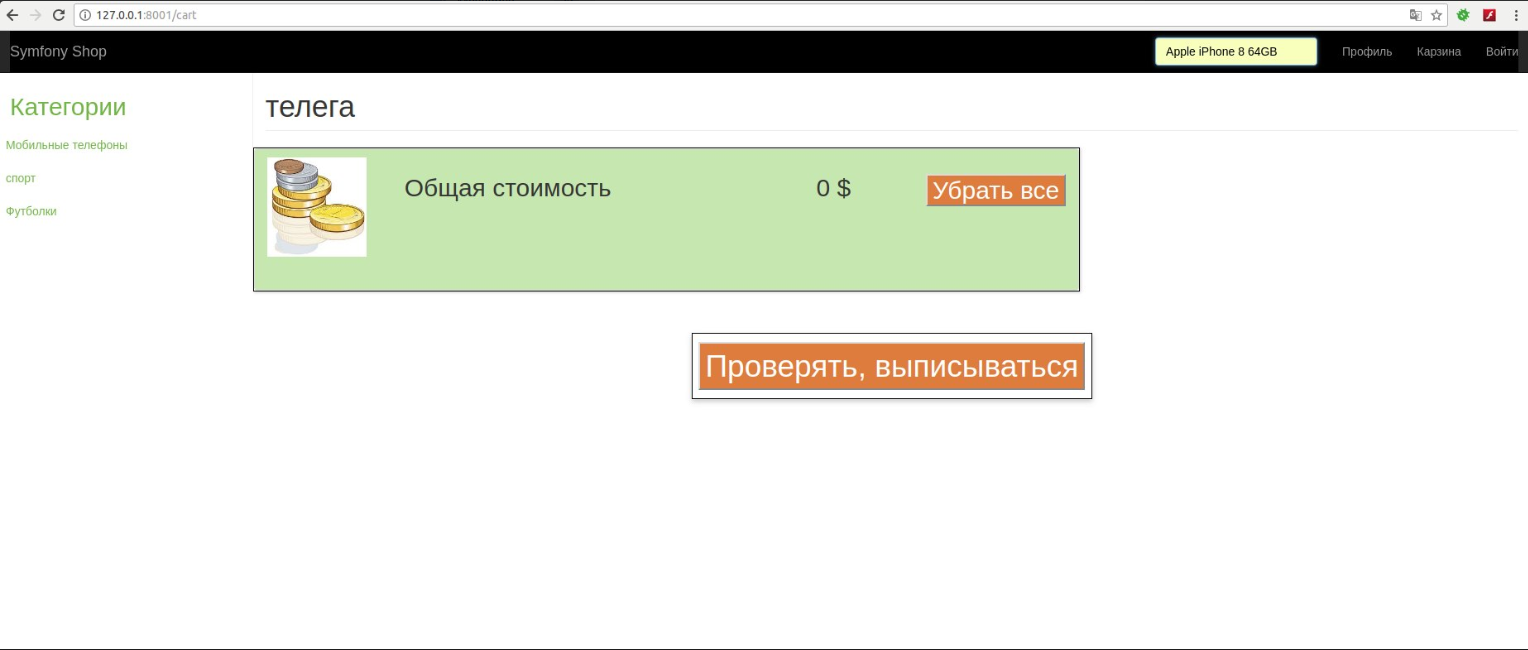


Рисунок 9.19 – Поиск продукта

Результаты поиска (рисунок 9.20).

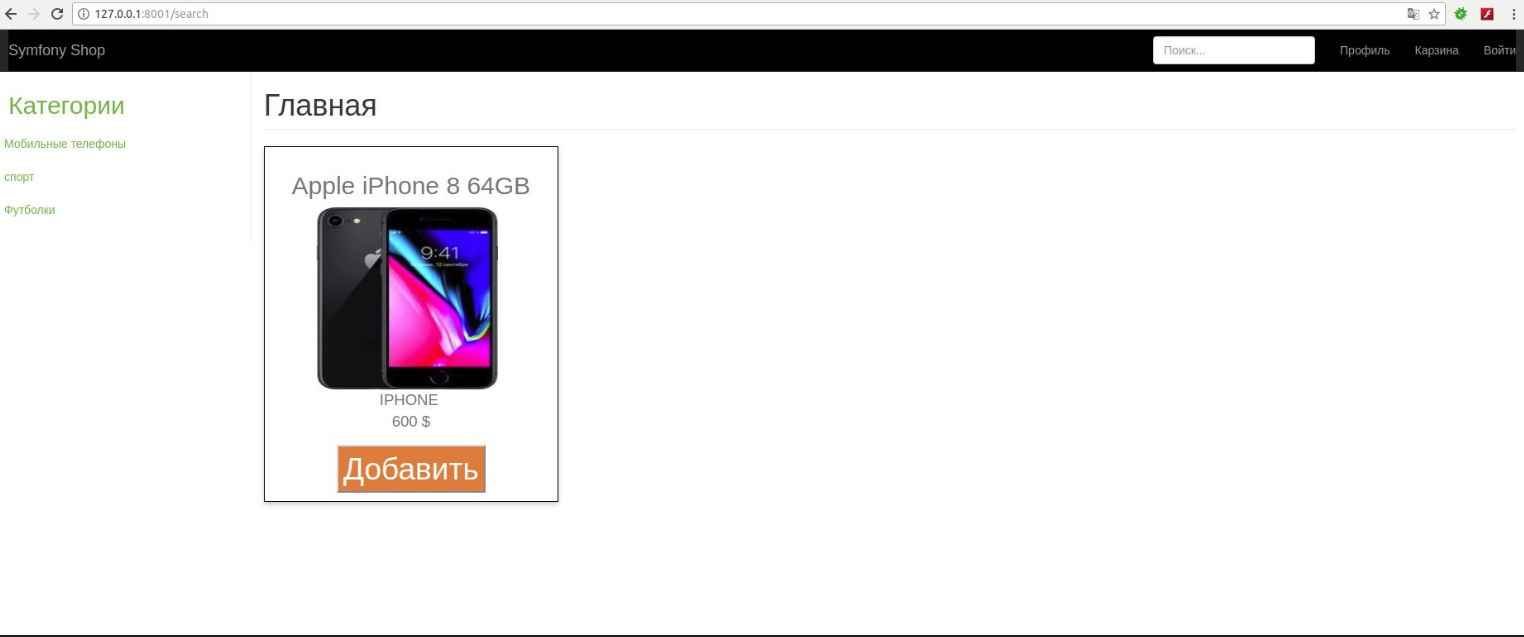


Рисунок 9.20 – Результаты поиска

На этом функционал пользователя заканчивается. Далее рассмотрим функционал администратора (рисунок 9.21). Вход в панель администратора производится по адресу 127.0.0.1:8001/admin/home. Если у пользователя пытающегося войти не хватает прав, то он будет перенаправлен на главную страницу.

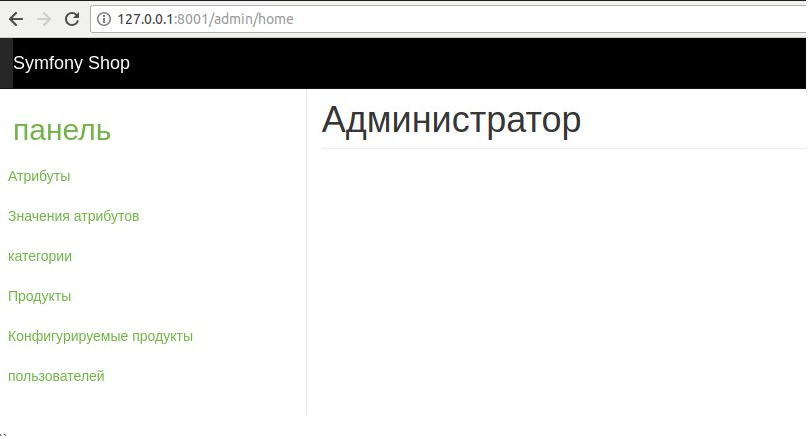


Рисунок 9.21 – Панель администратора

Справа можно выбрать сущность. Выбор “Атрибуты” (рисунок 9.22).

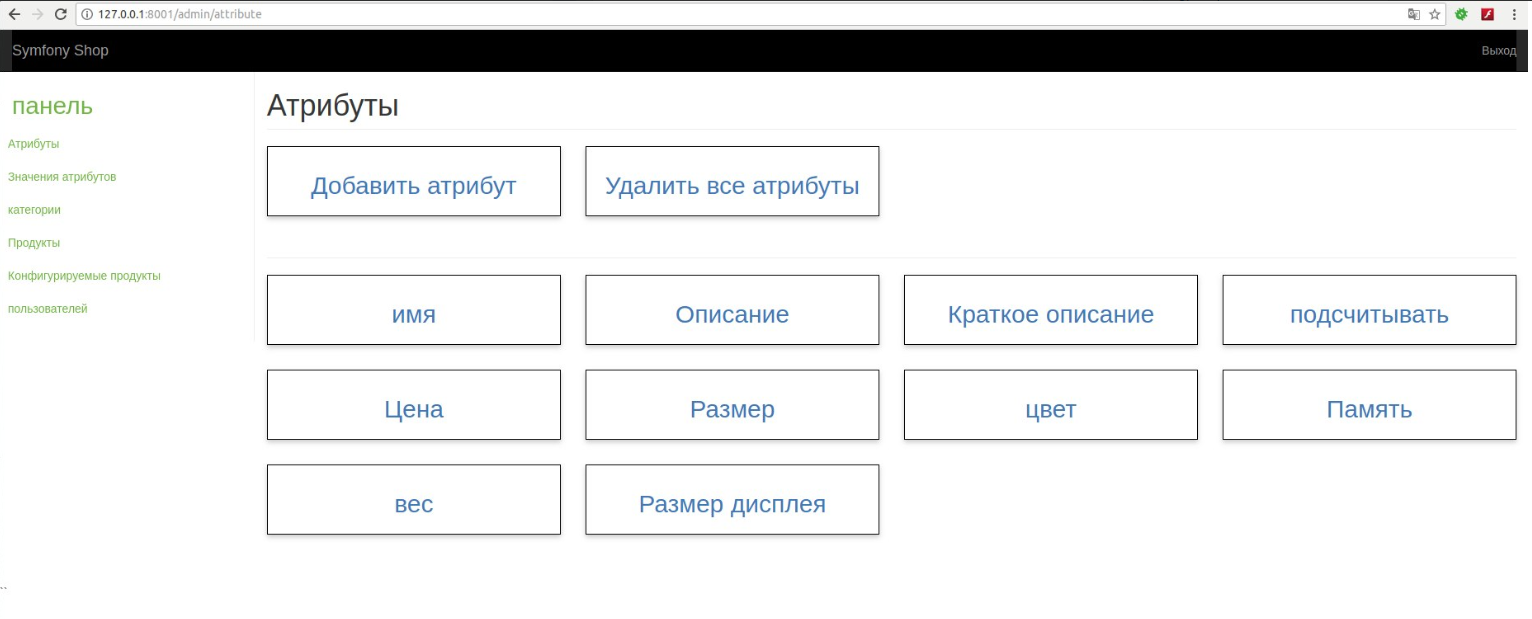


Рисунок 9.22 – Список атрибутов

Администратор может добавить новый атрибут, удалить все или просмотреть, изменить или удалить существующий. Попробуем создать новый атрибут (рисунок 9.23).

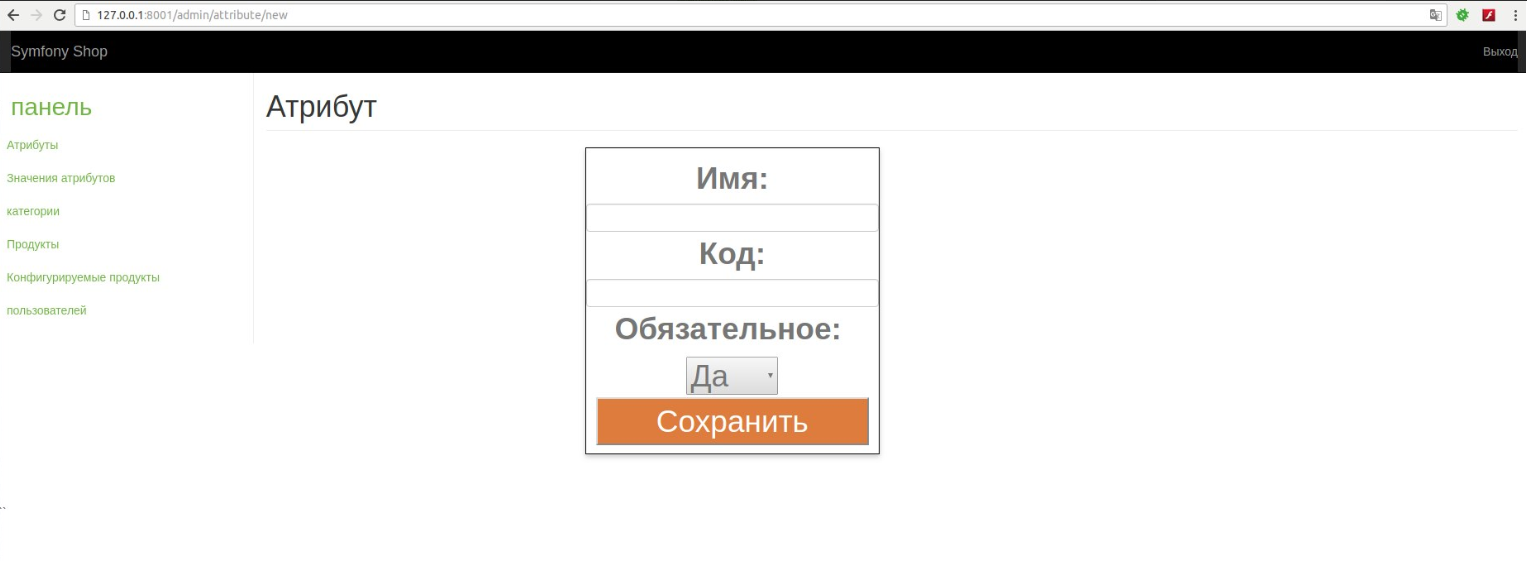


Рисунок 9.23 – Новый атрибут

Создадим атрибут “разработчик”, можно увидеть его на странице всех атрибутов (рисунок 9.24).

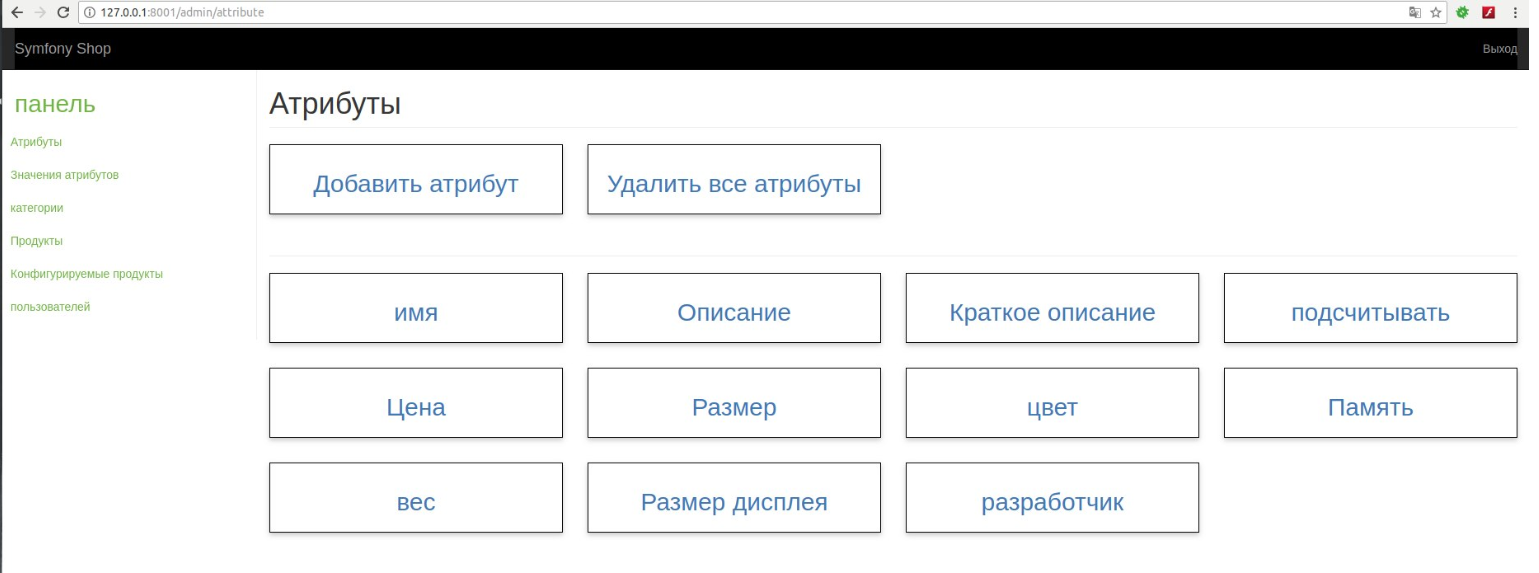


Рисунок 9.24 – Добавлен атрибут

При нажатии на атрибут он открывается для редактирования (рисунок 9.25).

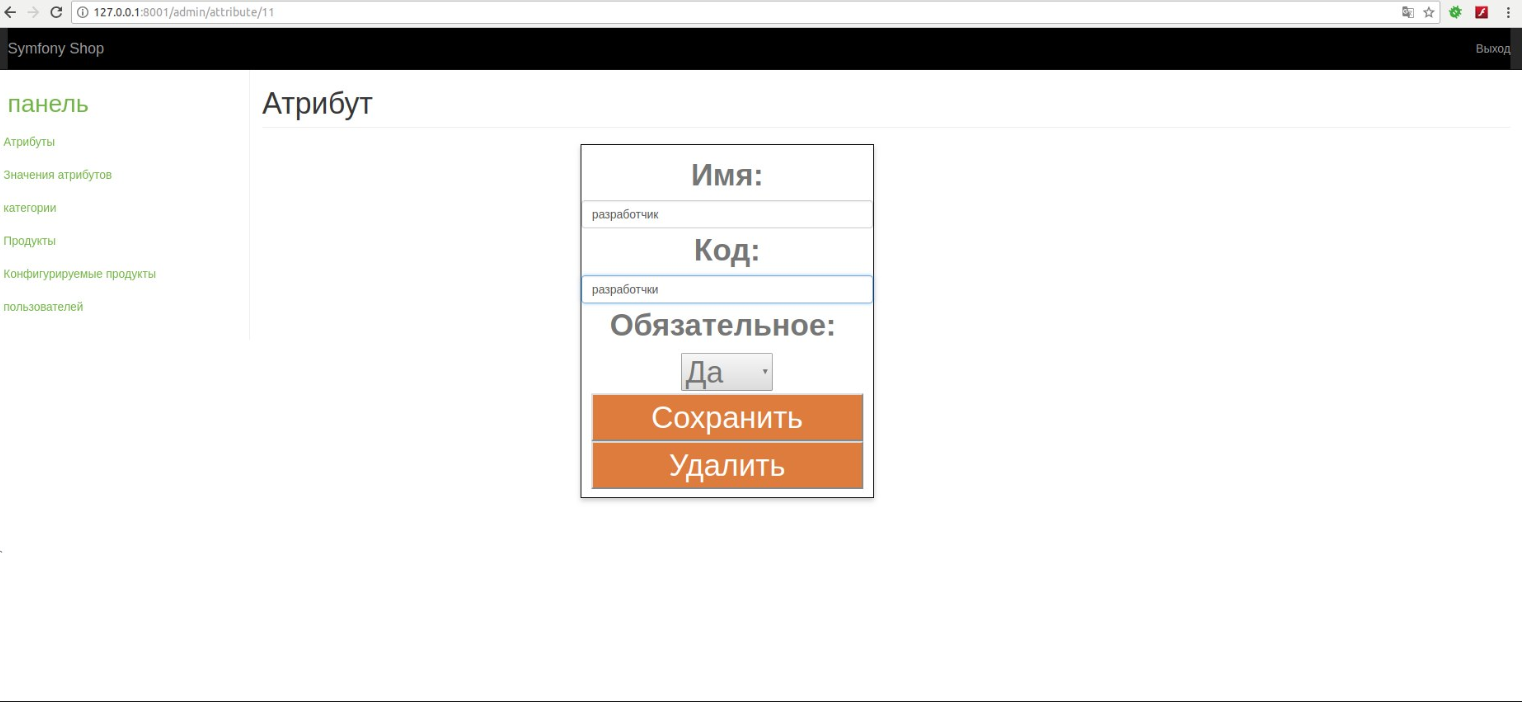


Рисунок 9.25 – Изменение атрибута

Удалим этот атрибут и убедимся, что он пропал с страницы всех продуктов (рисунок 9.26).

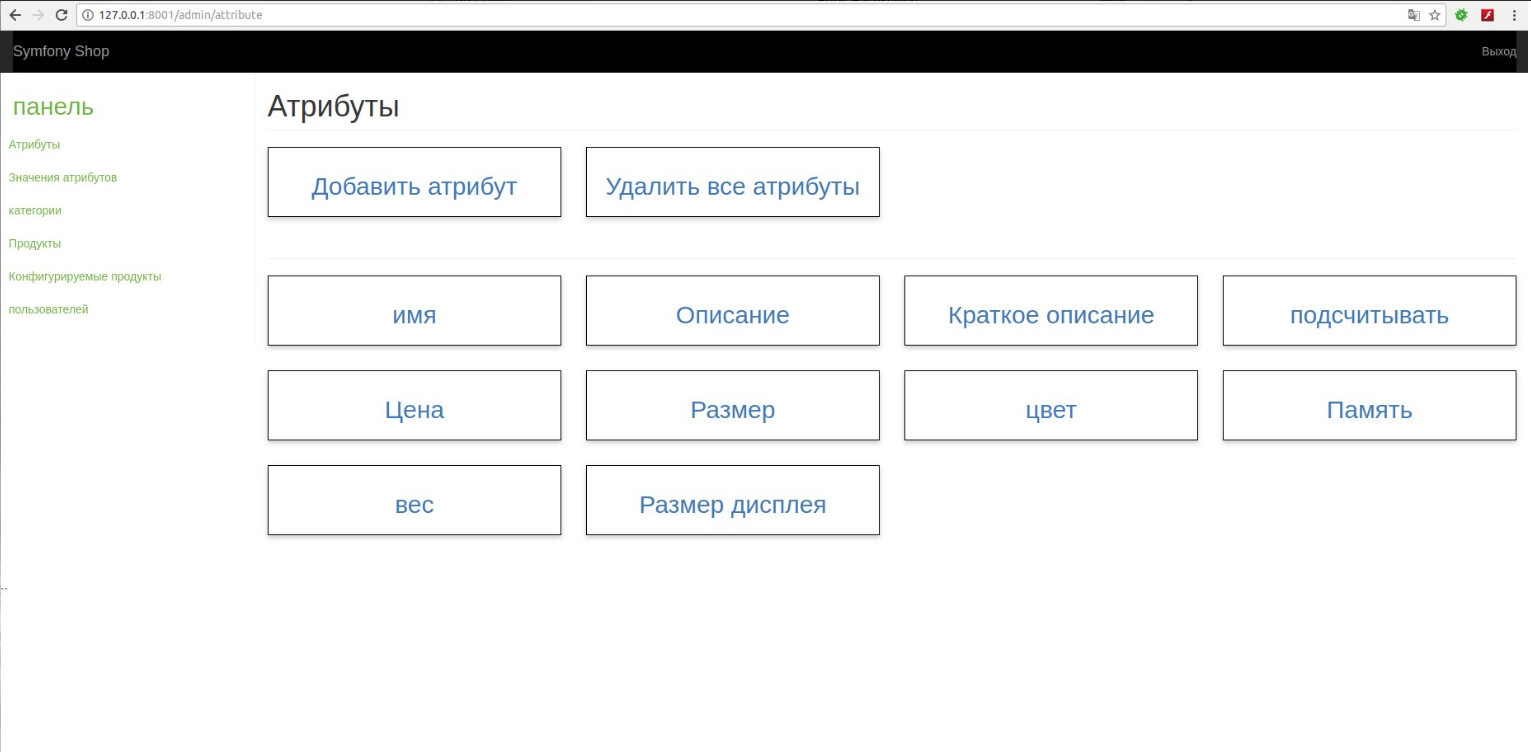


Рисунок 9.26 – Удаленный атрибут

Далее по списку идет “Значения атрибутов”, при нажатии на эту ссылку видно все значения атрибутов (рисунок 9.27).

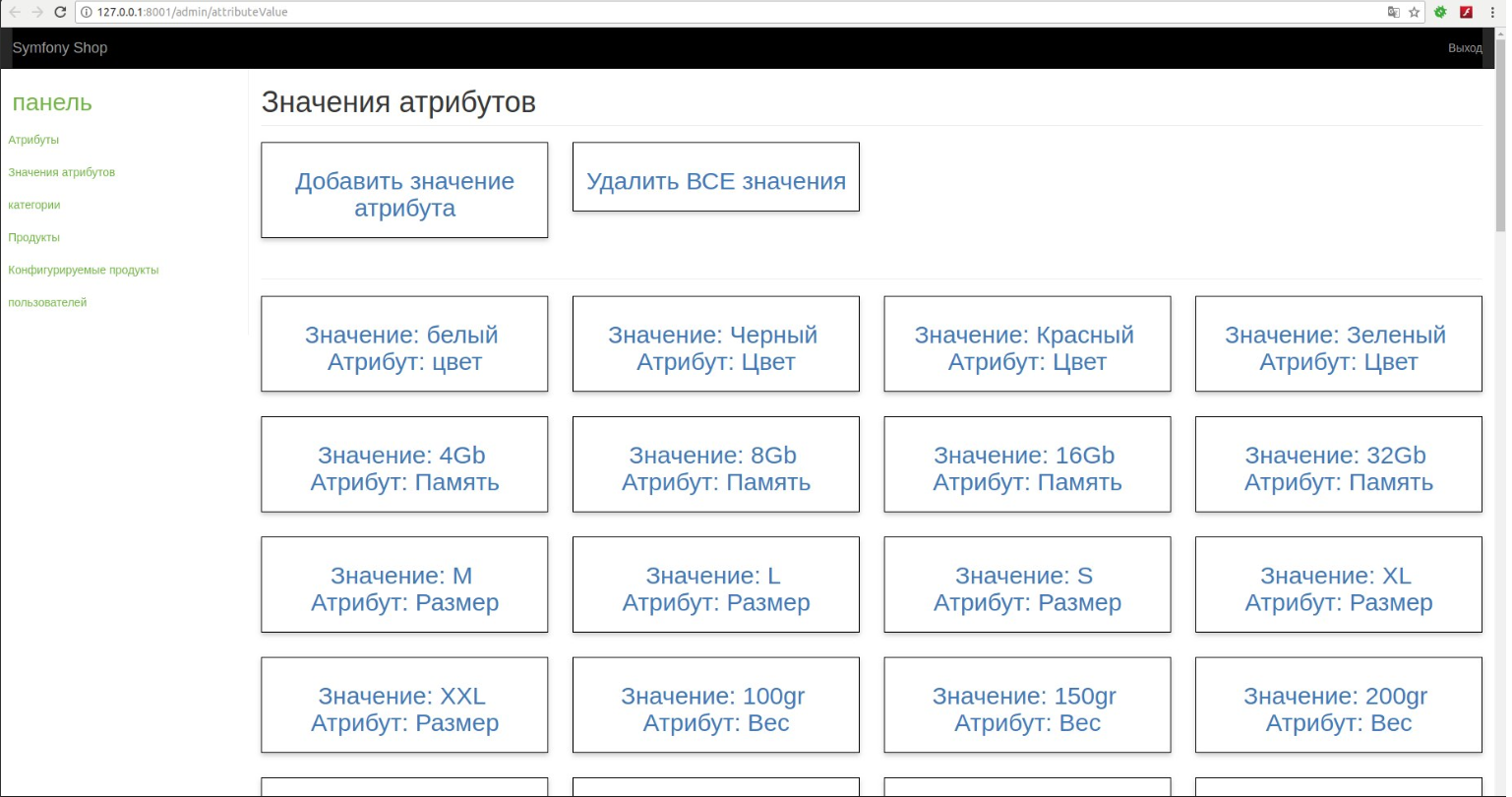


Рисунок 9.27 – Значения атрибутов

Изменим значение первого значения атрибута (рисунок 9.28).

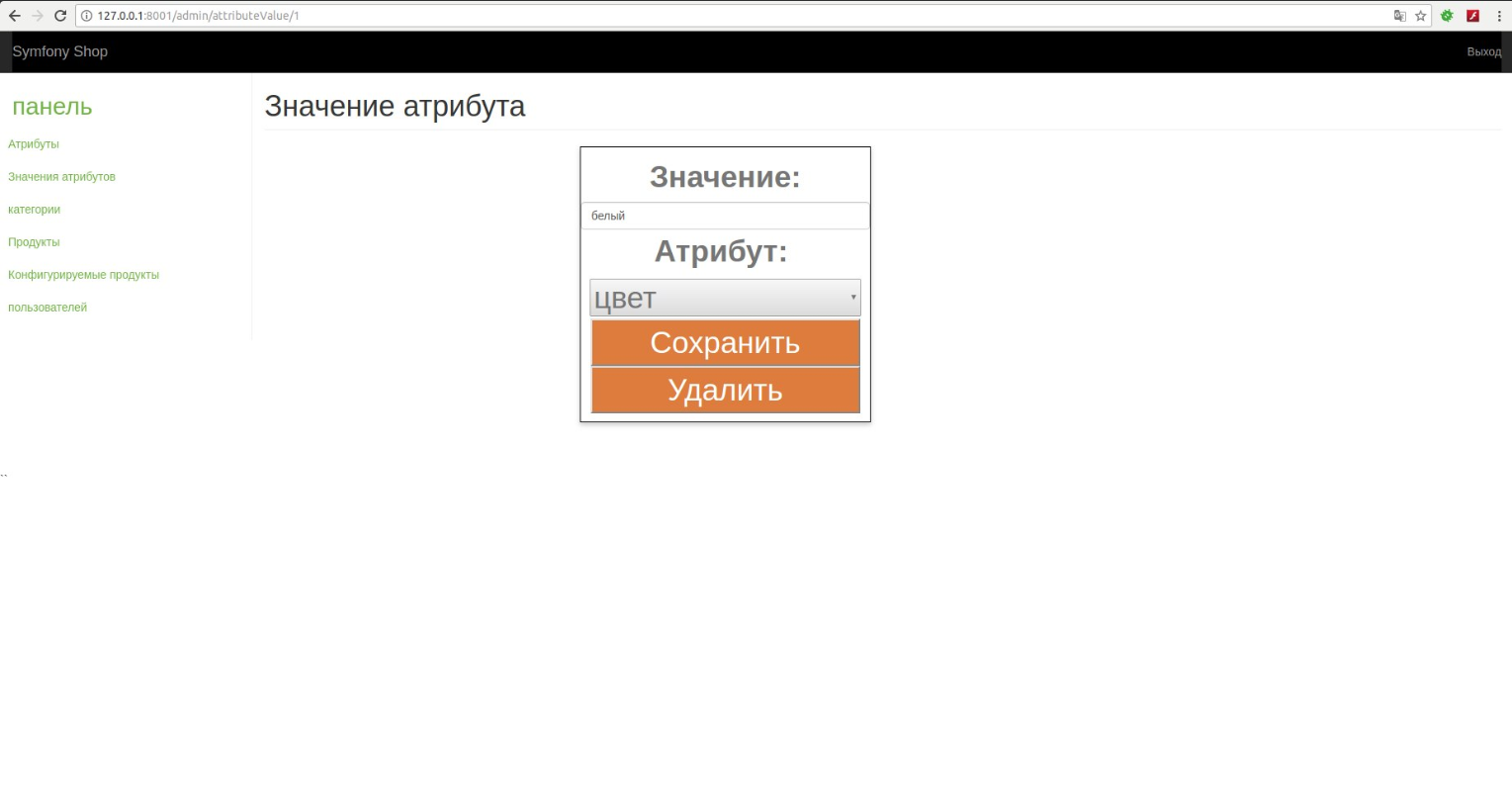


Рисунок 9.28 – Изменение значения атрибута

Результат изменения (рисунок 9.29).

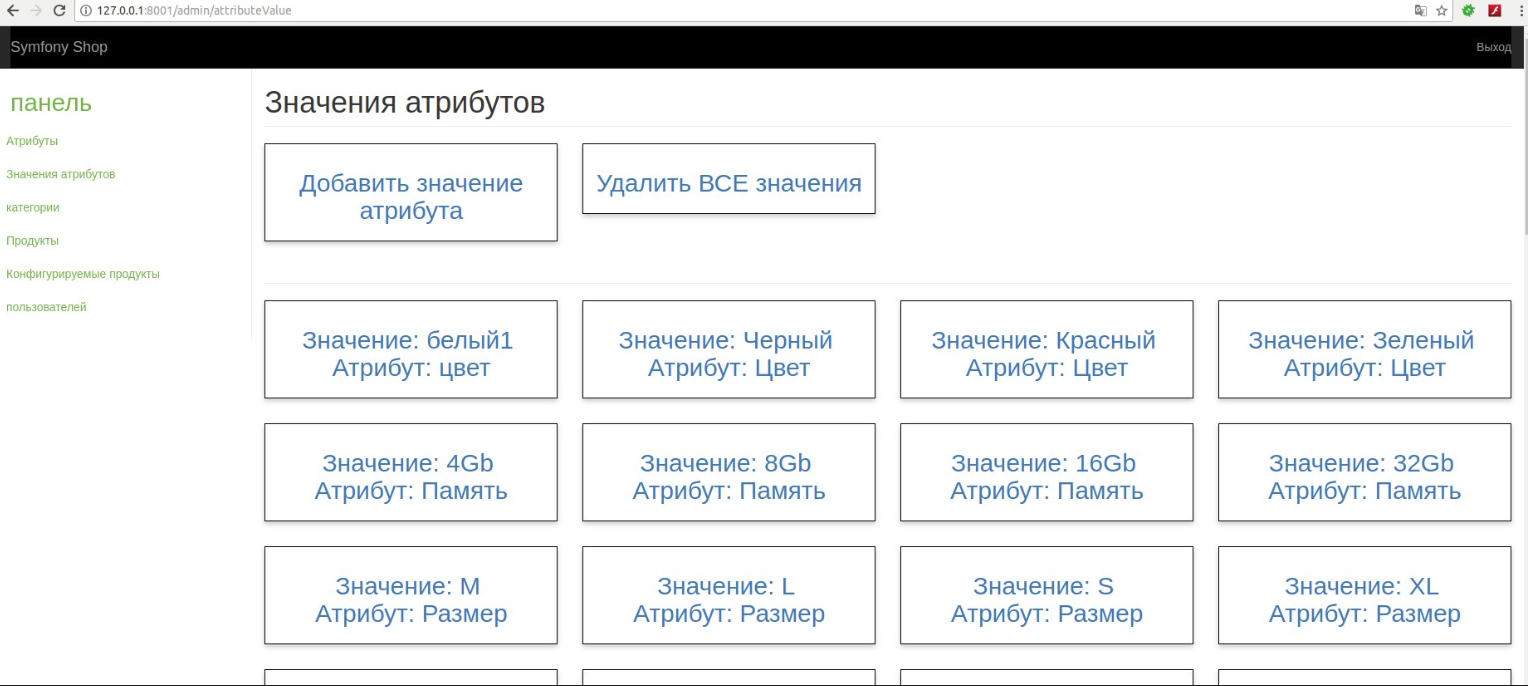


Рисунок 9.29 – Результат изменения

В принципе для категорий и пользователей все действия точно такие же и нету смысла их повторять. Рассмотрим товары (рисунок 9.30).

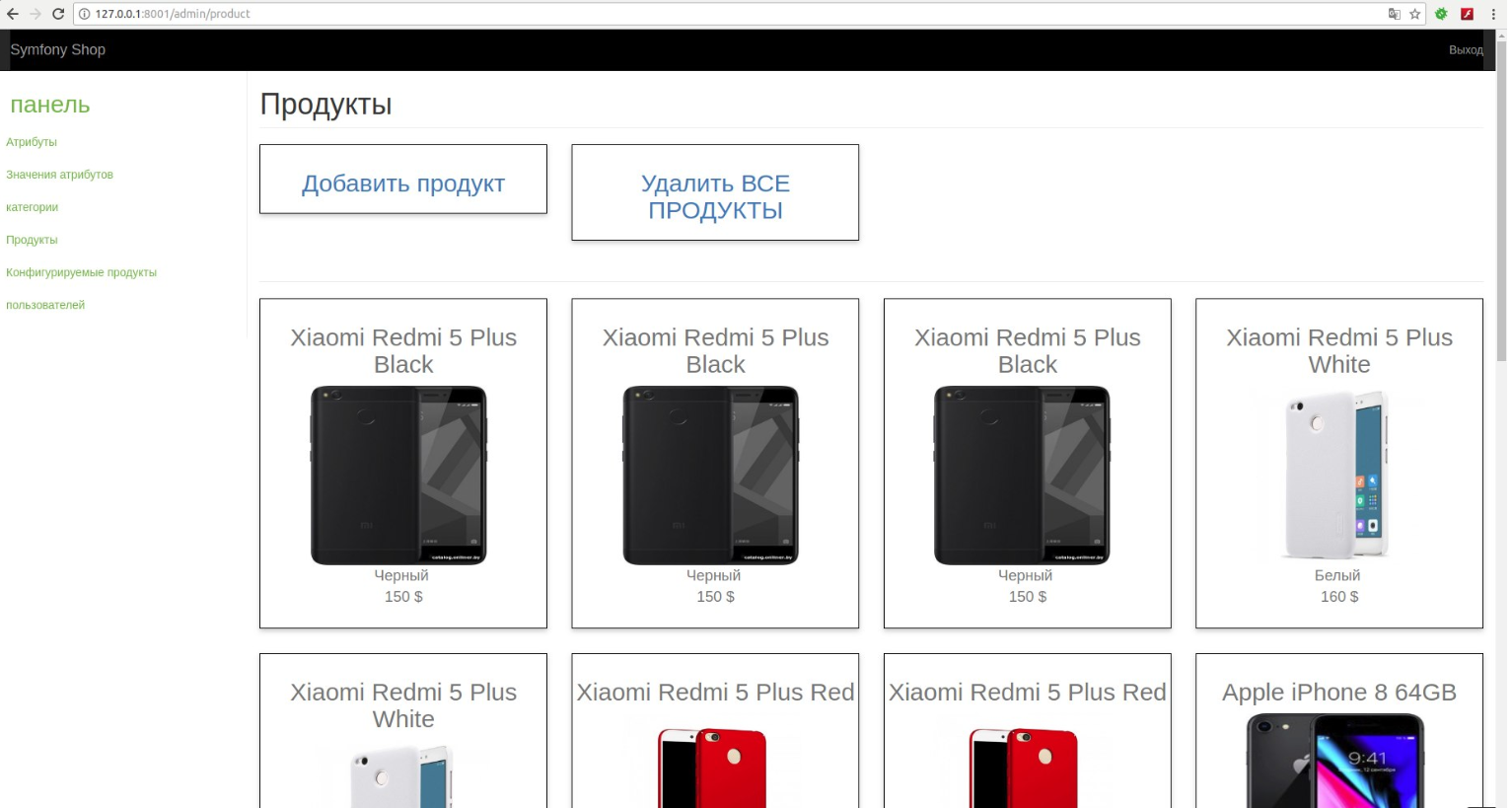


Рисунок 9.30 – Страница товаров

Выбрав простой товар можно увидеть и изменить значения всех обязательных атрибутов. Простые товары содержать только обязательные атрибуты (рисунок 9.31).

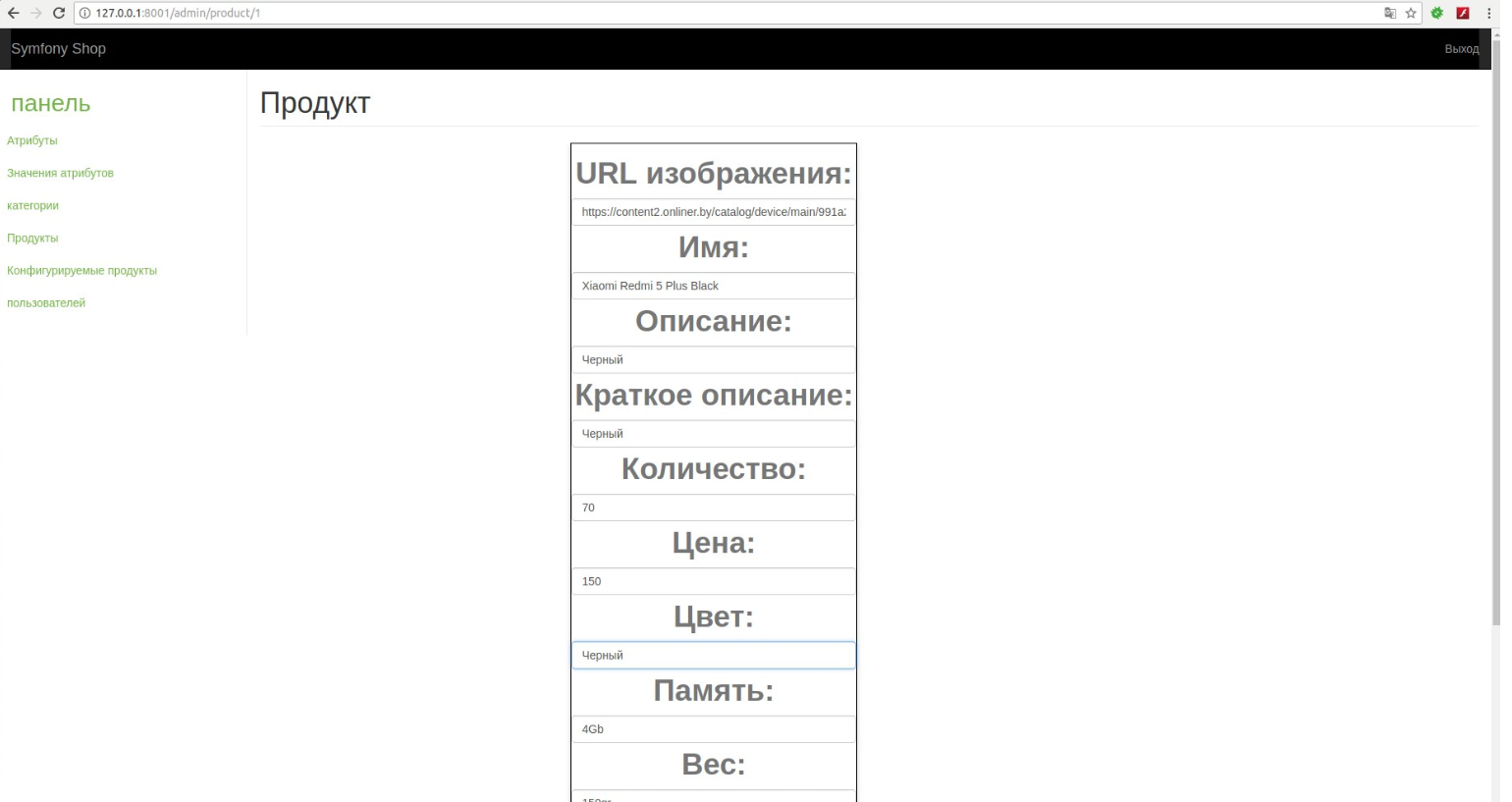


Рисунок 9.31 – Просмотр товара

Перейдем к настраиваемым товарам (рисунок 9.32 – 9.34), с ними все немного сложнее. Настраиваемый товар скажем так задает шаблон по которому можно создавать простые, кроме обязательных атрибутов у настраиваемых товаров есть список чекбоксов с необязательными атрибутами. Выбранные обязательные атрибуты будут присутствовать у простых продуктов принадлежащих этому настраиваемому.

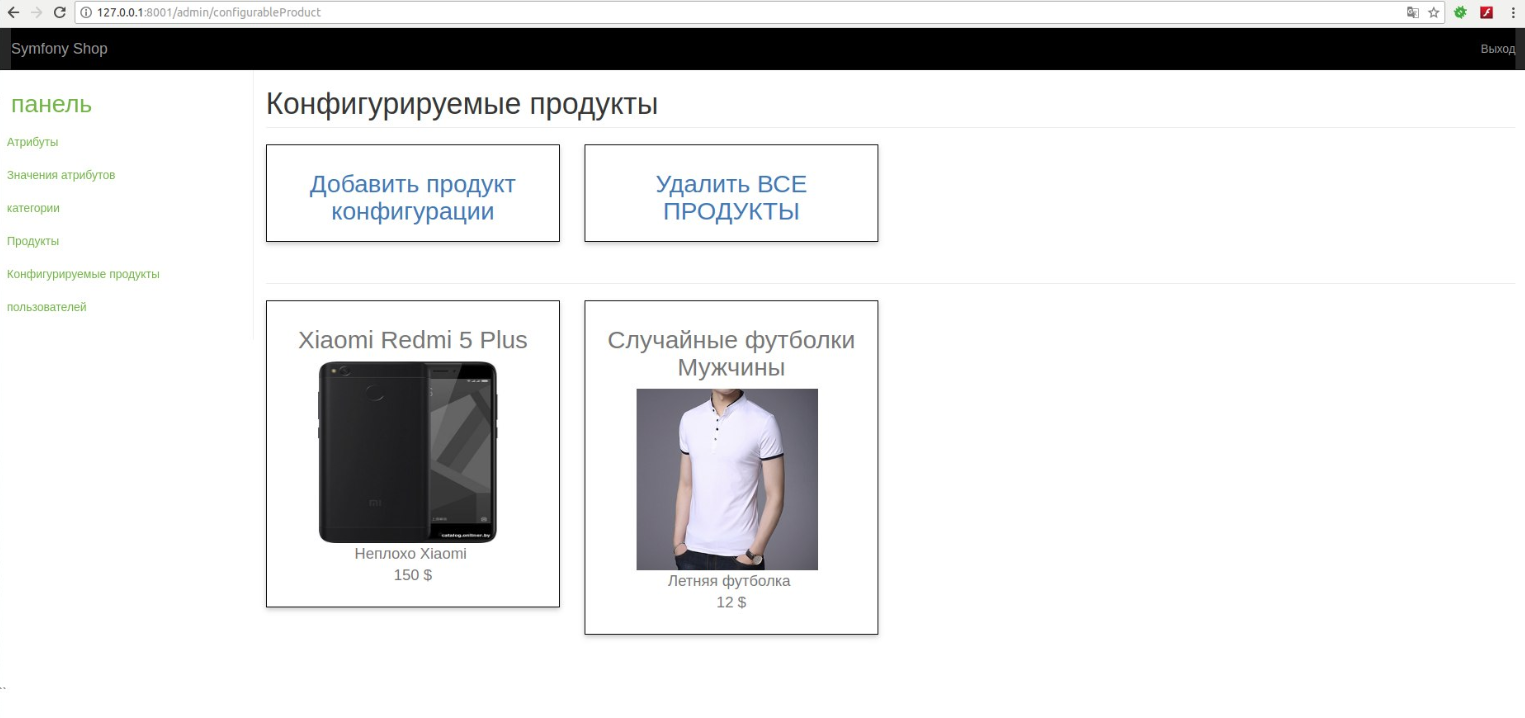


Рисунок 9.32 – Настраиваемые товары

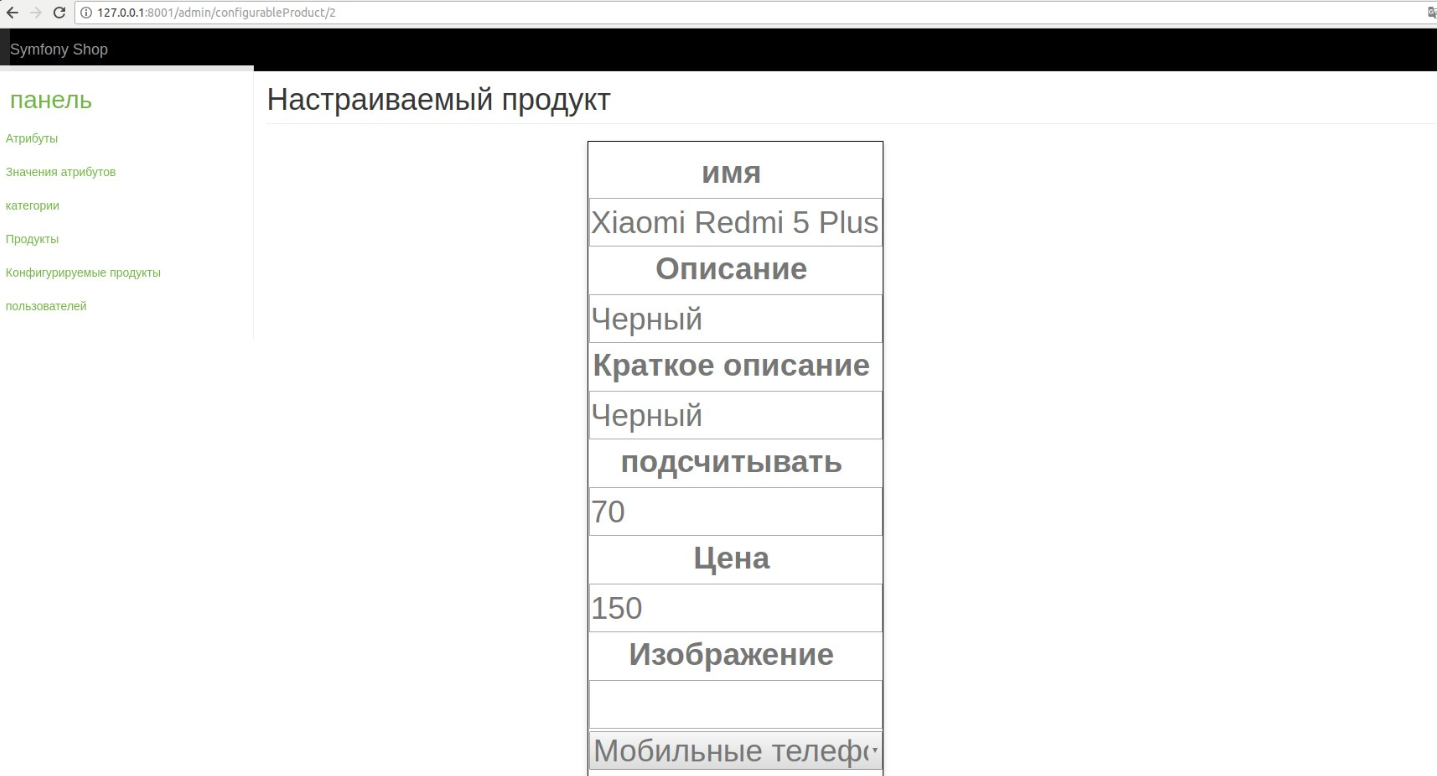


Рисунок 9.33 – Просмотр товара

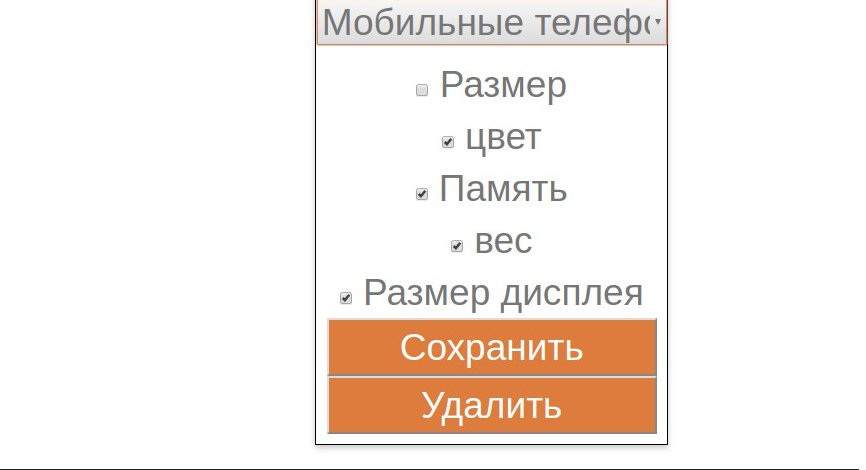


Рисунок 9.34 – необязательные атрибуты

Перечислим задачи, которые были поставлены в начале разработки данного приложения:

1. Изучить данную предметную область и предусмотреть современные тенденции формирования и продвижения интернет магазинов;
2. определить основные сущности базы данных;
3. изучить всевозможные технологии, готовые реализации и подходы к созданию веб-приложений и овладеть современными методологиями проектирования с применением на практике UML-диаграмм;
4. разработать онлайн магазин.

На этой стадии можно уверенно сказать, что в ходе разработки приложения все поставленные задачи были полностью выполнены. Первая глава это подробное описание данной предметной области, также были обнаружены различные проблемы, устраняемые данным приложением. В четвертой главе были разработаны и выбраны сущности базы данных и приведено доказательство приведения их к 3 нормальной форме. 5 раздел включил описание некоторых используемых инструментов и технологий, их преимущество и необходимость использования в данном курсовом проекте. Таким образом, можно уверено сказать, что все поставленные задачи были успешно выполнены и в результате было получено функционирующее веб-приложение, простое и удобное в использовании, совместимое с различными операционными системами. Поэтому и последняя, самая важная и главная задача данного курсового проекта была полностью выполнена.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данного курсового проекта была тщательно и досконально исследована предметная область интернет магазинов, и в результате получено реализованное веб приложение. Это подтверждает, что основные задачи данного курсового проекта были успешно выполнены.

Приложение полностью предусматривает основные заявленные характеристики. Оно достаточно практично и комфортно в повседневной эксплуатации, целостно, охарактеризовано и исполнено в рамках заданной предметной области. Приложение хорошо структурировано из-за четкого виденья целого проекта и разбиения на отдельные небольшие задачи и подзадачи.

Также были построены всевозможные UML-диаграммы для отображения связей между сщностями и взаимодействия друг с другом. Поэтому вторичные задачи проектирования данного проекта так же были достаточно успешно выполнены.

Необходимо заметить, что данная предметная область детально изучена и полностью проанализирована.

Во время разработки этого программного продукта были тщательно предусмотрены и разобраны способы обработки возможных ошибок, критические ситуации, которые могут возникнуть в ходе работы с приложением. Для того, чтобы избежать этого, были обеспечены различные проверки и подсказки об устранении ошибок.

В результате, можно уверенно утверждать, что заданная в начале курсового проекта цель о сокращении затрат времени пользователем на покупку необходимых товаров успешно достигнута.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] PHP [Электронный ресурс]. – Электронные данные. – Режим доступа: http://php.net/manual/ru/book.curl.php

[2] [Sitepoin](http://zametkinapolyah.ru/)t [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.sitepoint.com/symfony2-registration-login/

[3] Habrahabr [Электронный ресурс]. – Электронные данные. – Режим доступа: https://habr.com/post/334446/

[4] Symfony [Электронный ресурс]. – Электронные данные. – Режим доступа: https://symfony.com/doc/current/components/security/authentication

[5] PHPDocu [Электронный ресурс]. – Электронные данные. – Режим доступа: http://php.net/manual/ru/

[6] SQL [Электронный ресурс]. – Электронные данные. – Режим доступа: http://sql-tutorial.ru/ru/book\_subqueries.html/

[7] TWIG [Электронный ресурс]. – Электронные данные. – Режим доступа: https://twig.symfony.com/

[8] Habr [Электронный ресурс]. – Электронные данные. – Режим доступа: https://habr.com/post/145946/

ПРИЛОЖЕНИЕ А

**(обязательное)**

**Модели представления электронной доски объявлений**

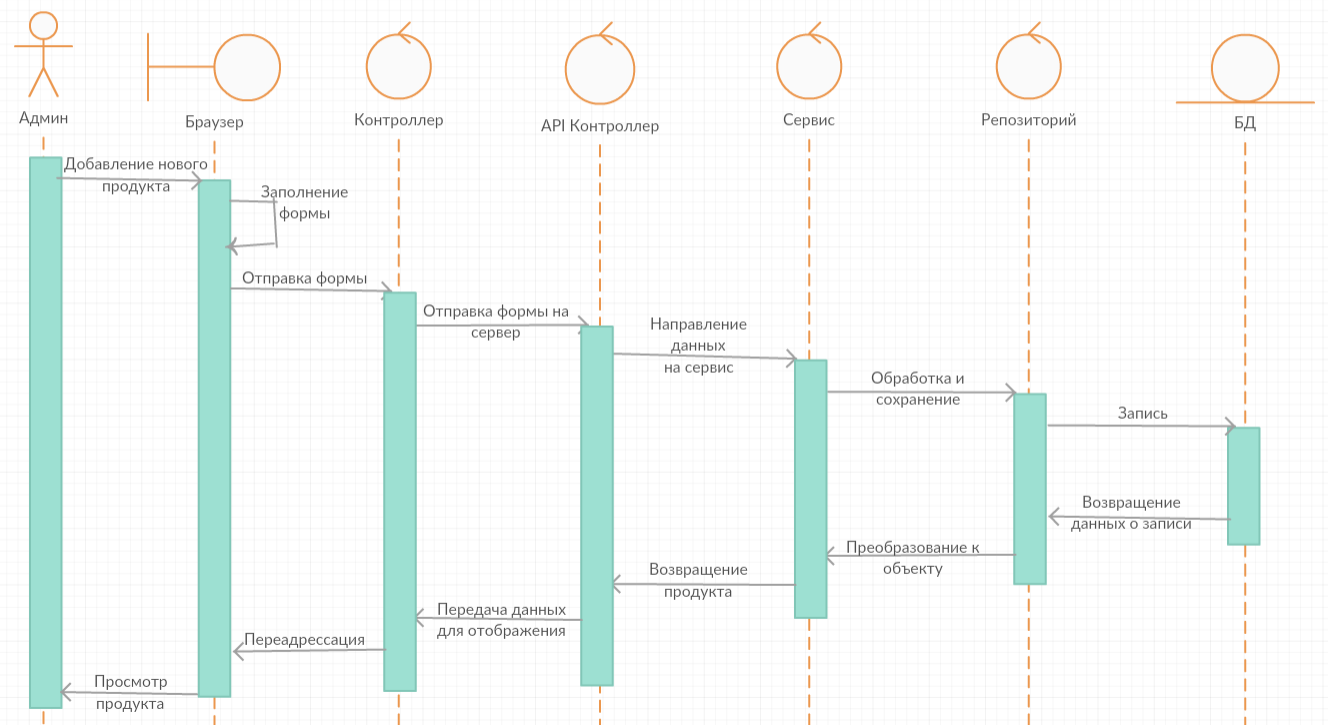


Рисунок А.1 – Диаграмма последовательности добавления нового продукта



Рисунок А.2 – Диаграмма состояний авторизации пользователя

Продолжение Приложения А

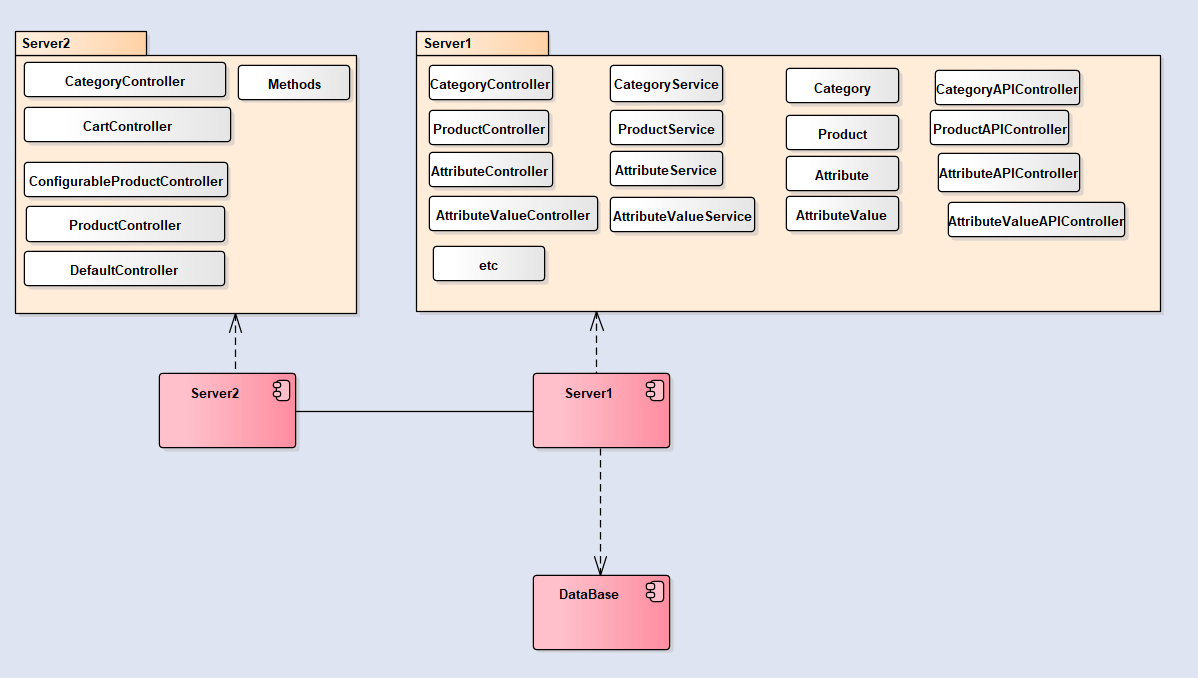


Рисунок А.3 – Диаграмма компонентов модуля базы данных

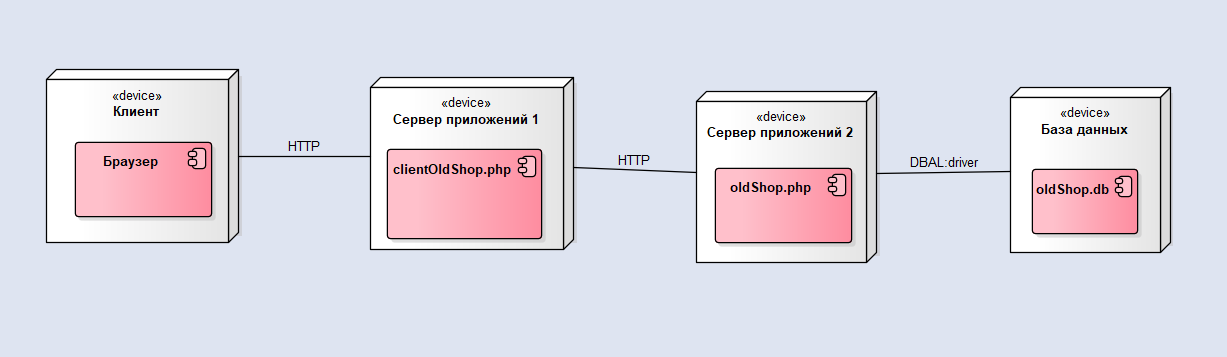


Рисунок А.4 – Диаграмма развертывания

Продолжение Приложения А

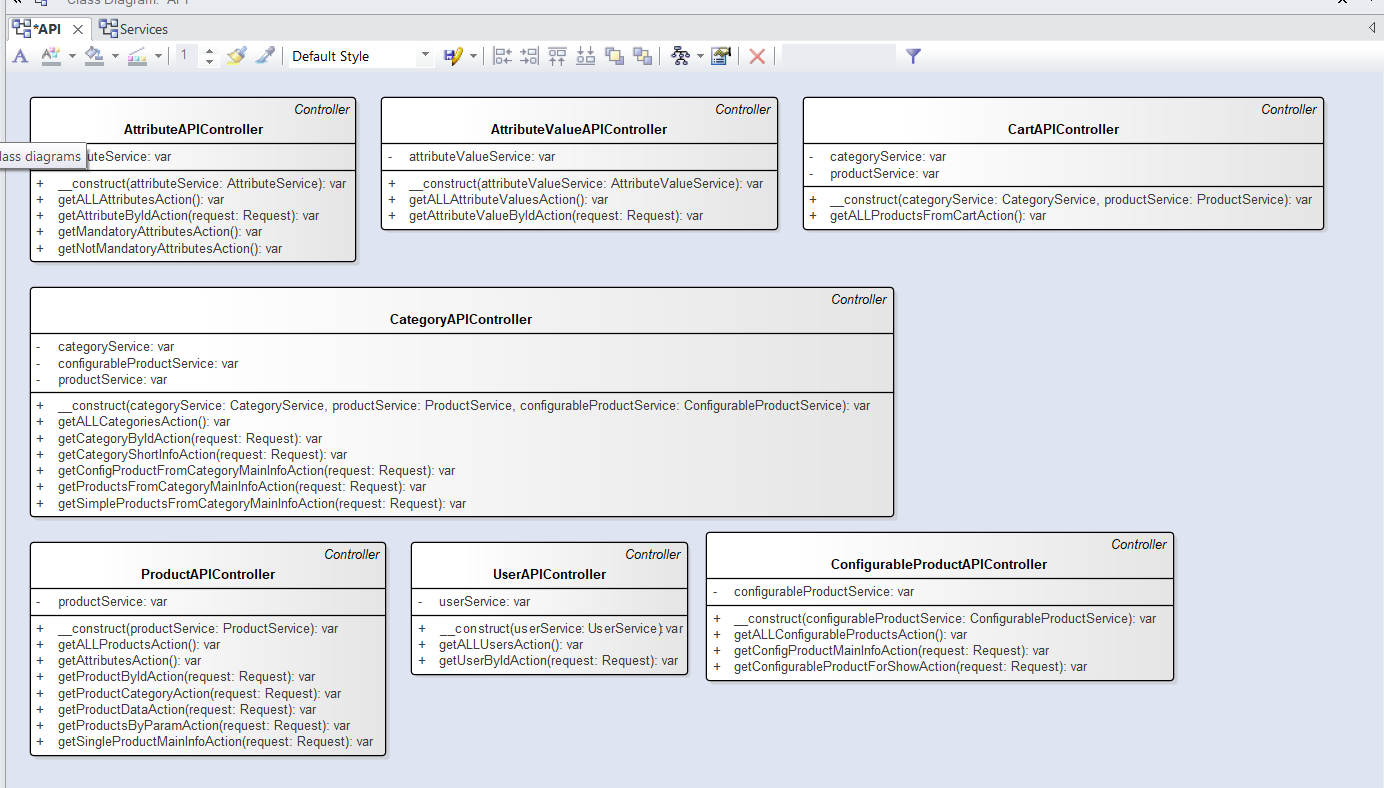


Рисунок А.5 – Диаграмма классов пакета API

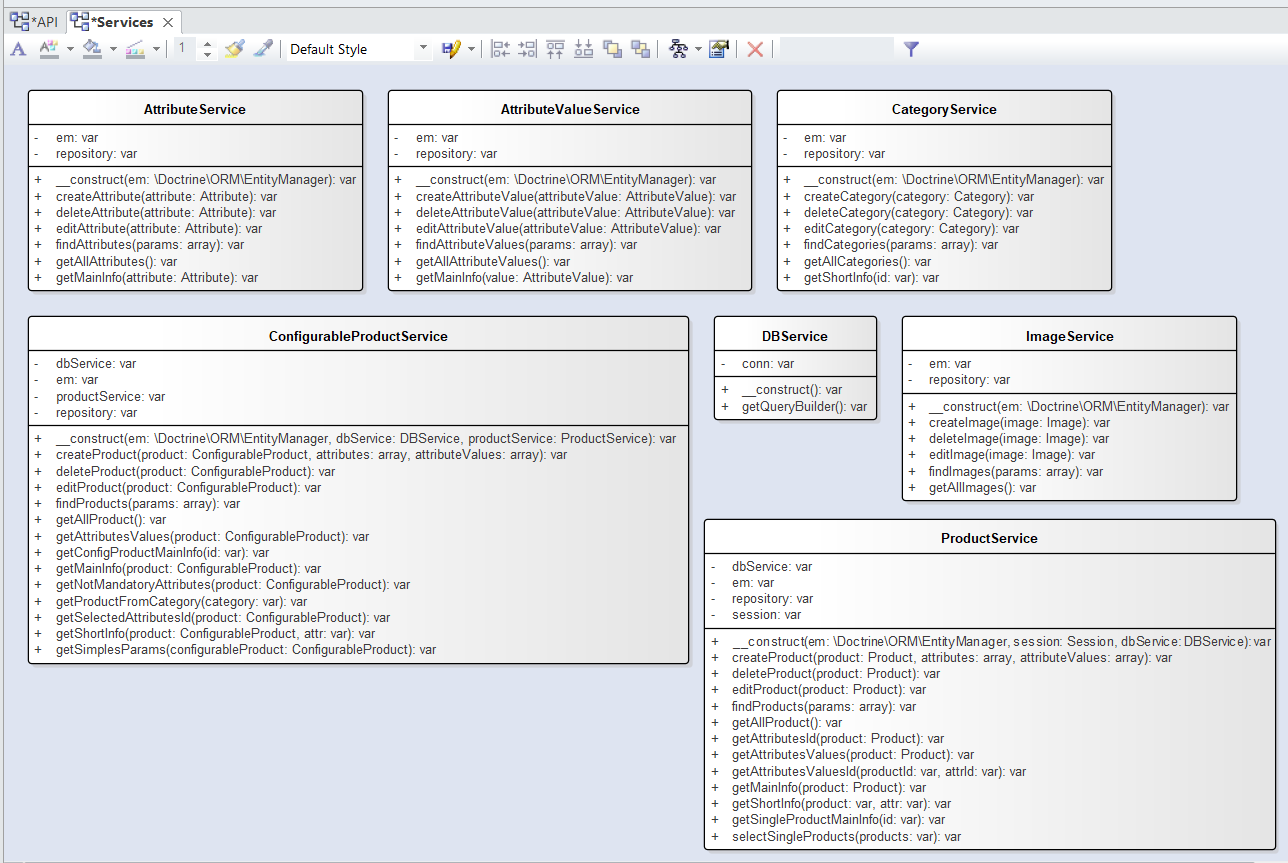


Рисунок А.6 – Диаграмма классов пакета Services

Продолжение Приложения А

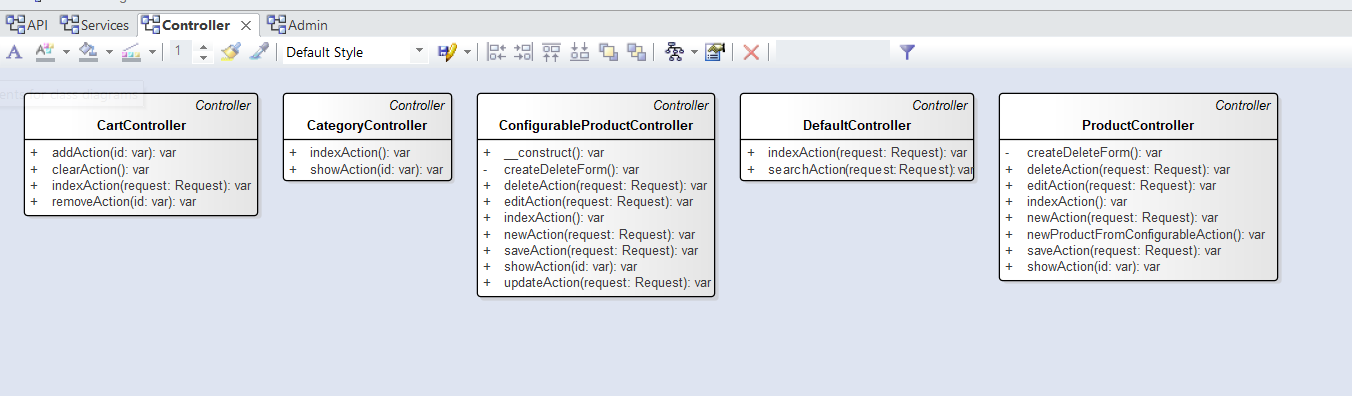


Рисунок А.6 – Диаграмма классов пакета Controller

|  |  |
| --- | --- |
|  |  |

ПРИЛОЖЕНИЕ Б

**(обязательное)**

**Блок-схемы методов, реализующих бизнес-логику**

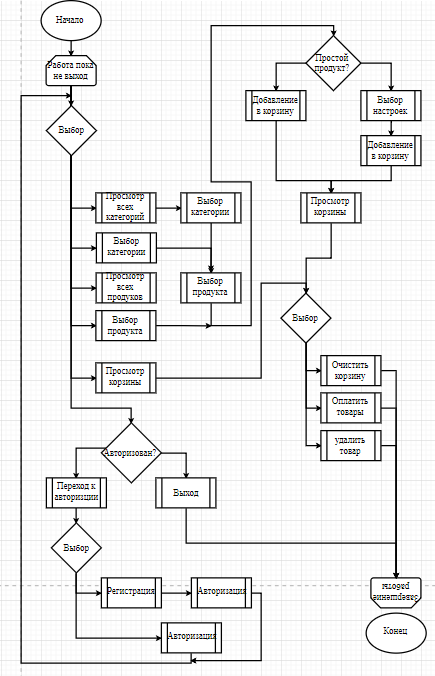


Рисунок Б.1 – Схема работы всей программы

Продолжение Приложения Б

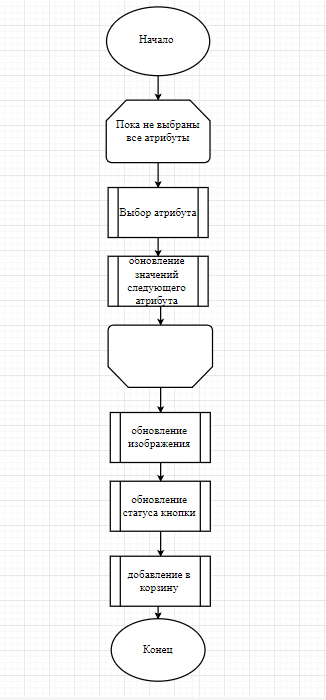


Рисунок Б.2 – Схема добавления в корзину настраиваемого продукта

ПРИЛОЖЕНИЕ В

**(обязательное)**

**Листинг программы**

// Скрипт создания базы данных

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- База данных: `oldShop`

--

-- --------------------------------------------------------

--

-- Структура таблицы `attribute`

--

CREATE TABLE `attribute` (

`id` int(11) NOT NULL,

`сode` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL,

`name` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL,

`mandatory` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `attribute`

--

INSERT INTO `attribute` (`id`, `сode`, `name`, `mandatory`) VALUES

(1, 'Name', 'Name', 1),

(2, 'Description', 'Description', 1),

(3, 'ShortDescription', 'Short description', 1),

(4, 'Count', 'Count', 1),

(5, 'Price', 'Price', 1),

(6, 'Size', 'Size', 0),

(7, 'Color', 'Color', 0),

(8, 'Memory', 'Memory', 0),

(9, 'Weight', 'Weight', 0),

(10, 'Display Size', 'Display Size', 0);

-- --------------------------------------------------------

--

-- Структура таблицы `attribute\_value`

--

CREATE TABLE `attribute\_value` (

`id` int(11) NOT NULL,

`attribute\_id` int(11) DEFAULT NULL,

`value` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `attribute\_value`

--

INSERT INTO `attribute\_value` (`id`, `attribute\_id`, `value`) VALUES

(1, 7, 'White'),

(2, 7, 'Black'),

(3, 7, 'Red'),

(4, 7, 'Green'),

(5, 8, '4Gb'),

(6, 8, '8Gb'),

(7, 8, '16Gb'),

(8, 8, '32Gb'),

(9, 6, 'M'),

(10, 6, 'L'),

(11, 6, 'S'),

(12, 6, 'XL'),

(13, 6, 'XXL'),

(14, 9, '100gr'),

(15, 9, '150gr'),

(16, 9, '200gr'),

(17, 9, '250gr'),

(18, 10, '5'),

(19, 10, '5.5'),

(20, 10, '6'),

(21, 1, 'Xiaomi Redmi 5 Plus'),

(22, 2, ' Qualcomm Snapdragon 625 MSM8953'),

(23, 3, 'Not bad Xiaomi'),

(24, 4, '100'),

(25, 5, '150'),

(26, 1, 'Xiaomi Redmi 5 Plus Black'),

(27, 2, 'Black'),

(28, 3, 'Black'),

(29, 4, '70'),

(30, 1, 'Xiaomi Redmi 5 Plus White'),

(31, 2, 'White classic'),

(32, 3, 'White'),

(33, 4, '10'),

(34, 5, '160'),

(35, 1, 'Xiaomi Redmi 5 Plus Red'),

(36, 2, 'Red'),

(37, 3, 'Red'),

(38, 5, '180'),

(39, 1, 'Apple iPhone 8 64GB'),

(40, 2, 'IPHONE'),

(41, 3, 'IPHONE'),

(42, 4, '20'),

(43, 5, '600'),

(44, 1, 'Huawei Mate 10 Lite'),

(45, 2, 'Huawei'),

(46, 3, 'HUAWEI'),

(47, 4, '80'),

(48, 5, '170'),

(49, 1, 'Soccer ball'),

(50, 2, 'Hot Sale professional soccer ball standard Size 5 PU leather genuine seamless training football for children and adults'),

(51, 3, 'soccer ball'),

(52, 4, '15'),

(53, 5, '30'),

(54, 1, 'Casual T-Shirt Men'),

(55, 2, 'ENWAYEL Summer Fashion Solid Cotton Casual T Shirt Men Mandarin Collar Short Sleeve Tshirt Mens Top Tee Patchwork T-Shirt Male'),

(56, 3, 'Summer T-Shirt '),

(57, 4, '79'),

(58, 5, '12'),

(59, 1, 'White T-Shirt'),

(60, 2, 'Some white T-Shirt'),

(61, 4, '12'),

(62, 5, '14'),

(63, 1, 'Black T-Shirt'),

(64, 2, 'Some black T-Shirt'),

(65, 4, '27'),

(66, 1, 'Red T-Shirt'),

(67, 2, 'Some red T-Shirt');

-- --------------------------------------------------------

--

-- Структура таблицы `category`

--

CREATE TABLE `category` (

`id` int(11) NOT NULL,

`name` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `category`

--

INSERT INTO `category` (`id`, `name`) VALUES

(4, 'cars'),

(1, 'Mobile Phones'),

(3, 'Sport'),

(2, 'T-Shirts');

-- --------------------------------------------------------

--

-- Структура таблицы `configurable\_product`

--

CREATE TABLE `configurable\_product` (

`id` int(11) NOT NULL,

`category\_id` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `configurable\_product`

--

INSERT INTO `configurable\_product` (`id`, `category\_id`) VALUES

(1, 1),

(2, 2);

-- --------------------------------------------------------

--

-- Структура таблицы `configurable\_product\_attribute`

--

CREATE TABLE `configurable\_product\_attribute` (

`configurable\_product\_id` int(11) NOT NULL,

`attribute\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `configurable\_product\_attribute`

--

INSERT INTO `configurable\_product\_attribute` (`configurable\_product\_id`, `attribute\_id`) VALUES

(1, 1),

(1, 2),

(1, 3),

(1, 4),

(1, 5),

(1, 7),

(1, 8),

(1, 9),

(1, 10),

(2, 1),

(2, 2),

(2, 3),

(2, 4),

(2, 5),

(2, 6),

(2, 7);

-- --------------------------------------------------------

--

-- Структура таблицы `configurable\_product\_attribute\_value`

--

CREATE TABLE `configurable\_product\_attribute\_value` (

`id` int(11) NOT NULL,

`product\_id` int(11) NOT NULL,

`attribute\_id` int(11) NOT NULL,

`value\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Дамп данных таблицы `configurable\_product\_attribute\_value`

--

INSERT INTO `configurable\_product\_attribute\_value` (`id`, `product\_id`, `attribute\_id`, `value\_id`) VALUES

(1, 1, 1, 21),

(2, 1, 2, 22),

(3, 1, 3, 23),

(4, 1, 4, 24),

(5, 1, 5, 25),

(6, 2, 1, 54),

(7, 2, 2, 55),

(8, 2, 3, 56),

(9, 2, 4, 57),

(10, 2, 5, 58);

-- --------------------------------------------------------

--

-- Структура таблицы `configurable\_product\_product\_order`

--

CREATE TABLE `configurable\_product\_product\_order` (

`configurable\_product\_id` int(11) NOT NULL,

`product\_order\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

-- --------------------------------------------------------

--

-- Структура таблицы `image`

--

CREATE TABLE `image` (

`id` int(11) NOT NULL,

`product\_id` int(11) DEFAULT NULL,

`configurable\_product\_id` int(11) DEFAULT NULL,

`url` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `image`

--

INSERT INTO `image` (`id`, `product\_id`, `configurable\_product\_id`, `url`) VALUES

(1, NULL, 1, 'https://content2.onliner.by/catalog/device/main/991a26aae0e6619107f2a644d4469a4f.jpeg'),

(2, 1, NULL, 'https://content2.onliner.by/catalog/device/main/991a26aae0e6619107f2a644d4469a4f.jpeg'),

(3, 2, NULL, 'https://content2.onliner.by/catalog/device/main/991a26aae0e6619107f2a644d4469a4f.jpeg'),

(4, 3, NULL, 'https://content2.onliner.by/catalog/device/main/991a26aae0e6619107f2a644d4469a4f.jpeg'),

(5, 4, NULL, 'http://digital812.su/media/catalog/product/cache/1/image/650x/040ec09b1e35df139433887a97daa66f/f/r/frostedredmi4xwhite.jpg'),

(6, 5, NULL, 'http://digital812.su/media/catalog/product/cache/1/image/650x/040ec09b1e35df139433887a97daa66f/f/r/frostedredmi4xwhite.jpg'),

(7, 6, NULL, 'http://img.dxcdn.com/productimages/sku\_471657\_1.jpg'),

(8, 7, NULL, 'http://img.dxcdn.com/productimages/sku\_471657\_1.jpg'),

(9, 8, NULL, 'https://content2.onliner.by/catalog/device/header/58cc3f28bed5a3789a8d59414b79c580.jpeg'),

(10, 9, NULL, 'https://content2.onliner.by/catalog/device/header/c3e2b364b63e368c6d0a3b81d8e512b0.jpeg'),

(11, 10, NULL, 'https://ae01.alicdn.com/kf/HTB1KH.fRFXXXXb7XVXXq6xXFXXXC/Hot-Sale-professional-soccer-ball-standard-Size-5-PU-leather-genuine-seamless-training-football-for-children.jpg\_640x640.jpg'),

(12, NULL, 2, 'http://ae01.alicdn.com/kf/HTB1.vwZXiOYBuNjSsD4q6zSkFXaR.jpg'),

(13, 11, NULL, 'http://ae01.alicdn.com/kf/HTB1.vwZXiOYBuNjSsD4q6zSkFXaR.jpg'),

(14, 12, NULL, 'http://ae01.alicdn.com/kf/HTB1.vwZXiOYBuNjSsD4q6zSkFXaR.jpg'),

(15, 13, NULL, 'http://ae01.alicdn.com/kf/HTB1.vwZXiOYBuNjSsD4q6zSkFXaR.jpg'),

(16, 14, NULL, 'http://ae01.alicdn.com/kf/HTB1fJDzXMaTBuNjSszfq6xgfpXag.jpg'),

(17, 15, NULL, 'http://ae01.alicdn.com/kf/HTB1fJDzXMaTBuNjSszfq6xgfpXag.jpg'),

(18, 16, NULL, 'http://ae01.alicdn.com/kf/HTB1fJDzXMaTBuNjSszfq6xgfpXag.jpg'),

(19, 17, NULL, 'http://ae01.alicdn.com/kf/HTB1fJDzXMaTBuNjSszfq6xgfpXag.jpg'),

(20, 18, NULL, 'http://ae01.alicdn.com/kf/HTB1OY8aX.R1BeNjy0Fmq6z0wVXaV.jpg'),

(21, 19, NULL, 'http://ae01.alicdn.com/kf/HTB1OY8aX.R1BeNjy0Fmq6z0wVXaV.jpg');

-- --------------------------------------------------------

--

-- Структура таблицы `product`

--

CREATE TABLE `product` (

`id` int(11) NOT NULL,

`category\_id` int(11) DEFAULT NULL,

`configurable\_product\_id` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `product`

--

INSERT INTO `product` (`id`, `category\_id`, `configurable\_product\_id`) VALUES

(1, 1, 1),

(2, 1, 1),

(3, 1, 1),

(4, 1, 1),

(5, 1, 1),

(6, 1, 1),

(7, 1, 1),

(8, 1, NULL),

(9, 1, NULL),

(10, 3, NULL),

(11, 2, 2),

(12, 2, 2),

(13, 2, 2),

(14, 2, 2),

(15, 2, 2),

(16, 2, 2),

(17, 2, 2),

(18, 2, 2),

(19, 2, 2);

-- --------------------------------------------------------

--

-- Структура таблицы `product\_attribute\_value`

--

CREATE TABLE `product\_attribute\_value` (

`id` int(11) NOT NULL,

`product\_id` int(11) NOT NULL,

`attribute\_id` int(11) NOT NULL,

`value\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Дамп данных таблицы `product\_attribute\_value`

--

INSERT INTO `product\_attribute\_value` (`id`, `product\_id`, `attribute\_id`, `value\_id`) VALUES

(1, 1, 1, 26),

(2, 1, 2, 27),

(3, 1, 3, 28),

(4, 1, 4, 29),

(5, 1, 5, 25),

(6, 1, 7, 2),

(7, 1, 8, 5),

(8, 1, 9, 15),

(9, 1, 10, 18),

(10, 2, 1, 26),

(11, 2, 2, 27),

(12, 2, 3, 28),

(13, 2, 4, 29),

(14, 2, 5, 25),

(15, 2, 7, 2),

(16, 2, 8, 6),

(17, 2, 9, 15),

(18, 2, 10, 18),

(19, 3, 1, 26),

(20, 3, 2, 27),

(21, 3, 3, 28),

(22, 3, 4, 29),

(23, 3, 5, 25),

(24, 3, 7, 2),

(25, 3, 8, 7),

(26, 3, 9, 15),

(27, 3, 10, 18),

(28, 4, 1, 30),

(29, 4, 2, 31),

(30, 4, 3, 32),

(31, 4, 4, 33),

(32, 4, 5, 34),

(33, 4, 7, 1),

(34, 4, 8, 5),

(35, 4, 9, 15),

(36, 4, 10, 18),

(37, 5, 1, 30),

(38, 5, 2, 31),

(39, 5, 3, 32),

(40, 5, 4, 33),

(41, 5, 5, 34),

(42, 5, 7, 1),

(43, 5, 8, 6),

(44, 5, 9, 15),

(45, 5, 10, 18),

(46, 6, 1, 35),

(47, 6, 2, 36),

(48, 6, 3, 37),

(49, 6, 4, 33),

(50, 6, 5, 38),

(51, 6, 7, 3),

(52, 6, 8, 6),

(53, 6, 9, 15),

(54, 6, 10, 18),

(55, 7, 1, 35),

(56, 7, 2, 36),

(57, 7, 3, 37),

(58, 7, 4, 33),

(59, 7, 5, 38),

(60, 7, 7, 3),

(61, 7, 8, 8),

(62, 7, 9, 15),

(63, 7, 10, 18),

(64, 8, 1, 39),

(65, 8, 2, 40),

(66, 8, 3, 41),

(67, 8, 4, 42),

(68, 8, 5, 43),

(69, 9, 1, 44),

(70, 9, 2, 45),

(71, 9, 3, 46),

(72, 9, 4, 47),

(73, 9, 5, 48),

(74, 10, 1, 49),

(75, 10, 2, 50),

(76, 10, 3, 51),

(77, 10, 4, 52),

(78, 10, 5, 53),

(79, 11, 1, 59),

(80, 11, 2, 60),

(81, 11, 3, 32),

(82, 11, 4, 61),

(83, 11, 5, 62),

(84, 11, 6, 9),

(85, 11, 7, 1),

(86, 12, 1, 59),

(87, 12, 2, 60),

(88, 12, 3, 32),

(89, 12, 4, 61),

(90, 12, 5, 62),

(91, 12, 6, 10),

(92, 12, 7, 1),

(93, 13, 1, 59),

(94, 13, 2, 60),

(95, 13, 3, 32),

(96, 13, 4, 61),

(97, 13, 5, 62),

(98, 13, 6, 12),

(99, 13, 7, 1),

(100, 14, 1, 63),

(101, 14, 2, 64),

(102, 14, 3, 28),

(103, 14, 4, 65),

(104, 14, 5, 58),

(105, 14, 6, 9),

(106, 14, 7, 2),

(107, 15, 1, 63),

(108, 15, 2, 64),

(109, 15, 3, 28),

(110, 15, 4, 65),

(111, 15, 5, 58),

(112, 15, 6, 10),

(113, 15, 7, 2),

(114, 16, 1, 63),

(115, 16, 2, 64),

(116, 16, 3, 28),

(117, 16, 4, 65),

(118, 16, 5, 58),

(119, 16, 6, 12),

(120, 16, 7, 2),

(121, 17, 1, 63),

(122, 17, 2, 64),

(123, 17, 3, 28),

(124, 17, 4, 65),

(125, 17, 5, 58),

(126, 17, 6, 13),

(127, 17, 7, 2),

(128, 18, 1, 66),

(129, 18, 2, 67),

(130, 18, 3, 37),

(131, 18, 4, 65),

(132, 18, 5, 58),

(133, 18, 6, 11),

(134, 18, 7, 3),

(135, 19, 1, 66),

(136, 19, 2, 67),

(137, 19, 3, 37),

(138, 19, 4, 65),

(139, 19, 5, 58),

(140, 19, 6, 12),

(141, 19, 7, 3);

-- --------------------------------------------------------

--

-- Структура таблицы `product\_order`

--

CREATE TABLE `product\_order` (

`id` int(11) NOT NULL,

`users\_id` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

-- --------------------------------------------------------

--

-- Структура таблицы `product\_product\_order`

--

CREATE TABLE `product\_product\_order` (

`product\_id` int(11) NOT NULL,

`product\_order\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

-- --------------------------------------------------------

--

-- Структура таблицы `user`

--

CREATE TABLE `user` (

`id` int(11) NOT NULL,

`username` varchar(15) COLLATE utf8\_unicode\_ci NOT NULL,

`password` varchar(500) COLLATE utf8\_unicode\_ci NOT NULL,

`email` varchar(50) COLLATE utf8\_unicode\_ci NOT NULL,

`phoneNumber` int(11) NOT NULL,

`role` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Дамп данных таблицы `user`

--

INSERT INTO `user` (`id`, `username`, `password`, `email`, `phoneNumber`, `role`) VALUES

(2, 'user', '$2y$13$5NjPyGkkLPyA3EV2LeKXb.cSCwfbr12MDfhoXFiahbo0UERnB.Y3O', 'user@mail.ru', 12345698, 'user'),

(3, 'zdarova', '$2y$13$2ZkGFDEhvI3/GUkkjs/riet5lkgeklOk2jBD0H3a3Yyh5djSwY8ne', 'zdarova@mail.ru', 777777777, 'user'),

(4, 'zzz', '$2y$13$Ay8nPVh9HIzP3yToMmO3T.VbX8QY6NjFvaIG0eS8PkrmAR1CXyXnO', 'zzz@gmail.com', 1237767, 'user'),

(5, 'Boris', '$2y$13$3TZvW5kKV20KmkN0xi8t4uSR7ld6uuZisoKA3Ora/CIUFMKFsb.ta', 'zzz@mail.ru', 1234567, 'user');

--

-- Индексы сохранённых таблиц

--

--

-- Индексы таблицы `attribute`

--

ALTER TABLE `attribute`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `UNIQ\_FA7AEFFB5890C7C3` (`сode`);

--

-- Индексы таблицы `attribute\_value`

--

ALTER TABLE `attribute\_value`

ADD PRIMARY KEY (`id`),

ADD KEY `IDX\_FE4FBB82B6E62EFA` (`attribute\_id`);

--

-- Индексы таблицы `category`

--

ALTER TABLE `category`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `UNIQ\_64C19C15E237E06` (`name`);

--

-- Индексы таблицы `configurable\_product`

--

ALTER TABLE `configurable\_product`

ADD PRIMARY KEY (`id`),

ADD KEY `IDX\_A07DD80212469DE2` (`category\_id`);

--

-- Индексы таблицы `configurable\_product\_attribute`

--

ALTER TABLE `configurable\_product\_attribute`

ADD PRIMARY KEY (`configurable\_product\_id`,`attribute\_id`),

ADD KEY `IDX\_6C322AD52905368` (`configurable\_product\_id`),

ADD KEY `IDX\_6C322AD5B6E62EFA` (`attribute\_id`);

--

-- Индексы таблицы `configurable\_product\_attribute\_value`

--

ALTER TABLE `configurable\_product\_attribute\_value`

ADD PRIMARY KEY (`id`);

--

-- Индексы таблицы `configurable\_product\_product\_order`

--

ALTER TABLE `configurable\_product\_product\_order`

ADD PRIMARY KEY (`configurable\_product\_id`,`product\_order\_id`),

ADD KEY `IDX\_A230971E2905368` (`configurable\_product\_id`),

ADD KEY `IDX\_A230971E462F07AF` (`product\_order\_id`);

--

-- Индексы таблицы `image`

--

ALTER TABLE `image`

ADD PRIMARY KEY (`id`),

ADD KEY `IDX\_C53D045F4584665A` (`product\_id`),

ADD KEY `IDX\_C53D045F2905368` (`configurable\_product\_id`);

--

-- Индексы таблицы `product`

--

ALTER TABLE `product`

ADD PRIMARY KEY (`id`),

ADD KEY `IDX\_D34A04AD12469DE2` (`category\_id`),

ADD KEY `IDX\_D34A04AD2905368` (`configurable\_product\_id`);

--

-- Индексы таблицы `product\_attribute\_value`

--

ALTER TABLE `product\_attribute\_value`

ADD PRIMARY KEY (`id`);

--

-- Индексы таблицы `product\_order`

--

ALTER TABLE `product\_order`

ADD PRIMARY KEY (`id`),

ADD KEY `IDX\_5475E8C467B3B43D` (`users\_id`);

--

-- Индексы таблицы `product\_product\_order`

--

ALTER TABLE `product\_product\_order`

ADD PRIMARY KEY (`product\_id`,`product\_order\_id`),

ADD KEY `IDX\_8E98F4364584665A` (`product\_id`),

ADD KEY `IDX\_8E98F436462F07AF` (`product\_order\_id`);

--

-- Индексы таблицы `user`

--

ALTER TABLE `user`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `UNIQ\_8D93D649F85E0677` (`username`),

ADD UNIQUE KEY `UNIQ\_8D93D649E7927C74` (`email`),

ADD UNIQUE KEY `UNIQ\_8D93D649E85E83E4` (`phoneNumber`);

--

-- AUTO\_INCREMENT для сохранённых таблиц

--

--

-- AUTO\_INCREMENT для таблицы `attribute`

--

ALTER TABLE `attribute`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=11;

--

-- AUTO\_INCREMENT для таблицы `attribute\_value`

--

ALTER TABLE `attribute\_value`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=68;

--

-- AUTO\_INCREMENT для таблицы `category`

--

ALTER TABLE `category`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=5;

--

-- AUTO\_INCREMENT для таблицы `configurable\_product`

--

ALTER TABLE `configurable\_product`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=3;

--

-- AUTO\_INCREMENT для таблицы `configurable\_product\_attribute\_value`

--

ALTER TABLE `configurable\_product\_attribute\_value`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=11;

--

-- AUTO\_INCREMENT для таблицы `image`

--

ALTER TABLE `image`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=22;

--

-- AUTO\_INCREMENT для таблицы `product`

--

ALTER TABLE `product`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=20;

--

-- AUTO\_INCREMENT для таблицы `product\_attribute\_value`

--

ALTER TABLE `product\_attribute\_value`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=142;

--

-- AUTO\_INCREMENT для таблицы `product\_order`

--

ALTER TABLE `product\_order`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT для таблицы `user`

--

ALTER TABLE `user`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=6;

--

-- Ограничения внешнего ключа сохраненных таблиц

--

--

-- Ограничения внешнего ключа таблицы `attribute\_value`

--

ALTER TABLE `attribute\_value`

ADD CONSTRAINT `FK\_FE4FBB82B6E62EFA` FOREIGN KEY (`attribute\_id`) REFERENCES `attribute` (`id`);

--

-- Ограничения внешнего ключа таблицы `configurable\_product`

--

ALTER TABLE `configurable\_product`

ADD CONSTRAINT `FK\_A07DD80212469DE2` FOREIGN KEY (`category\_id`) REFERENCES `category` (`id`);

--

-- Ограничения внешнего ключа таблицы `configurable\_product\_attribute`

--

ALTER TABLE `configurable\_product\_attribute`

ADD CONSTRAINT `FK\_6C322AD52905368` FOREIGN KEY (`configurable\_product\_id`) REFERENCES `configurable\_product` (`id`) ON DELETE CASCADE,

ADD CONSTRAINT `FK\_6C322AD5B6E62EFA` FOREIGN KEY (`attribute\_id`) REFERENCES `attribute` (`id`) ON DELETE CASCADE;

--

-- Ограничения внешнего ключа таблицы `configurable\_product\_product\_order`

--

ALTER TABLE `configurable\_product\_product\_order`

ADD CONSTRAINT `FK\_A230971E2905368` FOREIGN KEY (`configurable\_product\_id`) REFERENCES `configurable\_product` (`id`) ON DELETE CASCADE,

ADD CONSTRAINT `FK\_A230971E462F07AF` FOREIGN KEY (`product\_order\_id`) REFERENCES `product\_order` (`id`) ON DELETE CASCADE;

--

-- Ограничения внешнего ключа таблицы `image`

--

ALTER TABLE `image`

ADD CONSTRAINT `FK\_C53D045F2905368` FOREIGN KEY (`configurable\_product\_id`) REFERENCES `configurable\_product` (`id`),

ADD CONSTRAINT `FK\_C53D045F4584665A` FOREIGN KEY (`product\_id`) REFERENCES `product` (`id`);

--

-- Ограничения внешнего ключа таблицы `product`

--

ALTER TABLE `product`

ADD CONSTRAINT `FK\_D34A04AD12469DE2` FOREIGN KEY (`category\_id`) REFERENCES `category` (`id`),

ADD CONSTRAINT `FK\_D34A04AD2905368` FOREIGN KEY (`configurable\_product\_id`) REFERENCES `configurable\_product` (`id`);

--

-- Ограничения внешнего ключа таблицы `product\_order`

--

ALTER TABLE `product\_order`

ADD CONSTRAINT `FK\_5475E8C467B3B43D` FOREIGN KEY (`users\_id`) REFERENCES `user` (`id`);

--

-- Ограничения внешнего ключа таблицы `product\_product\_order`

--

ALTER TABLE `product\_product\_order`

ADD CONSTRAINT `FK\_8E98F4364584665A` FOREIGN KEY (`product\_id`) REFERENCES `product` (`id`) ON DELETE CASCADE,

ADD CONSTRAINT `FK\_8E98F436462F07AF` FOREIGN KEY (`product\_order\_id`) REFERENCES `product\_order` (`id`) ON DELETE CASCADE;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

ПРИЛОЖЕНИЕ Г

**(обязательное)**

**Листинг программы**

<?php

namespace BPashkevich\ProductBundle\Controller;

use BPashkevich\ProductBundle\Entity\Attribute;

use BPashkevich\ProductBundle\Entity\ConfigurableProduct;

use BPashkevich\ProductBundle\Entity\Product;

use BPashkevich\ProductBundle\Services\AttributeService;

use BPashkevich\ProductBundle\Services\AttributeValueService;

use BPashkevich\ProductBundle\Services\ConfigurableProductService;

use BPashkevich\ProductBundle\Services\ImageService;

use BPashkevich\ProductBundle\Services\ProductService;

use BPashkevich\ProductBundle\Services\CategoryService;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;

use BPashkevich\ProductBundle\Entity\AttributeValue;

use BPashkevich\ProductBundle\Entity\Category;

use BPashkevich\ProductBundle\Entity\Image;

use Symfony\Component\HttpFoundation\Request;

/\*\*

\* Product controller.

\*

\*/

class ProductController extends Controller

{

private $productService;

private $categoryService;

private $attributeService;

private $attributeValueService;

private $imageService;

private $configurableProductService;

public function \_\_construct(ProductService $productService, CategoryService $categoryService,

ImageService $imageService, AttributeService $attributeService,

AttributeValueService $attributeValueService,

ConfigurableProductService $configurableProductService)

{

$this->productService = $productService;

$this->categoryService = $categoryService;

$this->attributeService = $attributeService;

$this->attributeValueService = $attributeValueService;

$this->configurableProductService = $configurableProductService;

$this->imageService = $imageService;

}

/\*\*

\* Lists all product entities.

\*

\*/

public function indexAction()

{

$products = $this->productService->getAllProduct();

$attributes = [];

foreach ($products as $product){

$attributes[$product->getId()] = $product->getAttributes();

}

return $this->render('product/index.html.twig', array(

'products' => $products,

'attributes' => $attributes,

'categories' => $this->categoryService->getAllCategories(),

));

}

/\*\*

\* Creates a new product entity.

\*

\*/

public function newAction(Request $request, ConfigurableProduct $configurableProduct = null)

{

$result['attributes'] = $this->attributeService->findAttributes(array('mandatory' => 1,));

if ($configurableProduct){

$result['id'] = $configurableProduct->getId();

$result['configAttr'] = $this->configurableProductService->getNotMandatoryAttributes($configurableProduct);

}

else{

$result['categories'] = $this->categoryService->getAllCategories();

}

return $this->render('product/new.html.twig', $result);

}

public function saveAction(Request $request, ConfigurableProduct $configurableProduct = null)

{

$product = new Product();

$requestData = $request->request->all();

if ($configurableProduct){

$requestData['category'] = $configurableProduct->getCategory()->getId();

$product->setConfigurableProduct($configurableProduct);

}

$requestDataKeys = array\_keys($requestData);

$attributes = [];

$attributesValues = [];

$category = new Category();

$image = new Image();

foreach ($requestDataKeys as $key){

switch ($key){

case "category":

$category = $this->categoryService->findCategories(array(

'id' => $requestData[$key],

))[0];

break;

case "image":

$image->setUrl($requestData[$key]);

break;

default:

$attributes[] = $this->attributeService->findAttributes(array('id' => $key,))[0];

break;

}

}

unset($requestData['category']);

unset($requestData['image']);

$counter = 0;

foreach ($requestData as $attribute){

$val = $this->attributeValueService->findAttributeValues(array(

'attribute' => $attributes[$counter],

'value' => $attribute,

));

if(!$val){

$attributeValue = new AttributeValue();

$attributeValue->setAttribute($attributes[$counter]);

$attributeValue->setValue($attribute);

$this->attributeValueService->createAttributeValue($attributeValue);

$attributesValues[] = $attributeValue;

}

else{

$attributesValues[] = $val[0];

}

$counter++;

}

$product->setCategory($category);

$this->productService->createProduct($product, $attributes, $attributesValues);

$image->setProduct($product);

$this->imageService->createImage($image);

return $this->redirectToRoute('product\_show', array('id' => $product->getId()));

}

public function newProductFromConfigurableAction(){

}

/\*\*

\* Finds and displays a product entity.

\*

\*/

public function showAction(Product $product)

{

// die(var\_dump($this->productService->getAttributesValues($product)));

return $this->render('product/show.html.twig', array(

'data' => $this->productService->getAttributesValues($product),

'product' => $product,

'categoryName' => $product->getCategory(),

'categories' => $this->categoryService->getAllCategories(),

'isConsist' => in\_array($product->getId(), $\_SESSION['cart']),

));

}

/\*\*

\* Displays a form to edit an existing product entity.

\*

\*/

public function editAction(Request $request, Product $product)

{

$deleteForm = $this->createDeleteForm($product);

$editForm = $this->createForm('BPashkevich\ProductBundle\Form\ProductType', $product);

$editForm->handleRequest($request);

if ($editForm->isSubmitted() && $editForm->isValid()) {

$this->getDoctrine()->getManager()->flush();

return $this->redirectToRoute('product\_edit', array('id' => $product->getId()));

}

return $this->render('product/edit.html.twig', array(

'product' => $product,

'edit\_form' => $editForm->createView(),

'delete\_form' => $deleteForm->createView(),

));

}

/\*\*

\* Deletes a product entity.

\*

\*/

public function deleteAction(Request $request, Product $product)

{

$form = $this->createDeleteForm($product);

$form->handleRequest($request);

if ($form->isSubmitted() && $form->isValid()) {

$em = $this->getDoctrine()->getManager();

$em->remove($product);

$em->flush();

}

return $this->redirectToRoute('product\_index');

}

/\*\*

\* Creates a form to delete a product entity.

\*

\* @param Product $product The product entity

\*

\* @return \Symfony\Component\Form\Form The form

\*/

private function createDeleteForm(Product $product)

{

return $this->createFormBuilder()

->setAction($this->generateUrl('product\_delete', array('id' => $product->getId())))

->setMethod('DELETE')

->getForm()

;

}

}

<?php

namespace BPashkevich\ProductBundle\Services;

use BPashkevich\ProductBundle\Entity\Attribute;

use BPashkevich\ProductBundle\Entity\Product;

use Symfony\Component\HttpFoundation\Session\Session;

class ProductService

{

private $em;

private $session;

private $dbService;

private $repository;

public function \_\_construct(\Doctrine\ORM\EntityManager $em, Session $session, DBService $dbService)

{

$this->em = $em;

$this->session = $session;

$this->dbService = $dbService;

$this->repository = $this->em->getRepository('BPashkevichProductBundle:Product');

}

public function getAllProduct()

{

return $this->repository->findAll();

}

public function findProducts(array $params)

{

return $this->repository->findBy($params);

}

public function getShortInfo($product, $attr)

{

$queryBuilder = $this->dbService->getQueryBuilder();

$queryBuilder

->select('v.value')

->from('product\_attribute\_value', 'p')

->innerJoin('p', 'attribute', 'a', 'p.attribute\_id=a.id')

->innerJoin('p', 'attribute\_value', 'v','p.value\_id=v.id')

->where('p.product\_id = ?')

->andWhere('a.name = ?')

->setParameter(0, $product->getId())

->setParameter(1, $attr);

$sth = $queryBuilder->execute();

return $sth->fetch()['value'];

}

public function getMainInfo(Product $product)

{

return array(

'id' => $product->getId(),

'name' => $this->getShortInfo($product, 'Name'),

'price' => $this->getShortInfo($product, 'Price'),

'shortDescription' => $this->getShortInfo($product, 'Short Description'),

'img' => $product->getImages()[0]->getUrl()

);

}

public function getSingleProductMainInfo($id)

{

$result = array();

if($id != null) {

/\*\* @var Product $product \*/

$products = $this->findProducts(array('id' => $id, 'configurableProduct' => null));

} else {

$products = $this->findProducts(array('configurableProduct' => null));

}

foreach ($products as $product) {

$result[] = array(

'id' => $product->getId(),

'name' => $this->getShortInfo($product, 'Name'),

'price' => $this->getShortInfo($product, 'Price'),

'shortDescription' => $this->getShortInfo($product, 'Short Description'),

'img' => $product->getImages()[0]->getUrl()

);

}

return $result;

}

public function selectSingleProducts($products)

{

$singleProducts = array();

foreach ($products as $product){

if($product->getConfigurableProduct() == null){

$singleProducts[] = $product;

}

}

return $singleProducts;

}

public function createProduct(Product $product, array $attributes, array $attributeValues)

{

if(count($attributes) == count($attributeValues)){

$queryBuilder = $this->dbService->getQueryBuilder();

// $product->setSeller($this->session->get('currentUserName'))

$this->em->persist($product);

$this->em->flush();

for ($i=0; $i<count($attributes); $i++) {

$queryBuilder->insert('product\_attribute\_value')

->setValue('product\_id', $product->getId())

->setValue('attribute\_id', $attributes[$i]->getId())

->setValue('value\_id', $attributeValues[$i]->getId());

$queryBuilder->execute();

}

return $product;

}

return null;

}

public function getAttributesId(Product $product)

{

$queryBuilder = $this->dbService->getQueryBuilder();

$queryBuilder

->select('attribute\_id')

->from('product\_attribute\_value')

->where('product\_id = ?')

->setParameter(0, $product->getId());

$sth = $queryBuilder->execute();

$data = $sth->fetchAll();

$attrs = [];

for ($i = 0; $i < count($data); $i++){

$attrs[$i] = $data[$i]['attribute\_id'];

}

return $attrs;

}

public function getAttributesValuesId($productId, $attrId)

{

$queryBuilder = $this->dbService->getQueryBuilder();

$queryBuilder

->select('value\_id', 'value')

->from('product\_attribute\_value', 'p')

->innerJoin('p', 'attribute\_value', 'v','p.value\_id=v.id')

->where('p.product\_id = ? and p.attribute\_id = ?')

->setParameter(0, $productId)

->setParameter(1, $attrId);

$sth = $queryBuilder->execute();

$data = $sth->fetchAll();

// var\_dump($data);

// die();

return array($data[0]['value\_id'], $data[0]['value']);

}

public function getAttributesValues(Product $product)

{

$queryBuilder = $this->dbService->getQueryBuilder();

$queryBuilder

->select('a.name', 'v.value')

->from('product\_attribute\_value', 'p')

->innerJoin('p', 'attribute', 'a', 'p.attribute\_id=a.id')

->innerJoin('p', 'attribute\_value', 'v','p.value\_id=v.id')

->where('p.product\_id = ?')

->setParameter(0, $product->getId());

$sth = $queryBuilder->execute();

$value = $sth->fetchAll();

// var\_dump($value);

// die();

return $value;

}

public function editProduct(Product $product)

{

$this->em->flush();

return $product;

}

public function deleteProduct(Product $product)

{

$this->em->remove($product);

$this->em->flush();

}

}