

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: «Программное обеспечение средств
вычислительной техники и автоматизированных
систем»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

БАКАЛАВРА

на тему

**СЕТЕВОЕ ПРИЛОЖЕНИЕ «БИЗНЕС-
ПОТРЕБИТЕЛИ» ДЛЯ ЭЛЕКТРОННОГО
МАГАЗИНА**

Студент _____ 31.05.2022 г. Ткачев А.М.
(подпись, дата)

Руководитель _____ 31.05.2022 г. Карамышева Н.С.
(подпись, дата)

Нормоконтролёр _____ 31.05.2022 г. Бычков А.С.
(подпись, дата)

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

Заведующий кафедрой

(подпись)

М.А. Митрохин

Работа защищена с отметкой хорошо (протокол заседания ГЭК от
21 июня № 22)
2022 г.

Секретарь ГЭК

(подпись)

Е.И. Гурин

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 2022 г.

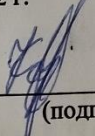
**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ БАКАЛАВАРА**

1. Студент Ткачев Артем Михайлович, гр.18ВВ2 факультета вычислительной техники.
2. Тема работы: «Сетевое приложение «Бизнес-потребители» для электронного магазина»
Тема утверждена приказом ПГУ №357/о от "26" апреля 2022 г.
3. Руководитель бакалаврской работы к.т.н., доцент каф. ВТ ПГУ Карамышева Н.С.
4. Задание на работу (назначение разработки, исходные данные и т.п.)
Сетевое приложение «бизнес-потребители» для электронного магазина, предназначенный для продажи японского жанра литературы «Манга».
5. Перечень подлежащих разработке вопросов
Требуется произвести анализ предметной области, выбрать среду программирования, базы данных и язык программирования для приложения. Изучение особенностей разработки под web-приложения. Также выполнить задание по БЖД.

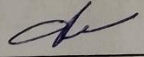
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	20	26.04.2022 – 29.04.2022	
2.	Выбор среды, базы данных и языка программирования	20	30.04.2022 – 02.05.2022	
3.	Изучение особенности разработки под web приложения	51	03.05.2022 – 07.05.2022	
4.	Разработка базы данных	30	08.05.2022 – 15.05.2022	
5.	Разработка <i>frontend</i> части	46	16.05.2022 – 19.05.2022	
6.	Разработка <i>backend</i> части	111	20.05.2022 – 27.05.2022	
7.	Оформление пояснительной записки	46	28.05.2022 – 31.05.2022	

Дата выдачи задания "26" апреля 2022 г.

Руководитель бакалаврской работы  26.04.22 Карамышева Н.С.
(подпись, дата)

Задание к исполнению принял студент  26.04.22 Ткачев А.М.
(подпись, дата)

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

23.06.2022

Реферат

Работа содержит 112 листов, 22 рисунка, 7 таблиц, 8 источников, 4 приложения.

СЕТЕВОЕ ПРИЛОЖЕНИЕ, ЭЛЕКТРОННЫЙ МАГАЗИНА, ИНТЕРНЕТ-МАГАЗИН, ЯПОНСКОЙ ЛИТЕРАТУРЫ, БАЗЫ ДАННЫХ

Цель ВКР – разработать сетевое приложение, позволяющее пользователям покупать товары.

Разработанный электронный магазин предназначен для предоставления каталога продуктов японской литературы и возможности её покупки.

В процессе выполнения работы производилось:

- разработана структура базы данных для СУБД *MySQL*;
- разработан фронтенд электронного магазина;
- разработан и отлажен бэкенд для электронного магазина.

В результате выполнения работы было создано сетевое приложение, удовлетворяющее поставленной задаче.

					ПГУ 1.090301.40.001 ПЗ			
Изм.	Лист	№ докум	Подпись	Дата	Сетевое приложение «Бизнес-потребители» для электронного магазина	Лит.	Лист	Листов
Разраб.		Ткачев А.М.					2	112
Провер.		Карамышева Н.С.				ФВТ, каф. ВТ гр. 18ВВ2		
Н. Контр.		Бычков А.С.						
Утверд.		Митрохин М.А.			Пояснительная записка			

Содержание

Перечень принятых сокращений и обозначений.....	4
Введение	5
1 Сетевое приложение «бизнес-потребители» для электронного магазина	8
1.1 Анализ аналогов, постановка задачи.....	8
1.1.1 Анализ аналогов	8
1.1.2 Постановка задачи	10
1.2 Анализ предметной области	11
1.2.1 Архитектура.....	11
1.2.1.1 Описание архитектуры приложения.....	11
1.2.2.2 Выбор реляционной СУБД.....	14
1.2.2.3 Выбор языка программирования для серверной части	17
1.2.2.4 Выбор языков программирования для разметки страницы.....	18
1.2.2.5 Анализ фреймворков	19
1.3 Реализация	21
1.3.1 Структура БД.....	21
1.3.2 Описание классов	23
1.4 Описание результата	24
2 Безопасность жизнедеятельности.....	33
2.1 Расчет эквивалентного уровня звука на рабочих местах в компьютерных залах, офисах и других помещениях, где установлены ПЭВМ	33
Заключение.....	40
Список использованных источников	41
Приложение А – UML-диаграммы	42
Приложение Б – Модель базы данных.....	46
Приложение В – Листинг программы.....	53
Приложение Д – Презентация	106

Перечень принятых сокращений и обозначений

API (Application Programming Interface) - интерфейс прикладных программ

CSS (Cascading Style Sheets) – каскадные таблицы стилей описания внешнего вида документа

HTML (HyperText Markup Language) - язык разметки гипертекста

HTTP (HyperText Transfer Protocol) - протокол передачи гипертекста

ID (Identifier) - идентификатор

JS - язык программирования *JavaScript*

MVC (Model-View-Controller) - шаблон проектирования Модель-Вид-Контроллер

NoSQL (не только *SQL*) - не просто *SQL*

PHP (Hypertext Preprocessor) - язык программирования

SQL (Structured Query Language) - структурированный язык запросов

UML (Unified Modelling Language) - унифицированный язык моделирования

БД - база данных

ОС - операционная система

ПО - программное обеспечение

ПК – персональный компьютер

СУБД - система управления базами данных

					ПГУ 1.090301.40.001 ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

На сегодняшний день развитие Интернета вышло на высокий уровень и все больше пользователей появляется в сети. Одним из больших плюсов Интернета, является электронная коммерция или другими словами – онлайн торговля.

В современном обществе большинство компаний переносят значительную часть делового общения в Сеть. Электронная коммерция (*e-commerce*) - это предпринимательская деятельность, которая, так или иначе, связана с распространением, рекламированием, продвижением, продажей услуг или товаров через Интернет. Это улучшает большинство бизнес-процессов за счет их проведения электронным образом. Иначе, электронная коммерция – это ведение бизнеса через Интернет.

Электронная коммерция объединяет такие глобальные категории, как онлайн-продажи, интернет-банкинг, бронирования билетов и отелей, транзакции в платежных системах, онлайн-маркетинг и реклама.

Среди отличительных особенностей онлайн коммерции перед традиционной торговлей можно отметить:

- отсутствие географических, временных и языковых барьеров, что позволяет продвигать товары и услуги на новые рынки сбыта;
- более низкий уровень издержек производства и обращения, что достигается путем внедрения новых технологий во все сферы деятельности компаний: начиная от закупок сырья и материалов и заканчивая дистрибуцией готовой продукции и пост-продажным обслуживанием;
- более высокий уровень конкуренции. Расстояние между магазинами всего несколько секунд - именно это время необходимо для загрузки соответствующего сайта;
- потенциальная емкость электронного магазина значительно превышает емкость традиционных магазинов по причине отсутствия физических ограничений на складские и торговые помещения.

Существует множество видов электронной коммерции, в работе будет

					ПГУ 1.090301.40.001 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

рассмотрен «Бизнес для потребителя» B2C (*Business-to-Consumer*).

Сфера «Бизнес для потребителя» предполагает торговлю товарами и услугами между юридическими и физическими лицами. Это своего рода розничные продажи, но только с помощью онлайн-площадок – магазины, сервисы, банки и прочее. Преимущество клиентов в большем ассортименте выбора, удобстве заказа и доставки товаров на дом или в офис. Электронная коммерция позволяет предпринимателю снизить затраты на содержание торговых и складских площадей.

Ниже представлен принцип работы электронной коммерции на основе «Бизнес для потребителя» или же интернет-магазин:

- взаимодействие с сервером. Покупатель, просматривая онлайн-каталог, решает выбрать продукт. Его ПК или мобильное устройство через браузер взаимодействует с сервером, на котором находится магазин;
- сервер – менеджер. Сервер, получив заявку пользователя, отправляет её менеджеру в систему обработки заказов;
- проверка товара менеджером. Сотрудник делает запрос в базу данных о наличии товара на складе. Если товара нет, то делается запрос производителю, выясняется время поставки на склад, после чего клиенту предоставляется конкретная информация;
- платежная система. Обращение к финансовой системе с запросом на обработку транзакции – оплаты от клиента;
- подтверждение заказа. Финансовая система разрешает или блокирует проведение сделки в зависимости от ситуации со счетом клиента (недостаток денег, нерабочая карта и другое). Если сделка совершается успешно, то менеджер подтверждает транзакцию и уведомляет об этом сервер;
- обработка заказа. Сервер выводит покупателю сообщение, что оплата успешно проведена, а заказ принят в обработку;
- упаковка товара. Заявка направляется на склад, где формируется товар для отправки по адресу клиента;

					ПГУ 1.090301.40.001 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

- доставка товара. Сотрудники логистического отдела доставляют товар покупателю. После отправки заказа со склада, сервер высылает клиенту оповещение на *e-mail* или телефон, что товар находится в пути.

В техническом плане электронная коммерция в интернете стоит на трёх основах – сервер, база данных и система доставки товара или услуги покупателю. Критически важна первая составляющая – качественный и быстрый сервер. База данных нужна крупным объектам, а доставка электронных товаров или услуг не требует сложной логистики.

В данной работе будет рассмотрена реализация сетевое приложение «бизнес-потребители» для электронного магазина. Целью создания сетевого приложения является предоставление каталога продуктов японской литературы и возможности её покупки.

Практическая значимость данной работы заключается в получении завершённого и автономного сетевого продукта, готового к использованию.

					ПГУ 1.090301.40.001 ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

1 Сетевое приложение «бизнес-потребители» для электронного магазина

1.1 Анализ аналогов, постановка задачи

По мере развития технологий и Интернета все больше информации хранится онлайн в самых разных местах. Сайты по продаже различных товаров не являются исключением, и все больше потребителей используют их вместо реальных магазинов и торговых центров, чтобы купить различный товар.

Сетевое приложение предоставляет доступ к интернет-магазину по продаже Манги.

При разработке программного продукта важно точно нацелиться на рынок, проанализировать его и выявить положительные и отрицательные стороны различных конкурентов. В данной работе описывается бэкенд микросервиса, отвечающего за сбор статистики по видам спорта, командам и игрокам.

1.1.1 Анализ аналогов

Рынок веб-приложений для покупки литературы довольно обширный.

Для сравнения были выбраны следующие критерии: дизайн, простота в использовании, информативность, стабильность работы.

Исследуемые разработки были разделены на 3 категории.

В первую входят интернет-магазины, занимающиеся продажей одежды, обуви, электроники, детских товаров, товаров для дома и множества других, такие как: *AliExpress*, *WildBerries*, *OZON*, на рисунке 1 показан данный ресурс.

Такие ресурсы отличаются своей простотой в использовании и поставили себя на рынке, как надежный и быстрый способ получить желаемые вещи через интернет, также имеют довольно приятный дизайн. Их главный недостаток в том, что они обладают небольшим каталогом японской литературы и нет четкого фильтра по жанрам и издателям.

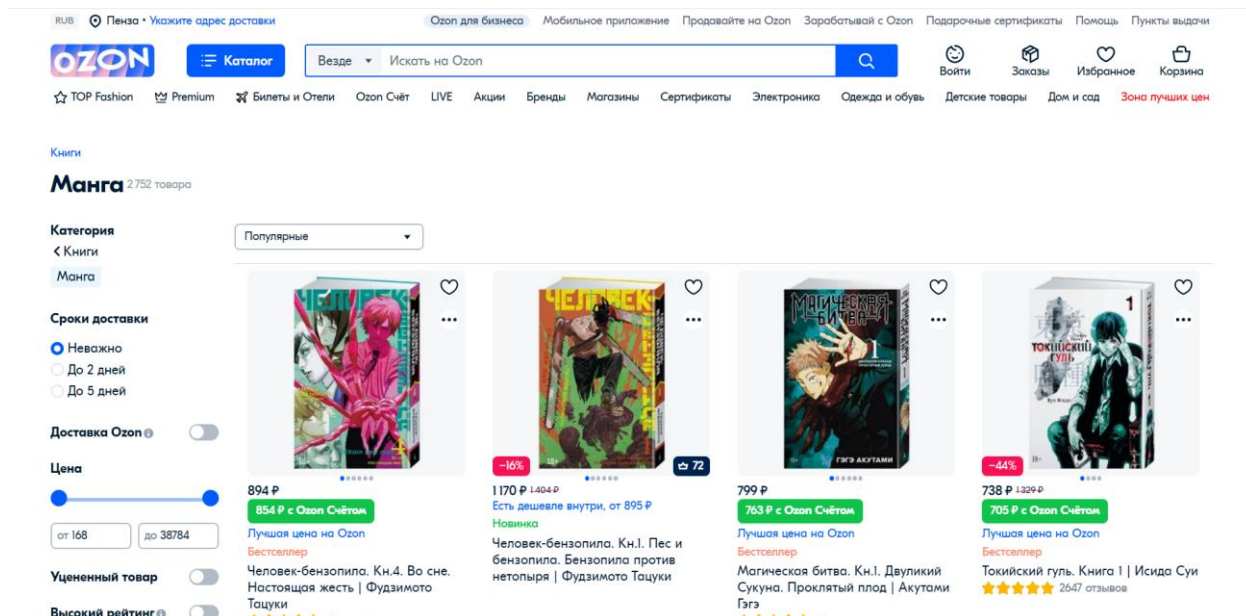


Рисунок 1 - Сайт «OZON»

Во второй категории сайты «Читай город» и «Буквоед» – это магазины, специализирующиеся на продаже литературы, сайт интернет-магазина представлен на рисунке 2.

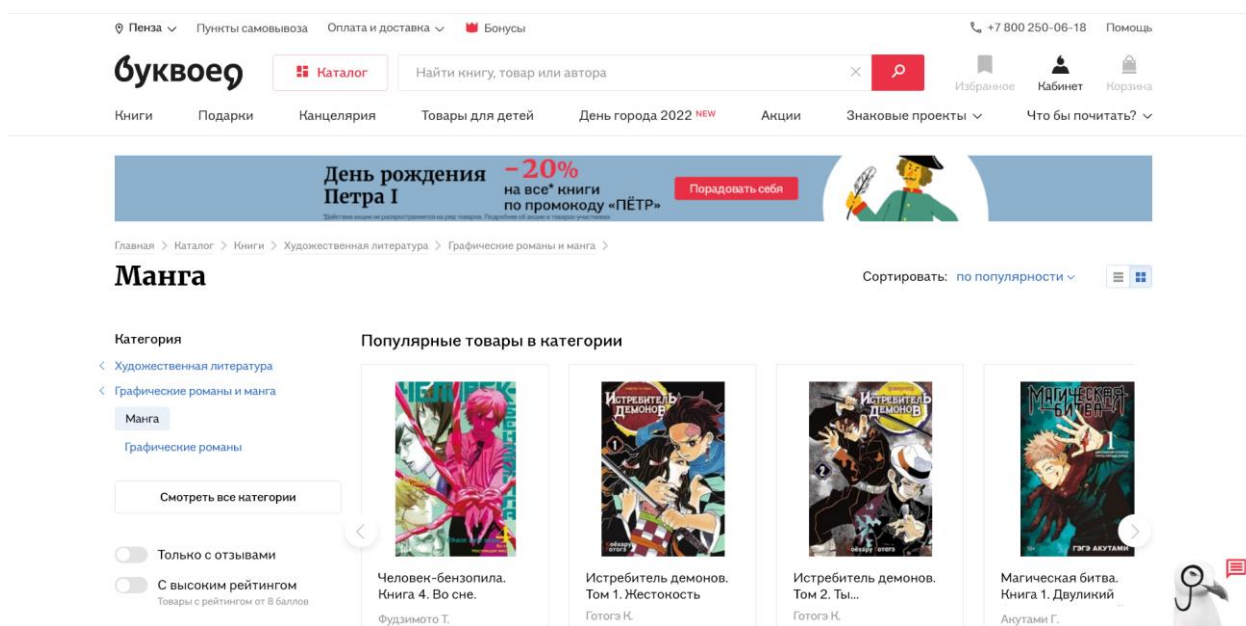


Рисунок 2 - Сайт «Буквоед»

Также, как и у прошлой категории они имеют приятный дизайн, но в системе фильтрации есть только общий раздел «Манга», это усложняет поиск

нужной книги, так как у данного жанра литературы есть десятки категорий, на разные вкусы.

И в последнюю категорию входят интернет-магазины по продаже аниме атрибутики, например, «Fast Anime», ресурс показан на рисунке 3.

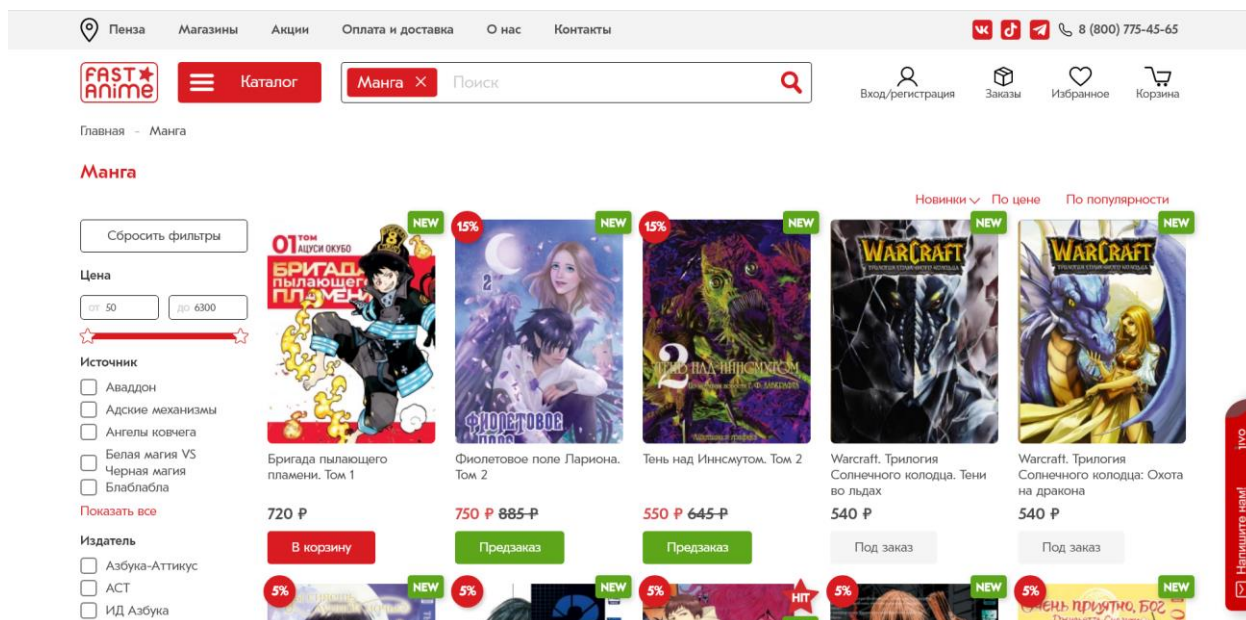


Рисунок 3 - Сайт «Fast Anime»

Данный ресурс занимается продажей не только Манги. Имеет широкий выбор товара и самое важное, систему фильтрации с каждым жанром японской литературы.

1.1.2 Постановка задачи

Исходя из проведённого анализа аналогичных ресурсов был сделан вывод, о том, что в разрабатываемом интернет-магазине должны обязательно присутствовать следующие функции:

- максимально понятным сайт и простой для восприятия;
- фильтрации данных по различным параметрам;
- вывод достаточной информации о продукте для покупателя.

Благодаря описанным выше функциям, интернет-магазин будет актуален среди пользователей и составит конкуренцию на рынке приложений по продаже Манги. Помимо функций, которые описаны выше, сайт должен содержать и

базовые: платежная система, фильтрация. Исходя из проведенного анализа аналогичных разработок, были поставлены следующие задачи.

Приложение будет реализовывать *beck-end* и *front-end* часть взаимодействия клиент-сервера, развёртку базы данных и реализацию *MVC* архитектуры.

В основе проекта будет лежать две роли, пользователь и администратор, изображенные в приложении А.1. Первый и самый основной будет получать информацию о товарах, добавлять их в корзину и регистрироваться на сайте, для покупки товара. Администратор - как пользователь с более широким спектром доступа. Он будет иметь возможность добавлять информацию и корректировать поступающий контент на сайт.

1.2 Анализ предметной области

При разработки программного обеспечение пользователю необходимо продумать архитектуру приложения. В этой главе обсуждаются общие теоретические вопросы, связанные с техническими элементами, необходимыми для внедрения программного обеспечения. Это программное обеспечение включает:

- выбор системы баз данных;
- архитектура программного обеспечения;
- языки программирования.

В каждом подразделе обосновывается выбор того или иного технического решения проблемы.

1.2.1 Архитектура

1.2.1.1 Описание архитектуры приложения

Архитектура всех интернет-магазинов состоит из программной части, администрирования и клиентской части.

Серверная часть содержит в себе размещение интернет-магазина на *web*-хостинге, поддерживающий технологии, используемые при создании интернет-магазина.

В качестве шаблона для разработки ПО был выбран *MVC* (Модель-Вид-

Контроллер) [1]. Паттерн разделяет логику приложения на три части, описывает простой способ построения структуры приложения, целью которого является отделение бизнес-логики от пользовательского интерфейса. В результате, приложение легче масштабируется, тестируется, сопровождается и конечно же реализуется.

1.2.1.2 Выбор архитектуры приложения

Для упрощения выполняемых задач существуют шаблоны или паттерны проектирования, они содержат готовые решения большинства задач, в том числе и архитектуру ПО.

В качестве шаблона для разработки ПО был выбран *MVC* шаблон (Модель-Вид-Контроллер). Паттерн разделяет логику приложения на три части, описывает простой способ построения структуры приложения, целью которого является отделение бизнес-логики от пользовательского интерфейса. В результате, приложение легче масштабируется, тестируется, сопровождается и конечно же реализуется. Структура шаблона представлена на рисунке 4.

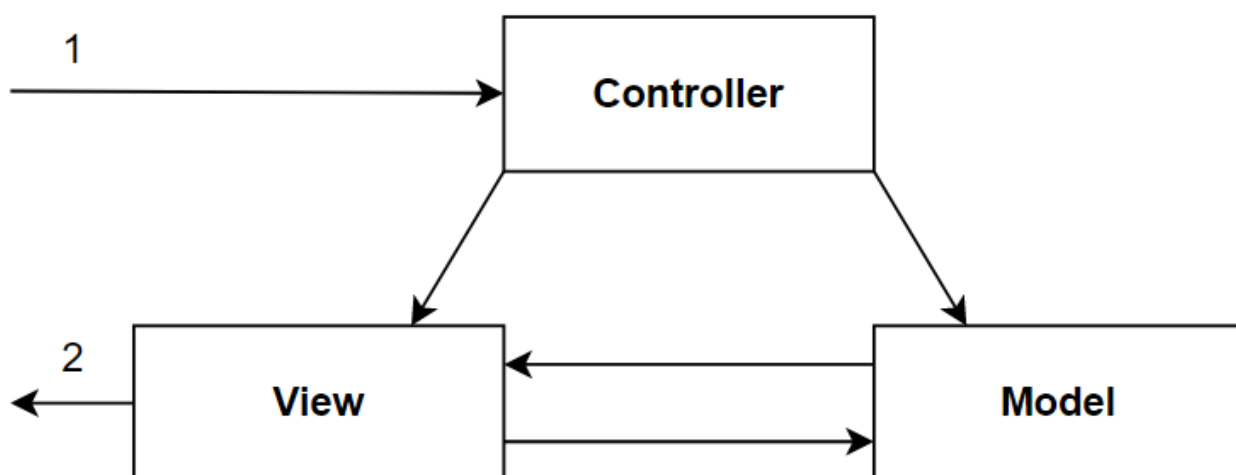


Рисунок 4 - Схема шаблона *MVC*

На рисунке видно два состояния: вход 1 и вывод информации 2.

На входе пользователь отправляет запрос приложению, создается экземпляр запрошенного контроллера и вызывается указанное действие.

На выходе пользователь видит результаты работы.

Model — содержит логику приложения и включает методы выборки (это могут быть методы *ORM-Object-Relational Mapping*), обработки (например, правила валидации) и предоставления конкретных данных.

View — используется для задания внешнего отображения данных, полученных из контроллера и модели.

Controller — связующее звено, соединяющее модели, виды и другие компоненты в рабочее приложение. Контроллер отвечает за обработку запросов пользователя.

MVC самый популярный паттерн, который используется пользователями на протяжении долгого времени. Он отличается своей простотой, так как обладает единой архитектурой, все данные разбиты на отдельные блоки, от этого также легко находить и исправлять ошибки.

Но у *MVC* есть и недостаток в виде усложненного механизма разделения программы на модули.

1.2.2 Выбор инструментов разработки

1.2.2.1 Анализ типов СУБД

Для интернет-магазина важно использование баз данных, предназначенные для хранения информации. Например, вы делаете заказ, а система сохраняет данные в базе.

В *IT*-среде существует два варианта баз данных: реляционные (*SQL*) и нереляционные (*NoSQL*) [2]. Простыми словами, первые – структурированные, а вторые – неструктурированные. Необходимо разобраться в них более подробно, чтобы сделать выбор между различными типами баз данных.

Система управления базами данных (СУБД) – передовое программное обеспечение высокого уровня, работающее с *API* низкого уровня.

NoSQL – аналог реляционной базы данных, в которой информация хранится без строгой структуры и явной связи между другими сведениями.

Существует шесть различий между этими БД, каждое из которых будет рассмотрено более подробно.

Структура и тип хранимых данных. *SQL* требует жесткой структуризации на основании шаблонов. В *NoSQL* по отношению к структуре не предъявляется никаких требований. В сетевом приложении очень важны запросы с несколькими условными вариантами выбора и использование нереляционной базы данных вызывает проблемы при создании запроса, поэтому в отношении структуры и хранения данных должна быть выбрана реляционная база данных.

Масштабируемость. В *SQL* предусмотрено вертикальное масштабирование. В *NoSQL* можно использовать и вертикальное, и горизонтальное. Но нереляционную базу данных лучше использовать горизонтально. В данном случае имеет смысл выбрать решение с горизонтальным масштабированием.

Запросы. В реляционных системах получить данные можно при помощи языка *SQL*. А вот в каждой *NoSQL*-базе предусмотрен свой алгоритм работы. Язык *SQL* прост в освоении и выбор пал на реляционный вариант.

Надежность. *SQL* более простые и удобные в последующей работе благодаря своей структуризации. *NoSQL* имеет высокую защиту от хакерских атак.

Работа с данными сложных структур. Здесь лучше подходят реляционные БД, что также связано с наличием строгой структуры.

Поддержка. *SQL* существуют намного дольше нереляционных аналогов, пользуются повышенной популярностью. А вот *NoSQL* пока не так популярны. Поэтому выбор снова в пользу *SQL*.

Реляционная *SQL* будет оптимальной при обработке большого числа сложных запросов и долгого анализа информации. Поэтому предпочтение отдано *SQL*.

1.2.2.2 Выбор реляционной СУБД

Проанализируем наиболее популярные и появляющиеся на рынке базы данных *SQL*, чтобы выбрать подходящую СУБД. Это *MySQL*, *PostgreSQL*, *SQLite*.

SQLite – это производительная библиотека, которую можно встраивать в приложения. Полноценная БД на основе файлов *SQLite* предлагает широкий

набор инструментов для обработки всех видов данных и накладывает намного меньше ограничений, чем другие реляционные базы данных.

Преимущества *SQLite*:

- простое строение на основе файлов. Вся БД состоит всего из одного файла;
- стандарты: система *SQLite* основана на *SQL*. Некоторые функции были опущены и заменены другими;
- отлично подходит для разработки или тестирования.

Недостатки:

- нет управления пользователями (взаимосвязи, привилегии и т.п.);
- невозможно повысить уровень производительности.

SQLite не подходит для многопользовательских приложений, её лучше использовать в простых встроенных приложения, когда нужна портативность, например, мобильные приложения и игры.

PostgreSQL.

PostgreSQL – это продвинутая открытая объектно-ориентированная СУБД. *PostgreSQL* реализует *SQL*-стандарты *ANSI/ISO*.

Преимущества *PostgreSQL*:

- масштабируемость и расширяемость;
- бесплатная и для личного и для коммерческого использования;
- объектно-ориентированность;
- поддержка сторонних инструментов для проектирования, управления данными и т.п.

Недостатки:

- иногда производительность ниже, чем у *MySQL*;
- трудности при обновлении;
- не все хостинг-провайдеры поддерживают *PostgreSQL*.

Данную СУБД лучше использовать, когда нужна целостность данных и сложные пользовательские задачи. Также если в будущем приложению понадобится более надёжная платная БД, с *PostgreSQL* легче будет перейти. Но не подходит к приложениям, в которых нужны быстрые операции чтения и не нужна

абсолютная целостность данных.

MySQL.

MySQL – самая популярная СУБД [3]. Это многофункциональное открытое приложение, поддерживающее работу огромного количества сайтов. Система *MySQL* довольно проста в работе и может хранить большие массивы данных.

Преимущества *MySQL*:

- простота в использовании;
- большая популярность;
- функциональность;
- безопасность, благодаря многим встроенным функциям для защиты информации;
- масштабируемость и производительность.

Недостатки:

- существуют некоторые ограничения, из-за которых могут не работать продвинутое приложения;
- менее уязвима по сравнению с другими СУБД;
- *MySQL* медленно развивается.

MySQL может поддерживать работу почти любого сайта и веб-приложения за счет своей гибкости и масштабируемости. Обладает высокой безопасностью данных. Также имеет возможность обрабатывать большое количество операций. *MySQL* не всегда можно интегрировать с другой СУБД, из-за того, что не полностью реализует стандарты *SQL*.

Несмотря на некоторые недостатки, была выбрана СУБД *MySQL*. С её помощью можно реализовать все функции, которые нужны в данном проекте.

MySQL обладает кроссплатформенностью, имеются дистрибутивы под самые различные ОС, в том числе наиболее популярные версии *Linux*, *Windows*, *MacOS*. Данная СУБД легко работает с большим количеством данных и легко масштабируется. Безопасность поддерживается по умолчанию. *MySQL* достаточно производительна за счет упрощения некоторых функций.

1.2.2.3 Выбор языка программирования для серверной части

Бэкенд (англ. *backend*) – содержит *API* и всю логику сайта. Также в бэкенд входит база данных, которая была выбрана в прошлом пункте.

При реализации серверной части (*backend*) одним из самых важных этапов является выбор языка программирования. От этого зависит продуктивность и чистота кода, а также длительность периода создания минимально жизнеспособного продукта.

В настоящее время наиболее популярными языками программирования для *backend* разработки являются: *PHP*, *Go*, *Java*, *Python*, *Ruby*, *C#*.

Язык программирования *Go*.

Go (*Golang*) – компилируемый многопоточный язык программирования, который разработали в *Google* [4]. Он подходит для бэкенда веб-приложений и других проектов. Является надёжным и быстрым, и поэтому он становится всё популярнее. Программа на таком языке может выполнять несколько строк одновременно. Но данный язык не является объектно-ориентированным, так как в нем не включена функция наследования.

Язык программирования *Python*.

Python – высокоуровневый язык программирования, который ориентирован на повышение производительности разработчика и читаемости кода [5]. Этот язык очень гибкий и обладает широким функционалом, а код высокой степенью читаемости. Он обладает низким быстродействием и поэтому часто используется как язык второго плана.

Язык программирования *Ruby*.

Ruby – это язык программирования общего назначения. Типы данных определяются на этапе выполнения программы, что увеличивает скорость разработки. Код на *Ruby* обрабатывается интерпретатором в момент выполнения без предварительной компиляции, это обеспечивает независимость от платформ и уменьшает размер исполняемых программ.

Язык программирования *C#*.

C# – объектно-ориентированный и типобезопасный язык

Изм.	Лист	№ докум.	Подпись	Дата

программирования. Позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в *.NET*.

Язык программирования *Java*.

Java – язык программирования общего назначения. Относится к объектно-ориентированным языкам программирования, к языкам с сильной типизацией [6]. Основной особенностью данного языка является то, что программы на *Java* транслируются в байт-код, поэтому написанное на *Java* приложение можно запустить на любой платформе, если на ней установлена среда исполнения *Java*. В настоящее время *Java* является одним из самых популярных языков программирования.

Язык программирования *PHP*.

PHP – скриптовый язык программирования общего назначения, созданный для генерации *HTML*-страниц на веб-сервере и работы с базами данных [2]. На сегодняшний момент поддерживается подавляющим большинством представителей хостингов. *PHP* специально сконструирован для веб-разработок и его код может внедряться непосредственно в *HTML* или наоборот.

Выбор языка программирования.

В качестве языка программирования был выбран язык *PHP*. Это крайне гибкий язык программирования, что позволяет решить одну задачу десятками способов, достаточно много литературы по этому языку и готовых способов решения задач. На *PHP* код самой программы проще и короче, что уменьшает затраты на разработку и обслуживание.

1.2.2.4 Выбор языков программирования для разметки страницы

Фронтенд (англ. *frontend*) — то, что пользователь видит на экране и с чем может взаимодействовать: страницы сайта, меню, кнопки, поля ввода и другие части интерфейса.

В клиентской (*frontend*) части разработки был выбран основной стек фронтенд-разработки: *HTML*, *CSS* и *JavaScript* [2].

HTML – стандартный язык разметки в *web*-программировании, с помощью которого будет производится вёрстка сайта. Веб-браузер преобразует *HTML*-

документ в удобной для человека форме. С этими документами можно работать с любого текстового редактора.

CSS – язык таблиц стилей используется для описания параметров оформления документа. С помощью этого языка можно более детально оформить веб-страницу.

JavaScript – объектно-ориентированный язык сценариев и скриптов [7]. Он предназначен для связи веб-приложения с действиями пользователя. *JavaScript* мультипарадигменный язык сценариев, который поддерживает динамический, объектно-ориентированный, императивный и функциональный стили программирования. Так как язык является интерпретируемым, то производительность и безопасность зависит от конкретной реализации интерпретатора.

1.2.2.5 Анализ фреймворков

Framework – программная платформа, определяющая структуру программной системы. Содержит программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта. Выбор фреймворка зависит от потребностей каждого проекта.

Рассмотрим несколько фреймворков для разработки сетевого приложения.

Фреймворк *Symfony*.

Symfony – отличается от других *PHP*-фреймворков своей надёжностью и долгим нахождением на рынке. Он популярен благодаря соответствию веб-стандартам и шаблонам проектирования *PHP* и возможности работы с различными БД. Он включает в себя встроенную систему тестирования и основан на обширном *MVC*-шаблоне. Это делает его идеальным выбором для крупномасштабных веб-проектов, создаваемых на уровне организаций.

Фреймворк *Yii*.

Yii — это простой, но высокопроизводительный универсальный фреймворк, основанный на компонентной структуре. Он известен благодаря высокой производительности, но, в первую очередь, он знаменит своей простотой, позволяющей использовать код сторонних разработчиков. Его создатели очень

серьёзно подошли к вопросам безопасности. В частности, в него встроены механизмы хеширования паролей, основанные на *bcrypt*, и средства шифрования.

Фреймворк *Laravel*.

Laravel — это бесплатный опенсорсный *PHP*-фреймворк [8]. Чаще всего веб-разработчики выбирают именно этот фреймворк. Одна из главных причин — это возможность разрабатывать сложные веб-приложения проще, быстрее и безопаснее, чем это могут сделать другие фреймворки.

Laravel облегчает решение множества задач, путем не малого количества встроенных классов и методов, что позволит разработчикам писать меньше кода. В список этих задач входят: маршрутизация, кеширование, авторизация, аутентификация и т.д.

В качестве *PHP*-фреймворка был выбран *Laravel* за его немалые преимущества и простоту в использовании.

Также в *Laravel* встроены некоторые функции из фреймворка *Symfony*. В состав этого фреймворка входит интерфейс командной строки *Artisan*. Он предлагает ряд полезных команд, которые помогут при создании приложения и в дальнейшем. Например, запуск миграций для базы данных или создание отдельного класса, с установлением нужных связей.

Плюсы использования *Laravel*:

- совместимость с *MVC* моделью;
- код веб-проекта, *CSS*, *JS*, *HTML*-код страниц разделены в отдельные директории. Фреймворк использует шаблонизатор *Blade*, который позволяет отделить вёрстку от *PHP*-кода;
- удобная маршрутизация, валидация входящих параметров;
- работа с различными БД;
- *Laravel Collections* — это *PHP* массивы, но с продвинутыми возможностями, которые экономят массу времени;
- поддержка *WebSockets* используется для создания интерактивных приложений;
- кэширование файлов маршрутизации, файлов конфигурации, шаблонов,

Изм.	Лист	№ докум.	Подпись	Дата

всё это ускоряет работу фреймворка;

- обширные возможности для тестирования веб-проекта;

В момент разработки была использована последняя версия *Laravel 9.17.0*.

1.3 Реализация

1.3.1 Структура БД

При разработке электронного магазина была использована СУБД *MySQL*.
Далее представлено описание структуры, разработанной БД.

Таблица 1 содержит описание структуры таблицы «Продукты (*products*)».

Таблица 1 — Структура таблицы «Продукты (*products*)»

Название поля	Описание поля	Тип данных
<i>id</i>	Идентификатор продукта	<i>int</i>
<i>category_id</i>	<i>id</i> категории	<i>integer</i>
<i>name</i>	Название продукта	<i>string</i>
<i>code</i>	Уникальный код продукта	<i>string</i>
<i>description</i>	Описание продукта	<i>text</i>
<i>image</i>	Загружаемое изображение продукта	<i>text</i>
<i>price</i>	Стоимость продукта	<i>double</i>

Таблица 2 содержит описание структуры таблицы «Категории (*categories*)».

Таблица 2 — Структура таблицы «Категории (*categories*)»

Название поля	Описание поля	Тип данных
<i>id</i>	Идентификатор категории	<i>int</i>
<i>name</i>	Название категории	<i>string</i>
<i>code</i>	Уникальный код категории	<i>string</i>
<i>description</i>	Описание категории	<i>text</i>
<i>image</i>	Загружаемая изображение категории	<i>text</i>

Таблица 3 содержит описание структуры таблицы «Пользователи (*users*)».

Таблица 3 — Структура таблицы «Пользователи (*users*)»

Название поля	Описание поля	Тип данных
<i>id</i>	Идентификатор пользователя	<i>id</i>
<i>name</i>	Имя пользователя	<i>string</i>
<i>email</i>	Электронный адрес пользователя	<i>string</i>
<i>email_verified_at</i>	Статус проверки электронного адреса	<i>timestamp</i>
<i>password</i>	Зашифрованный пароль пользователя	<i>string</i>
<i>is_admin</i>	Проверка прав доступа у пользователя	<i>tinyInteger</i>

Таблица 4 содержит описание структуры таблицы «Заказы (*orders*)».

Таблица 4 — Структура таблицы «Заказы (*orders*)»

Название поля	Описание поля	Тип данных
<i>id</i>	Идентификатор заказа	<i>id</i>
<i>status</i>	Статус заказа(выполнен/не выполнен)	<i>tinyInteger</i>
<i>name</i>	Имя пользователя	<i>string</i>
<i>phone</i>	Контактный телефон пользователя	<i>string</i>
<i>user_id</i>	<i>id</i> пользователя	<i>integer</i>

Таблица 5 содержит описание структуры таблицы «Заказ продуктов (*order_product*)».

Таблица 5 — Структура таблицы «Заказ продуктов (*order_product*)»

Название поля	Описание поля	Тип данных
<i>id</i>	Идентификатор заказа	<i>id</i>
<i>order_id</i>	<i>id</i> заказа	<i>integer</i>
<i>product_id</i>	<i>id</i> продукта	<i>integer</i>
<i>count</i>	Количество продуктов	<i>integer</i>

1.3.2 Описание классов

Для удобства взаимодействия архитектура *Laravel* разбита на следующие директории:

- директория *\app*, код ядра приложения, *\app* содержит директорию *\Http*, в которой расположены контроллеры, посредники и запросы форм, они задают всю логику обработки запросов, входящих в приложение;
- директория *\database*, содержит миграции и классы для наполнения начальными данными БД;
- директория *\public*, содержит файл *«index.php»*, который является входной точкой для всех запросов, поступающих в приложение, также содержит ресурсы, такие как изображения, *JavaScript*, *CSS*;
- директория *\resources*, содержит все языковые файлы, в которых задается разметка;
- директория *\routes*, содержит все определения маршрутов приложения.

В примере рассмотрим сущность «Пользователь», а именно, как работает программа и что выдает, когда пользователь заходит на основную страницу сайта, эта работа представлена в виде *UML*-диаграммы в приложении А.1.

После открытия страницы пользователям, класс-маршрутизатор *«web.php»*, в котором заключены все маршруты сайта, определяет какая страница открыта и перенаправляет работу программы в класс *«MainController.php»*, этот класс является классом-контроллер, в нем задана основная логика программы, он получает данные из БД *«products»* и *«categories»* и выводит их в файл *«index.blade.php»*, который в свою очередь содержится *blade*-конструкции и *Html* код и выводит информацию на сайте, дальше пользователь может перейти на другую вкладку, тем самым программа возвращается к классу-маршрутизатору и определяет по какому маршруту работать дальше.

Также есть множество других классов, отвечающих за различные функции сайта.

Класс-контроллер *«BasketController.php»* отвечает за логику «Корзины»: добавление и удаление товара из корзины, оформление заказа и сохранения

корзины текущего пользователя, если он решил выйти на главную страницу и посмотреть каталог.

Классы-посредники в директории *App\Http\Middleware* определяют права доступа у пользователя, авторизован ли пользователь и статут корзины, пустая она или нет, например, класс «*CheckIsAdmin.php*» обращается к параметру «*is_admin*» сущности «*users*», проверяя, относится ли пользователь к категории Admin и запрещает заходить на панель администратора.

В директории *resources\views* заключены основные классы, описывающие внешний вид сайта и выводимую на него информацию. Мастер-класс «*master.blade.php*» содержит в себе часть кода, которая может повторяться в других классах, тем самым уменьшается значительная часть кода и упрощается разработка электронного магазина, информация из этого класса передается с помощью *blade*-конструкции «*@extends()*». В директории выше *\auth* находятся классы содержащие, страницы авторизации и панели администратора.

1.4 Описание результата

С помощью консоли разработчика и фреймворка был запущен локальный сервер по адресу «*http://127.0.0.1:8000/*». Результат запуска электронного магазина представлен на рисунке 5.

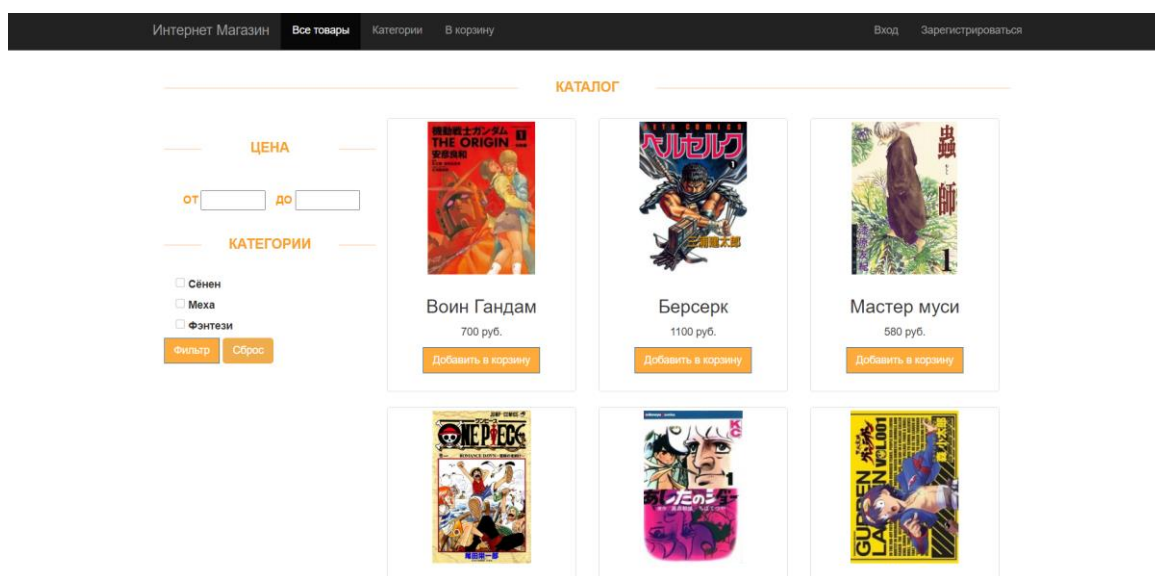


Рисунок 5 — Основная страница сайта

Изм.	Лист	№ докум.	Подпись	Дата

Меню категорий и продукты отдельной категории представлены на рисунках 6 и 7.

Интернет Магазин Все товары Категории В корзину Вход Зарегистрироваться

Сёнен

Сёнен аниме и манга в основном рассчитаны на мальчиков или юношей от 12 до 18 лет. В центре повествования сильный и харизматичный персонаж, который упорно добивается поставленных целей (чаще всего труднодостижимых) в определенных сферах (например, спорт, музыка и т.д.). Герой сенена добрый, справедливый, удачливый парень, который противостоит силам зла. Сюжет в таком аниме развивается стремительно: главному персонажу и его друзьям предстоит испытать множество интересных приключений, смешных происшествий, принять участие в трудных соревнованиях и т.д.

Меха

Это научно-фантастический жанр, отличительной чертой которого являются человекоподобные машины, чаще всего используемые в бою. Главными героями являются персонажи, управляющие этими роботами изнутри. Основной акцент делается на контакте и синхронизации людей и машин. Характерной чертой меха является детальное описание запуска такой машины и взаимодействие с ней.

Фэнтези

В фэнтезийных аниме и манге часто встречаются элементы фольклора, чаще всего западноевропейского, но нередко используется и колорит Востока, и элементы японской и китайской мифологии. Основное действие происходит в вымышленном мире со сказочными существами. Непременный атрибут фэнтези - магия и волшебство. Характерное отличие этого жанра состоит в том, что особенности этого мира и существование вымышленных существ не объясняются с научной точки зрения, а являются нормой.

Рисунок 6 — Страница *categories*

Интернет Магазин Все товары Категории В корзину Вход Зарегистрироваться

Меха

Найдено товара: 3

Это научно-фантастический жанр, отличительной чертой которого являются человекоподобные машины, чаще всего используемые в бою. Главными героями являются персонажи, управляющие этими роботами изнутри. Основной акцент делается на контакте и синхронизации людей и машин. Характерной чертой меха является детальное описание запуска такой машины и взаимодействие с ней.



Воин Гандам
700 руб.
Добавить в корзину



Гуррен-Лаганн
1200 руб.
Добавить в корзину



Рыцари Сидонии
600 руб.
Добавить в корзину

Рисунок 7 — Страница *categories/mekha*

Пользователь можешь войти в свой аккаунт или зарегистрироваться, если такого не имеется. На рисунках 8 и 9 видно окна регистрации и входа в аккаунт.

					ПГУ 1.090301.40.001 ПЗ	Лист
						25
Изм.	Лист	№ докум.	Подпись	Дата		

Регистрация

Имя

Арте́м

E-Mail

artem@mail.ru

Пароль

.....

Подтвердите пароль

.....

Зарегистрироваться

Рисунок 8 — Страница с формой регистрации

Авторизация

E-Mail

artem@mail.ru

Пароль

.....

Войти

Рисунок 9 — Страница с формой авторизации пользователя

После этого, в сущности «users» в поле *is_admin* можно определить, будет ли пользователь относиться к группе *Admin* или нет. 1 – пользователь становится администратором, 2 – обычный пользователь. Данная реализация приведена на рисунке 10.

Изм.	Лист	№ докум.	Подпись	Дата

id	name	email	email_verified_at	is_admin
1	Артем	artem.tkachev.00@mail.ru	NULL	0
2	Артем	artem@mail.ru	NULL	1

Рисунок 10 — Добавление прав администратора отдельному пользователю

Авторизированный пользователь может добавить продукт в корзину и перейти на страницу *basket*, показанная на рисунке 11, тут же можно увеличивать и уменьшать количество товара, после добавления товара в корзину выводится сообщения «Добавлен товар *name*», а после удаление «Удален товар *name*». После чего есть возможность оформить заказ, заполнив форму с данными на рисунке 12.

Интернет Магазин Все товары Категории В корзину Панель администратора Выйти

Добавлен товар Берсерк

Корзина

Оформление заказа

Название	Кол-во	Цена	Стоимость
 Воин Гандам	2 - +	700 руб.	1400 руб.
 Берсерк	4 - +	1100 руб.	4400 руб.
Общая стоимость:			5800 руб.

Оформить заказ

Рисунок 11 — Корзина с товарами

Подтвердите заказ:

Общая стоимость: 5100 руб.

Укажите свои имя и номер телефона, чтобы наш менеджер мог с вами связаться:

Имя:

Номер телефона:

Подтвердите заказ

Рисунок 12 — Форма подстверждения заказа

					ПГУ 1.090301.40.001 ПЗ	Лист
						27
Изм.	Лист	№ докум.	Подпись	Дата		

Если пользователь попытается зайти в пустую корзину, то его перенаправит на главную страницу с соответствующим сообщением, показанное на рисунке 13.

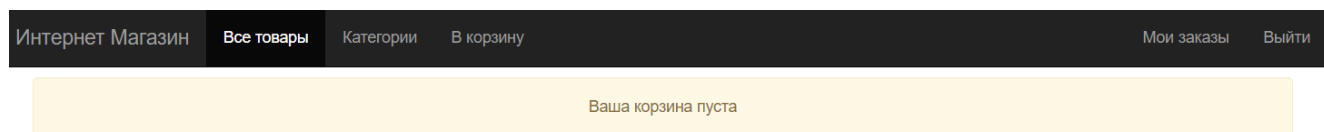
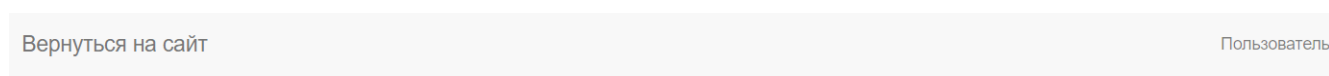


Рисунок 13 — Сообщение о пустой корзине

Также пользователь можешь просматривать свои заказы, сделанные ранее, результат работы на рисунке 14.



Заказы

#	Имя	Телефон	Когда отправлен	Сумма	Действия
25	artem.tkachev.00@mail.ru	12345	21:07 18/06/2022	0 руб.	<button>Открыть</button>

Рисунок 14 — Заказы авторизованного пользователя

Пользователь с правами администратора может посмотреть список продуктов на рисунке 15, а также посмотреть содержание продукта, редактировать его, удалить и добавить новый продукт, заполнив его данные на рисунке 16, которые сразу же заносятся в нашу базу данных. Аналогично можно добавить новую категорию по адресу, процесс добавления новой категории представлен на рисунках 17 и 18.

Товары

#	Код	Название	Категория	Кол-во товарных предложений	Действия
6	Kidou_Senshi_Gundam:_The_Origin	Воин Гандам	Меха		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>
8	Berserk	Берсерк	Фэнтези		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>
9	Mushishi	Мастер муси	Фэнтези		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>
18	One_Piece	Ван-Пис	Сёнен		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>
19	Ashita_no_Joe	Завтрашний Джо	Сёнен		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>
20	Gurren_Lagann	Гуррен-Лаганн	Меха		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>
21	Sidonia_no_Kishi	Рыцари Сидонии	Меха		<div>Открыть</div> <div>Редактировать</div> <div>Удалить</div>

Добавить товар

Рисунок 15 — Список продуктов на странице администратора

Добавить товар

Код:

Gintama

Название:

Гинтама

Категория:

Сёнен ▾

Цена:

790

Описание:

Захватив Землю, пришельцы Аманто запретили ношение мечей. Единственный, в ком ещё жив подлинно японский дух — самоуверенный сластёна Гинтоки Саката. Неуклюжий очкарик Шинпачи нанялся к нему в ученики. Третьим в их команде стала прелестная Кагура из сильнейшего во вселенной клана Ято, а с ней её питомец Садахару — пёсик размером с бегемота, обладающий забавной привычкой грызть головы всем, кто под морду подвернётся. Они называют себя «мастерами на все руки» и

Картинка:

Загрузить

Сохранить

Рисунок 16 — Добавление нового продукта

Категории

#	Код	Название	Действия
1	senen	Сёнен	<button>Открыть</button> <button>Редактировать</button> <button>Удалить</button>
2	mekha	Меха	<button>Открыть</button> <button>Редактировать</button> <button>Удалить</button>
3	fentezi	Фэнтези	<button>Открыть</button> <button>Редактировать</button> <button>Удалить</button>
10	drama	Драма	<button>Открыть</button> <button>Редактировать</button> <button>Удалить</button>

Добавить категорию

Рисунок 17 — Список категорий на странице администратора

Добавить категорию

Код:	<input type="text" value="drama"/>
Название:	<input type="text" value="Драма"/>
Описание категории:	<div>В драматических аниме или манге основной акцент делается на игре сильных чувств, глубоких противоречиях и конфликтах, трудноразрешимых проблемах, которые нередко имеют непоправимые последствия. Драме свойственна трагичность, но может быть и счастливый конец.</div>
Картинка:	<input type="button" value="Загрузить"/>
<input type="button" value="Сохранить"/>	

Рисунок 18 — Добавление новой категории

Добавленные категории и продукты автоматически заносятся в базы данных «products», показанную на рисунке 19 и «categories», показанную на рисунке 20.

id	category_id	name	code	description	image	price	created_at	updated_at
6	2	Воин Гандан	Kidou_Senshi_Gund...	Лучший пилот Зеона, Чар Азнабль, считал, ...	produc...	700	NULL	2022-06-19 22:58:32
8	3	Берсерк	Berserk	Наёмник Гатс, по прозвищу Чёрный Мечник...	produc...	1100	NULL	2022-06-19 22:54:38
9	3	Мастер муси	Mushishi	В мире всё закономерно — спускаясь по вид...	produc...	580	NULL	2022-06-19 22:55:51
18	1	Ван-Пис	One_Piece	путешествие — курс на Гранд Лайн и Ван-Пис...	produc...	800	2022-06-19 22:44:07	2022-06-19 22:44:07
19	1	Завтрашний Джо	Ashita_no_Joe	Джо Ябуки — сирота-беспризорник, котор...	produc...	700	2022-06-19 22:47:01	2022-06-19 22:47:01
20	2	Гуррен-Лаганн	Gurren_Lagann	В их закрытой подземной деревне, Камина ...	produc...	1200	2022-06-19 22:50:24	2022-06-19 22:50:44
21	2	Рыцари Сидонии	Sidonia_no_Kishi	«Сидония» — космическое судно, Ноев ков...	produc...	600	2022-06-19 22:53:13	2022-06-19 22:53:13
22	1	Гинтама	Gintama	Захватив Землю, пришельцы Аманто запре...	produc...	790	2022-06-20 18:36:07	2022-06-20 18:36:07

Рисунок 19 — База данных «products»

id	name	code	description	created_at	updated_at
1	Сёнен	senen	Сёнен аниме и манга в основн...	NULL	2022-06-19 23:10:00
2	Меха	mekha	Это научно-фантастический ж...	NULL	2022-06-19 23:10:43
3	Фэнтези	fentezi	В фэнтезийных аниме и манге ...	NULL	2022-06-19 23:11:05
10	Драма	drama	В драматических аниме или ма...	2022-06-20 18:40:44	2022-06-20 18:40:44
NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 20 — База данных «categories»

Каждое поле в добавлении нового продукта и категории имеет статус обязательного, а поле «code» должно быть уникальным и не совпадать с другими кодами, добавленными ранее. Соответствующие предупреждения для администратора показаны на рисунке 21 при добавлении нового продукта и на рисунке 22 при добавлении категории.

Добавить товар

Код:	Такой код уже существует
	<input type="text" value="Berserk"/>
Название:	Поле "Название" обязательно для заполнения
	<input type="text" value="Berserk"/>
Категория:	Сёнен ▼
Цена:	Поле "Цена" обязательно для заполнения
	<input type="text"/>
Описание:	Поле "Описание категории" обязательно для заполнения
	<input type="text"/>

Рисунок 21 — Сообщения об ошибках при добавлении продукта

Добавить категорию

Код:

Такой код уже существует

senen

Название:

Поле "Название" обязательно для заполнения

Описание категории:

Поле "Описание категории" обязательно для заполнения

Рисунок 22 — Сообщения об ошибках при добавлении категории

2 Безопасность жизнедеятельности

2.1 Расчет эквивалентного уровня звука на рабочих местах в компьютерных залах, офисах и других помещениях, где установлены ПЭВМ

В гигиенической практике шумом называют всякий нежелательный для человека звук. Под его воздействием увеличивается утомляемость, ухудшается восприятие звуковых сигналов и разборчивость речи. У человека может возникать головокружение при выполнении всех видов работ или затрудненное зрение; возникают различные профессиональные заболевания — тугоухость, гипертония.

Любое физическое явление представляет собой распространяющееся механическое колебательное движение части упругой среды с малыми амплитудами. Источником его является любое колеблющееся тело, выведенное из устойчивого состояния какой-либо внешней силой и перемещавшееся вокруг своего положения равновесия в периоды циклического разряжения или уплотнения. Этот процесс может распространяться последовательно на смежные частицы в виде волны λ .

$$\lambda = \frac{c}{f} = cT, \quad (1)$$

где c — скорость звука в среде,

f — частота колебаний,

T — период колебаний.

При распространении звуковой волны происходит перенос энергии в пространстве, называемом звуковым полем. Общее количество энергии, которое источник звука излучает в окружающее пространство называется звуковой мощностью источника (W). Реальные звуковые мощности некоторых источников звука характеризуют следующие приблизительные цифры (речь идет о порядке величин):

- шепот — 10^{-9} Вт;

- обычный разговор — 10^{-5} Вт;

- крик – 10^{-3} Вт;
- цепная пила по дереву – 1 Вт;
- большой оркестр – 10 Вт;
- 4 турбореактивный самолетный двигатель – 10^4 Вт;
- стартовый двигатель мощной космической ракеты – 10^8 Вт.

При оценке шума в какой-либо точке звукового поля (например, на рабочем месте) интерес представляет не общая акустическая мощность источника звука, и даже его часть. Часть общей мощности источника шума, приходящаяся на единицу площади заданного поля и расположенной перпендикулярно распространяющейся звуковой волне, называется интенсивностью звука (J).

Непосредственное измерение интенсивности звука связано с большими техническими трудностями, и в настоящее время нет приборов для непосредственного измерения этого параметра. При помощи акустических приборов (например, шумомеров) можно измерить звуковое давление (P) в зависимости от интенсивности звука (J):

$$J = \frac{P^2}{\rho a}, \quad (2)$$

где ρ – плотность среды,

a – скорость звука в среде.

Звуковым давлением называется разность между мгновенным значением полного давления в какой-либо точке звукового поля и средним давлением, которое наблюдается при невозмущенной среде («атмосферное давление»). Поскольку в течение полного колебательного цикла звуковое давление меняется от нуля до отрицательного максимума, а затем вновь возвращается к нулю и т.д. (в пределах полной цикличности):

$$P = \sqrt{P^2} = \sqrt{\frac{1}{T} \int_0^T P^2(\tau) d\tau}, \quad (3)$$

где P – означает осреднение во времени,

T – продолжительность цикла.

В органе слуха человека такое осреднение происходит примерно за 30 —

100 мс и воспринимается человеком как специфичный звуковой сигнал, если частоту колебаний в диапазонах 16 Гц — 20 кГц. Колебания с частотой менее 16 Гц и более 20 кГц называются инфразвуком. Интенсивность звуков и частота шума, с которыми приходится иметь дело на практике (как в теории), могут изменяться или меняться по бесконечному диапазону. В этих условиях очень удобно использовать логарифмическую шкалу, которая позволяет значительно уменьшить диапазон измеряемых величин. Применению логарифмической шкалы в акустике способствовало обстоятельство, что громкость звуков при оценке на слух увеличивается примерно пропорционально его коэффициенту интенсивности или звука.

Основными характеристиками шума являются уровни звукового давления и уровни интенсивности звука, определяемые по формулам (2,3), дБ,

$$L_p = 20 \lg \frac{J}{J_0},$$

$$L_p = 20 \lg \frac{P}{P_0},$$

где P_0 и J_0 – пороговые (опорные) значения звукового давления и интенсивности шума на частоте 1000 Гц. Они стандартизованы и равны соответственно $2 \cdot 10^{-5}$ Па и 10^{-12} Вт/м². При нормальных атмосферных условиях $L_p = L_J$, что очень удобно при расчетах и акустических измерениях. При условиях, отличных от нормальных, связь между L_p и L_J определяется выражением:

$$L_J = L_p + 10 \lg \frac{p_0 c_0}{\rho c},$$

где p_0 и c_0 – плотность воздуха и скорость звука при нормальных атмосферных условиях; ρ и c – то же при условиях, отличных от нормальных. В производственном помещении обычно бывает несколько источников шума. Суммарный уровень звукового давления нескольких различных источников звука определяется по формуле, дБ,

$$L = 10 \lg [10^{(L_1/10)} + 10^{(L_2/10)} + \dots + 10^{(L_n/10)}],$$

где $L_1, L_2 \dots L_n$ – уровни звукового давления, создаваемые каждым из источников

звука в исследуемой точке пространства.

Суммарный уровень шума от одинаковых по своему уровню источников определяется по формуле

$$L_{\Sigma} = L_i + 10 \lg n,$$

где L_i – уровень звукового давления одного источника, дБ;

n – количество источников шума.

Например, два одинаковых источника совместно создадут уровень на 3 дБ больше, чем каждый источник. Суммарный уровень шума от двух различных по своему уровню источников можно определить по формуле

$$L_{\Sigma} = L_{\max} + \Delta L,$$

где $L_{\max}$ – максимальный уровень звукового давления одного из двух источников,

ΔL – поправка, зависящая от разности между максимальным и минимальным уровнем звукового давления в соответствии с приведенными ниже данными.

В результате сравнения, полученного по данным этой таблицы, можно определить суммарный уровень звукового давления нескольких различных источников звука. По разности двух уровней L_1 и L_2 по таблице 5 определяют добавку щёлкнувшей к большему уровню L_1 . Уровень $L_{1,2}$ суммируется с уровнем L_3 и получают уровень $L_{\text{сум}}$.

Метод расчета применяется в тех случаях, когда есть данные об уровне и продолжительности воздействия шума на рабочем месте или различных помещениях. Эквивалентный уровень звука рассчитывается с использованием поправок на время действия каждого уровня звука, определяемых по таблице 6.

Расчет производится следующим образом. В каждом измеренном уровне звука добавляется поправка в таблице, соответствующая его времени действия (в часах или процентах от общего срока). Затем полученные уровни звука складываются в соответствии с описанным выше.

Пример расчета.

Уровни шума за 8-часовую рабочую смену составляли 80, 86 и 94 дБА в

течение 5, 2 и 1 ч соответственно. Этим значениям времени соответствуют поправки из предыдущей таблицы, равные -2, -6, -9 дБ. Складывая их с уровнями шума, получаем 78, 80, 85 дБА. Теперь, используя таблицу 2 значений поправок ΔL , складываем эти уровни попарно:

$$l_{1,2} = L_2 + \Delta L = 80 + 2,2 = 82 \text{ дБА.}$$

Таблица 6 — Значение поправки ΔL при сложении уровней шума

Разность слагаемых уровней $L_1 - L_2$, дБ	Добавка ΔL , прибавляемая к большему из уровней L_1 , дБ
0	3,0
1	2,5
2	2,2
3	1,8
4	1,5
5	1,2
6	1,0
7	0,8
8	0,6
10	0,4

$$L_{1,2,3} = l_3 + \Delta L = 85 + 1,8 = 86,8 \text{ дБА.}$$

Округляя, получаем окончательный эквивалентный уровень шума 87 дБА. При исследовании шумов можно обнаружить, что воздействие этих звуков равно воздействию шума с постоянным уровнем 87 дБА в течение 8 ч. При исследовании шумов обычно весь слышимый диапазон звуковых колебаний по частоте разбивается на отдельные полосы, каждая из которых характеризуется граничными частотами: нижней (f_n), верхней (f_v) и средней (f_{cp}). За среднюю частоту полосы принято принимать среднегеометрическую частоты, которая определяется по формуле:

$$f_{\text{ср}} = \sqrt{f_{\text{н}} f_{\text{в}}}.$$

Таблица 7 — Значение поправок на время действия шума

Поправка в дБ	Время	
	в часах	в %
0	8	100
-0,6	7	88
-1,2	6	75
-2	5	62
-3	4	50
-4,2	3	38
-6	2	25
-9	1	12
-12	0,5	6
-15	15	3
-20	5	1

Чаще всего применяются октавные и третьоктавные полосы. Октавная полоса (октава) – это такая частотная полоса, в которой верхняя граничная 9 частота ($f_{\text{в}}$) в два раза больше нижней частоты ($f_{\text{н}}$). В третьоктавной полосе это соотношение равно 1,26. При гигиенической оценке шума и его нормировании акустический диапазон частот разделяют на девять октавных полос со среднегеометрическими частотами 31,5; 63; 125; 250; 500; 1000; 2000; 4000 и 8000 Гц. Граничные частоты для этих полос соответственно равны 22–44 Гц., 44–88 Гц., 88–177 Гц., 177–355 Гц., 355–710 Гц., 710–1420 Гц., 1420–2840 Гц., 2840–5680 Гц., 5680–11360 Гц.

В качестве одночисловой характеристики шума применяется оценка уровня звука в децибелах (A), получаемая посредством измерения звукового сигнала на характеристике «А» чувствительности шумомера. С помощью специальных

фильтров характеристика «А» чувствительности шумомеров подобрана таким образом, чтобы между субъективной реакцией человека и уровнем звукового давления существовало соответствие. Иными словами, характеристика «А» шумомеров хорошо имитирует чувствительность человеческого уха во всем звуковом диапазоне.

					ПГУ 1.090301.40.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		39

Заключение

В ходе выполнения данной работы была изучена методология проектирования сетевого приложения «бизнес-потребители» для электронного магазина и приобретены навыки работы с фрейворком *Laravel*. При реализации поставленной задачи были получены следующие результаты:

- разработана структура базы данных для СУБД *MySQL*;
- разработан фронтенд электронного магазина;
- разработан и отлажен бэкенд для электронного магазина

Сетевое приложение обладает адаптивной версткой и кроссбраузерностью, для запуска на любых устройствах.

					ПГУ 1.090301.40.001 ПЗ	Лист
						40
Изм.	Лист	№ докум.	Подпись	Дата		

Список использованных источников

1. Эрик Ф, Элизабет Ф. Паттерны проектирования //Питер.— 2011. – с. 565-573.
2. К. Дж. Дейт. Введение в системы баз данных. 8-е издание //Вильямс (ГПП Печ. Двор).— 2005.— с. 1328.
3. Николай Прохоренок, Владимир Дронов. *HTML, JavaScript, PHP и MySQL*. Джентльменский набор Web-мастера //БХВ-Петербург.— 2015.— с. 20-348.
4. Марк Саммерфилд. Программирование на Go. Разработка приложений XXI века. //ДМК Пресс.— 2016.— с. 34-251.
5. Марк Саммерфилд. Python на практике //ДМК Пресс.— 2014.— с. 120-310.
6. Шилдт Г. Java 8: руководство для начинающих //Вильямс.— 2018.— с. 149- 348.
7. Эрик Фримен, Элизабет Робсон. Изучаем программирование на *JavaScript* //Питер.— 2018.— с.125-340.
8. *Laravel* — php-фреймворк нового поколения [Электронный ресурс]/ – Режим доступа: <https://laravel.ru/>, свободный (дата обращения 01.06.2022 г.).

Приложение А

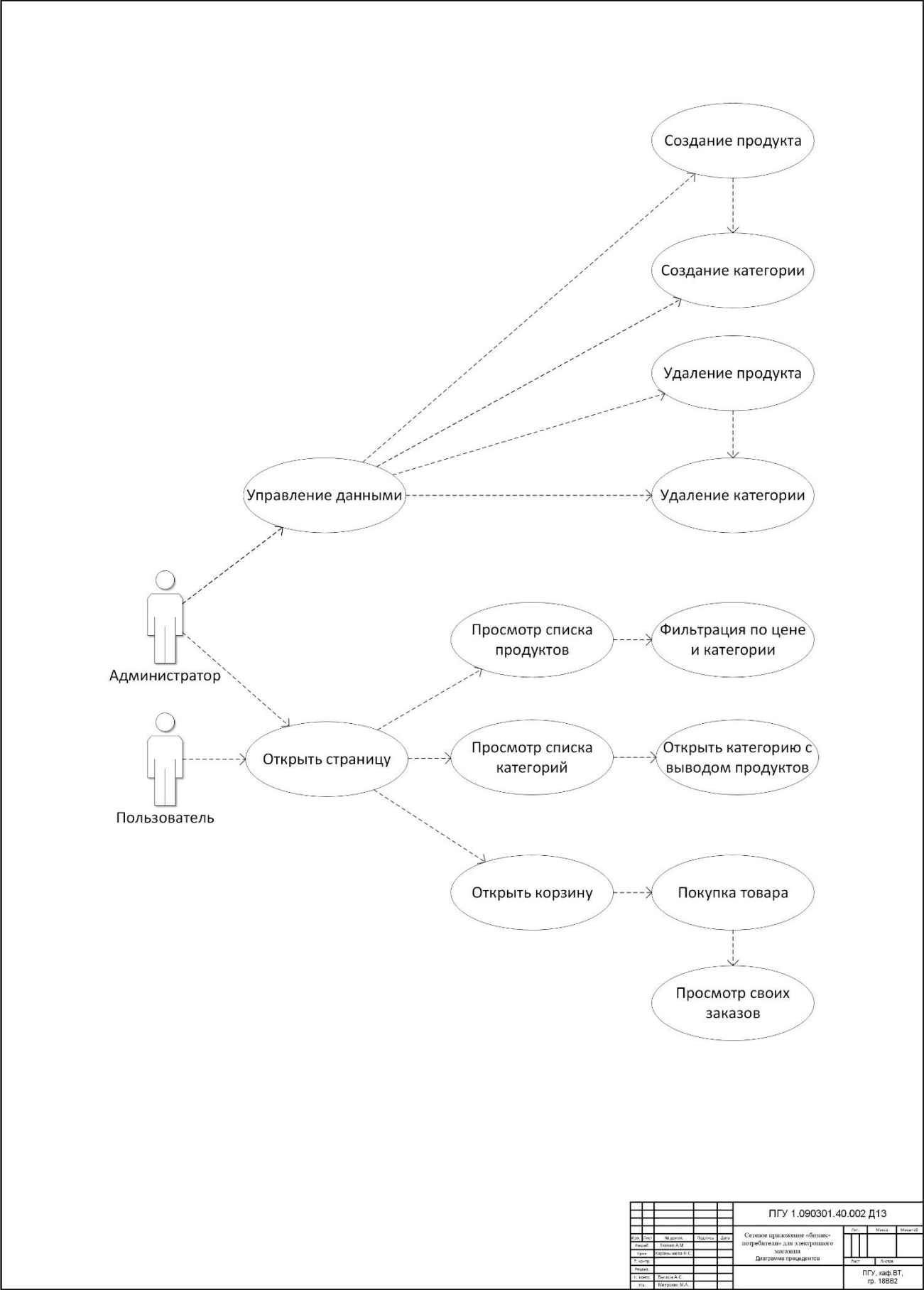
(обязательное)

Сетевое приложение «бизнес-потребители» для электронного магазина

UML-диаграммы

					ПГУ 1.090301.40.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		42

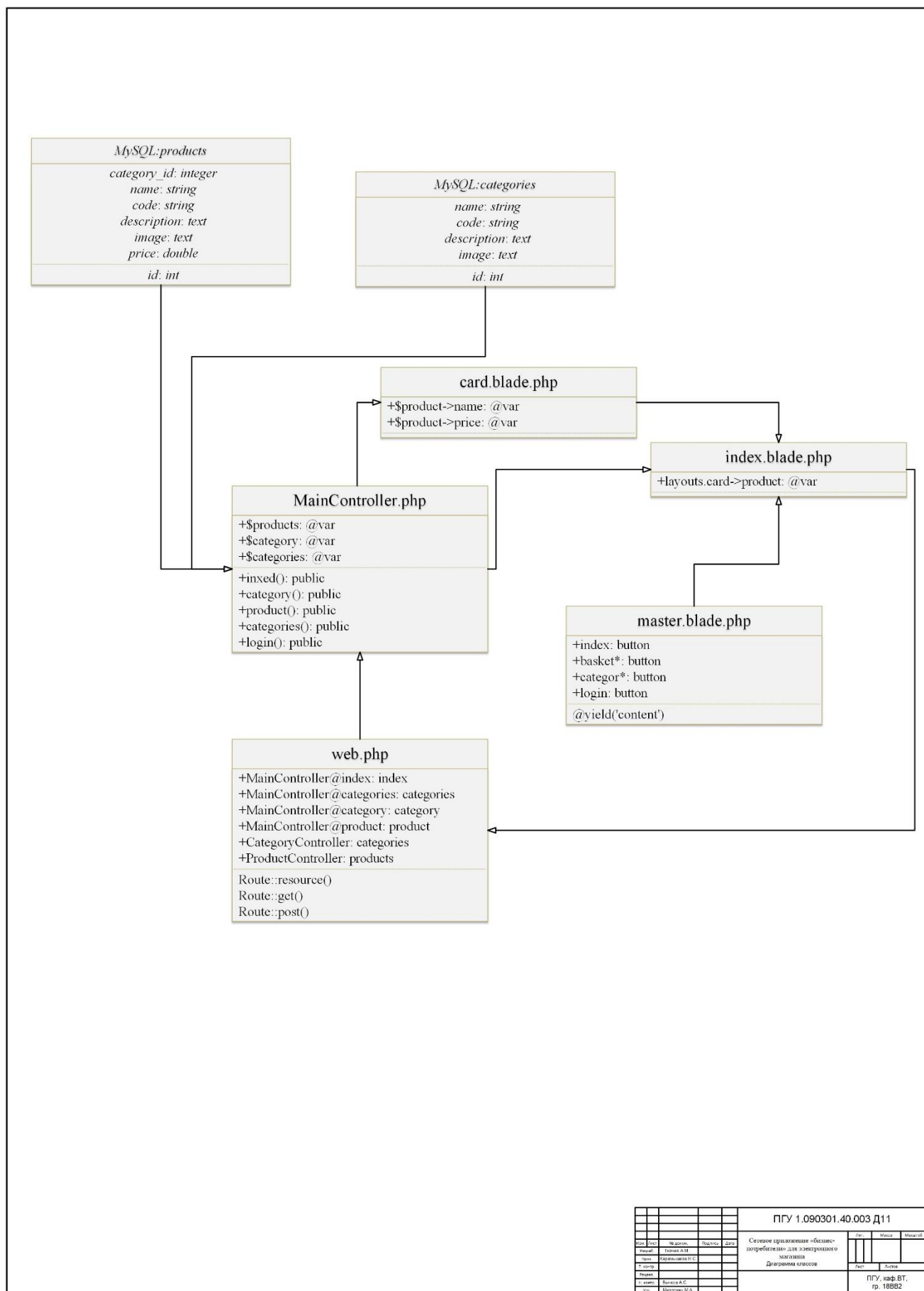
Приложение А.1 – Диаграмма прецедентов



					ПГУ 1.090301.40.002 Д13				
Имя	Фамилия	Имя	Фамилия	Подпись	Дата	Система управления «Бизнес-потребитель» для электронного магазина			
Имя	Фамилия	Имя	Фамилия	Подпись	Дата	Диаграмма прецедентов			
Имя	Фамилия	Имя	Фамилия	Подпись	Дата	ПГУ, каф. ВТ, гр. 188В2			
Имя	Фамилия	Имя	Фамилия	Подпись	Дата				

Изм.	Лист	№ докум.	Подпись	Дата

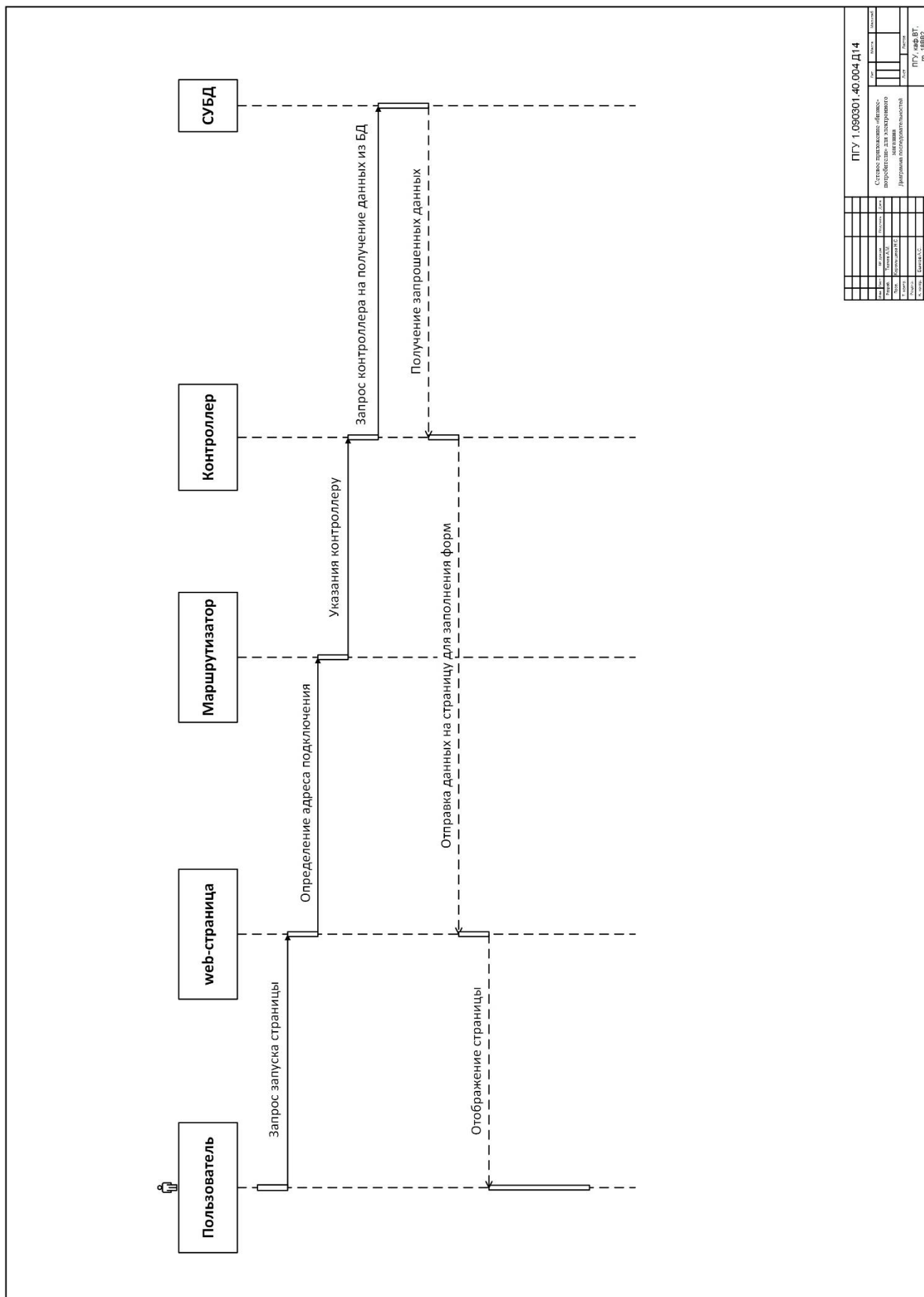
Приложение А.2 – Диаграмма классов



ПГУ 1.090301.40.003 Д11									
Изм.	Лист	№ докум.	Подпись	Дата	Сетевое приложение «Бизнес» потребитель для экстерниции «Молочная» Диаграмма классов				
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.					
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.
Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.	Исполн.

Изм.	Лист	№ докум.	Подпись	Дата

Приложение А.3 – Диаграмма последовательностей

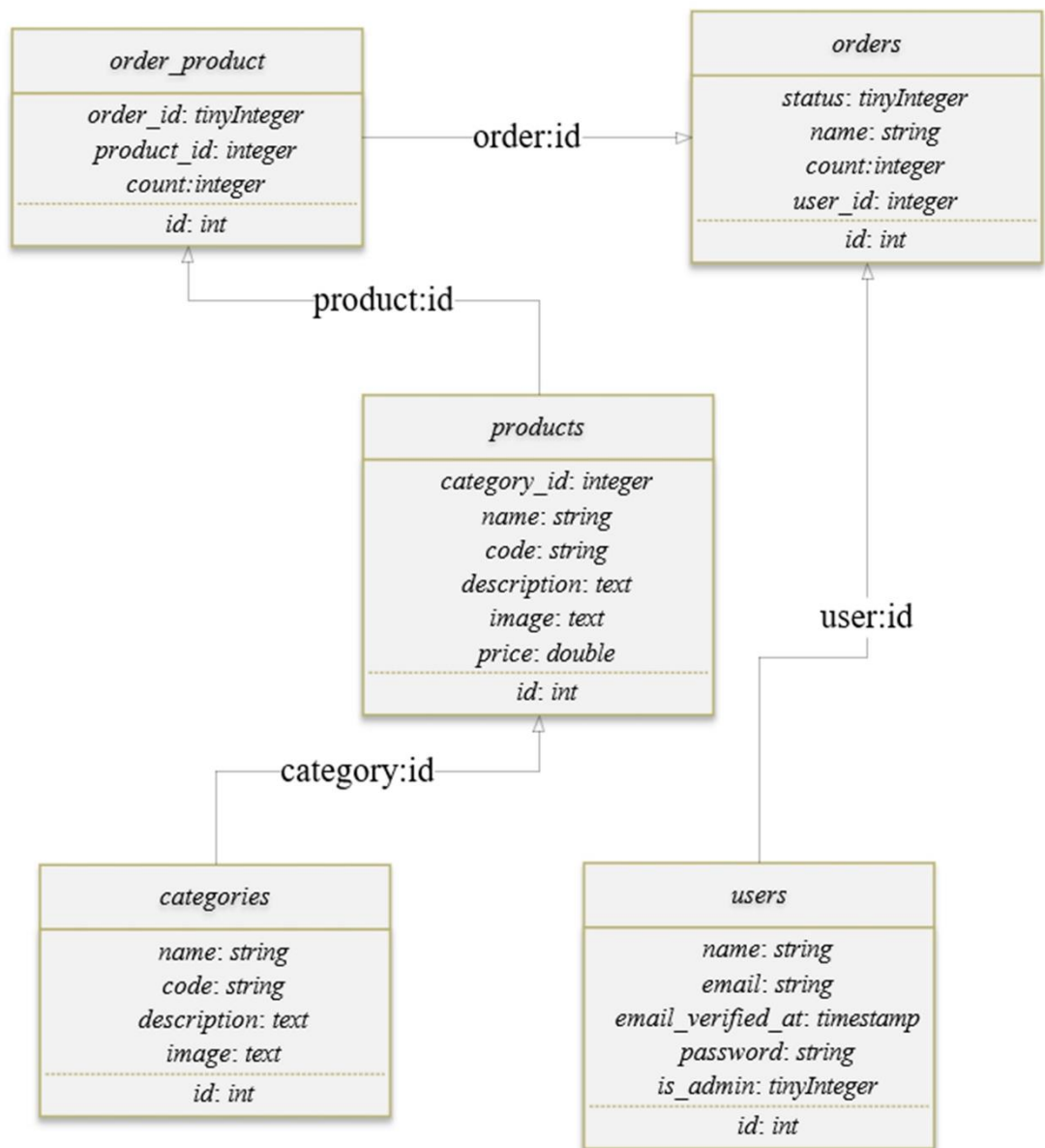
[illegible]

Приложение Б

(обязательное)

Сетевое приложение «бизнес-потребители» для электронного магазина

Модель базы данных



ПГУ 1.090301.40.005 Д7				
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата
Имя	Фамилия	Имя	Фамилия	Дата

Приложение В

(обязательное)

Сетевое приложение «бизнес-потребители» для электронного магазина

Листинг программы

					ПГУ1 090301.40.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		53

web.php

```
<?php
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Route;
Auth::routes([
    'reset' => false,
    'confirm' => false,
    'verify' => false,
]);
Route::get('/logout', 'Auth\LoginController@logout')->name('get-logout');
Route::middleware(['auth'])->group (function () {
    Route::group([
        'prefix' => 'person',
        'namespace' => 'Person',
        'as' => 'person.',
    ], function() {
        Route::get('/orders', 'OrderController@index')->name('orders.index');
        Route::get('/orders/{order}', 'OrderController@show')->name('orders.show');
    });
    Route::group([
        'middleware' => 'auth',
        'namespace' => 'Admin',
        'prefix' => 'admin',
    ], function() {
        Route::group(['middleware' => 'is_admin'], function() {
            Route::get('/orders', 'OrderController@index')->name('home');
            Route::get('/orders/{order}', 'OrderController@show')->name('orders.show');
        });
        Route::resource('categories', 'CategoryController');
```

```

        Route::resource('products', 'ProductController');
    });
});
Route::get('/', 'MainController@index')->name('index');
Route::get('/categories', 'MainController@categories')->name('categories');
Route::group(['prefix' => 'basket'], function() {
    Route::post('/add/{id}', 'BasketController@basketAdd')->name('basket-add');

    Route::group([
        'middleware' => 'basket_not_empty',
    ], function () {
        Route::get('/', 'BasketController@basket')->name('basket');
        Route::get('/place', 'BasketController@basketPlace')->name('basket-place');
        Route::post('/remove/{id}', 'BasketController@basketRemove')->name('basket-remove');
        Route::post('/place', 'BasketController@basketConfirm')->name('basket-confirm');
    });
});
Route::get('/{category}', 'MainController@category')->name('category');
Route::get('/{category}/{product?}', 'MainController@product')->name('product');

```

MainController.php

```

<?php
namespace App\Http\Controllers;
use App\Filters\ProductFilter;
use App\Models\Category;
use App\Models\Product;
use Illuminate\Http\Request;

```

					ПГУ1 090301.40.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		55

```

class MainController extends Controller
{
    public function index(Request $request, ProductFilter
$filter) {
        $productsQuery = Product::query();
        if($request->filled('price_from')) {
            $productsQuery->where('price', '>=', $request-
>price_from);
        }
        if($request->filled('price_to')) {
            $productsQuery->where('price', '<=', $request-
>price_to);
        }
        $products = Product::filter($filter)->paginate(6);
        $category = Category::get();
        $categories = Category::get();
        return view('index', compact('products', 'category',
'categories'));
    }
    public function categories() {
        $categories = Category::get();
        return view('categories', compact('categories'));
    }
    public function category($code) {
        $category = Category::where('code', $code)->first();
        return view('category', compact('category'));
    }
    public function product($category, $product = null) {
        return view('product', ['product' => $product]);
    }
    public function login() {
        return view('login');
    }
}

```

BasketController.php

```
<?php
namespace App\Http\Controllers;
use App\Models\Order;
use App\Models\Product;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
class BasketController extends Controller
{
    public function basket()
    {
        $orderId = session('orderId');
        if(!is_null($orderId)){
            $order = Order::findOrFail($orderId);
        }
        return view('basket', compact('order'));
    }
    public function basketConfirm(Request $request)
    {
        $orderId = session('orderId');
        if (is_null($orderId)) {
            return redirect()->route('index');
        }
        $order = Order::find($orderId);
        if ($order->status != 0) {
            $success = false;
        }
        $order->name = $request->name;
        $order->phone = $request->phone;
        $order->status = 1;
        $order->save();
        session()->forget('orderId');
        $success = true;
        if ($success) {
```

					ПГУ1 090301.40.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		57

```

        session()->flash('success', 'Ваш заказ принят в
обработку!');
    } else {
        session()->flash('warning', 'Ошибка при оформлении
заказа!');
    }
    return redirect()->route('index');
}
public function basketPlace()
{
    $orderId = session('orderId');
    if (is_null($orderId)) {
        return redirect()->route('index');
    }
    $order = Order::find($orderId);
    return view('order', compact('order'));
}
public function basketAdd($productId)
{
    $orderId = session('orderId');
    if (is_null($orderId)) {
        $order = Order::create();
        session(['orderId' => $order->id]);
    } else {
        $order = Order::find($orderId);
    }
    if($order->products->contains($productId)) {
        $pivotRow = $order->products()->where('product_id',
$productId)->first()->pivot;
        $pivotRow->count++;
        $pivotRow->update();
    } else {
        $order->products()->attach($productId);
    }
}

```

```

        if(Auth::check()) {
            $order->user_id = Auth::id();
            $order->save();
        }
        $product = Product::find($productId);
        session()->flash('success', 'Добавлен товар ' .
$product->name);
        return redirect()->route('basket');
    }
    public function basketRemove($productId)
    {
        $orderId = session('orderId');
        if (is_null($orderId)) {
            return redirect()->route('basket');
        }
        $order = Order::find($orderId);
        if($order->products->contains($productId)) {
            $pivotRow = $order->products()->where('product_id',
$productId)->first()->pivot;
            if ($pivotRow->count < 2) {
                $order->products()->detach($productId);
            } else {
                $pivotRow->count--;
                $pivotRow->update();
            }
        }
        $product = Product::find($productId);
        session()->flash('warning', 'Удален товар ' . $product-
>name);
        return redirect()->route('basket');
    }
}

```

ProductController.php

<?php

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 59 |

```

namespace App\Http\Controllers\Admin;
use App\Models\Category;
use App\Http\Controllers\Controller;
use App\Models\Product;
use App\Http\Requests\ProductRequest;
use Illuminate\Support\Facades\Storage;
class ProductController extends Controller
{
    public function index()
    {
        $products = Product::paginate(8);
        return view('auth.products.index', compact('products'));
    }
    public function create()
    {
        $categories = Category::get();
        return view('auth.products.form',
compact('categories'));
    }
    public function store(ProductRequest $request)
    {
        $params = $request->all();
        unset($params['image']);

        if ($request->has('image')) {
            $path = $request->file('image')->store('products');
            $params['image'] = $path;
        }

        Product::create($params);
        return redirect()->route('products.index');
    }
    public function show(Product $product)

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 60 |

```

    {
        return view('auth.products.show', compact('product'));
    }
    public function edit(Product $product)
    {
        $categories = Category::get();
        return view('auth.products.form', compact('product',
'categories'));
    }
    public function update(ProductRequest $request, Product
$product)
    {
        $params = $request->all();
        unset($params['image']);

        if ($request->has('image')) {
            Storage::delete($product->image);
            $path = $request->file('image')->store('products');
            $params['image'] = $path;
        }

        $product->update($params);
        return redirect()->route('products.index');
    }
    public function destroy(Product $product)
    {
        $product->delete();
        return redirect()->route('products.index');
    }
}

```

OrderController.php

<?php

namespace App\Http\Controllers\Admin;

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 61 |

```

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use App\Models\Order;
class OrderController extends Controller
{
    public function index()
    {
        $orders = Order::where('status', 1)->get();
        return view('auth.orders.index', compact('orders'));
    }
    public function show(Order $order)
    {
        return view('auth.orders.show', compact('order'));
    }
}

```

CategoryController.php

```

<?php
namespace App\Http\Controllers\Admin;
use App\Http\Controllers\Controller;
use App\Models\Category;
use App\Http\Requests\CategoryRequest;
use Illuminate\Support\Facades\Storage;

class CategoryController extends Controller
{
    public function index()
    {
        $categories = Category::get();
        return view('auth.categories.index',
compact('categories'));
    }
    public function create()
    {

```

| | | | | | | |
|------|------|----------|---------|------|------------------------------|------|
| | | | | | <i>ПГУ1 090301.40.001 ПЗ</i> | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 62 |

```

        return view('auth.categories.form');
    }

    public function store(CategoryRequest $request)
    {

        $params = $request->all();
        unset($params['image']);

        if ($request->has('image')) {
            $path = $request->file('image')->store('categories');
            $params['image'] = $path;
        }

        Category::create($params);
        return redirect()->route('categories.index');
    }

    public function show(Category $category)
    {
        return view('auth.categories.show',
compact('category'));
    }

    public function edit(Category $category)
    {
        return view('auth.categories.form',
compact('category'));
    }

    public function update(CategoryRequest $request, Category
$category)
    {

        $params = $request->all();
        unset($params['image']);

        if ($request->has('image')) {

```

```

        Storage::delete($category->image);
        $path = $request->file('image')->store('categories');
        $params['image'] = $path;
    }

    $category->update($params);
    return redirect()->route('categories.index');
}

public function destroy(Category $category)
{
    $category->delete();
    return redirect()->route('categories.index');
}
}

```

RegisterController.php

```

<?php
namespace App\Http\Controllers\Auth;
use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use App\Models\User;
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use Illuminate\Support\Facades\Auth;

class RegisterController extends Controller
{
    use RegistersUsers;

    protected function redirectTo()
    {
        if (Auth::user()->isAdmin()) {
            return route('home');
        } else {

```

```

        return route('person.orders.index');
    }
}

public function __construct()
{
    $this->middleware('guest');
}

protected function validator(array $data)
{
    return Validator::make($data, [
        'name' => ['required', 'string', 'max:255'],
        'email' => ['required', 'string', 'email',
'max:255', 'unique:users'],
        'password' => ['required', 'string', 'min:8',
'confirmed'],
    ]);
}

protected function create(array $data)
{
    return User::create([
        'name' => $data['name'],
        'email' => $data['email'],
        'password' => Hash::make($data['password']),
    ]);
}
}

```

LoginController.php

<?php

```

namespace App\Http\Controllers\Auth;
use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Foundation\Auth\AuthenticatesUsers;
use Illuminate\Support\Facades\Auth;

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 65 |

```

class LoginController extends Controller
{
    use AuthenticatesUsers;
    protected function redirectTo()
    {
        if (Auth::user()->isAdmin()) {
            return route('home');
        } else {
            return route('person.orders.index');
        }
    }
    public function __construct()
    {
        $this->middleware('guest')->except('logout');
    }
}

```

BasketIsNotEmpty.php

```

<?php
namespace App\Http\Middleware;
use Closure;
use Illuminate\Http\Request;
use App\Models\Order;

class BasketIsNotEmpty
{
    public function handle(Request $request, Closure $next)
    {
        $orderId = session('orderId');

        if (!is_null($orderId)) {
            $order = Order::findOrFail($orderId);

            if($order->products->count() > 0) {
                return $next($request);
            }
        }
    }
}

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 66 |

```

        }

    }

    session()->flash('warning', 'Ваша корзина пуста');
    return redirect()->route('index');
}
}

```

CheckIsAdmin.php

```

<?php
namespace App\Http\Middleware;
use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class CheckIsAdmin
{
    public function handle(Request $request, Closure $next)
    {
        $user = Auth::user();

        if (!$user->isAdmin()) {
            session()->flash('warning', 'У вас не прав
администратора');
            return redirect()->route('index');
        }
        return $next($request);
    }
}

```

RedirectIfAuthenticated.php

```

<?php
namespace App\Http\Middleware;
use App\Providers\RouteServiceProvider;
use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 67 |

```

class RedirectIfAuthenticated
{
    public function handle(Request $request, Closure $next,
...$guards)
    {
        $guards = empty($guards) ? [null] : $guards;
        foreach ($guards as $guard) {
            if (Auth::guard($guard)->check()) {
                return redirect()->route('home');
            }
        }
        return $next($request);
    }
}

```

CategoryRequest.php

```

<?php
namespace App\Http\Requests;
use Illuminate\Foundation\Http\FormRequest;

class CategoryRequest extends FormRequest
{
    public function authorize()
    {
        return true;
    }

    public function rules()
    {
        $rules = [
            'code' => 'required|unique:categories,code',
            'name' => 'required',
            'description' => 'required',
        ];
        if ($this->route()->named('categories.update')) {

```

```

        $rules['code'] .= ',' . $this->route()-
>parameter('category')->id;
    }
    return $rules;
}
public function messages()
{
    return [
        'required' => 'Поле :attribute обязательно для
заполнения',
        'unique' => 'Такой код уже существует',
        'code.required' => 'Поле "Код" обязательно для
заполнения',
        'name.required' => 'Поле "Название" обязательно для
заполнения',
        'description.required' => 'Поле "Описание категории"
обязательно для заполнения',
    ];
}
}

```

ProductRequest.php

```

<?php
namespace App\Http\Requests;
use Illuminate\Foundation\Http\FormRequest;

class ProductRequest extends FormRequest
{
    public function authorize()
    {
        return true;
    }
    public function rules()
    {
        $rules = [

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 69 |

```

        'code' => 'required|unique:products,code',
        'name' => 'required',
        'description' => 'required',
        'price' => 'required',
    ];

    if ($this->route()->named('products.update')) {
        $rules['code'] .= ', ' . $this->route()-
>parameter('product')->id;
    }

    return $rules;
}

public function messages()
{
    return [
        'required' => 'Поле :attribute обязательно для
заполнения',
        'unique' => 'Такой код уже существует',
        'code.required' => 'Поле "Код" обязательно для
заполнения',
        'name.required' => 'Поле "Название" обязательно для
заполнения',
        'description.required' => 'Поле "Описание категории"
обязательно для заполнения',
        'price.required' => 'Поле "Цена" обязательно для
заполнения',
    ];
}
}

```

User.php

```

<?php
namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 70 |

```

use Illuminate\Notifications\Notifiable;
use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable;

    protected $fillable = [
        'name',
        'email',
        'password',
    ];

    protected $hidden = [
        'password',
        'remember_token',
    ];

    protected $casts = [
        'email_verified_at' => 'datetime',
    ];

    public function isAdmin ()
    {
        return $this->is_admin === 1;
    }

    public function orders()
    {
        return $this->hasMany(Order::class);
    }
}

```

Product.php

```

<?php
namespace App\Models;
use App\Filters\QueryFilter;
use Illuminate\Database\Eloquent\Builder;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 71 |

```

class Product extends Model
{
    protected $fillable = ['name', 'code', 'price',
'category_id', 'description', 'image'];
    public function category()
    {
        return $this->belongsTo(Category::class);
    }
    public function getPriceForCount()
    {
        if(!is_null($this->pivot)) {
            return $this->pivot->count * $this->price;
        }
        return $this->price;
    }
    public function scopeFilter(Builder $builder, QueryFilter
$filter)
    {
        return $filter->apply($builder);
    }
    use HasFactory;
}

```

Order.php

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Order extends Model
{
    public function products()
    {
        return $this->belongsToMany(Product::class)-

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 72 |

```

>withPivot('count')->withTimestamps();
    }
    public function getFullPrice()
    {
        $sum = 0;
        foreach ($this->products as $product) {
            $sum += $product->getPriceForCount();
        }
        return $sum;
    }
    use HasFactory;
}

```

Category.php

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Category extends Model
{
    protected $fillable = ['code', 'name', 'description',
    'image'];
    public function products()
    {
        return $this->hasMany(Product::class);
    }
    use HasFactory;
}

```

AppServiceProvider.php

```

<?php
namespace App\Providers;
use Illuminate\Support\ServiceProvider;
use Illuminate\Support\Facades\Blade;
use Illuminate\Support\Facades\Auth;

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 73 |

```

use Illuminate\Pagination\Paginator;

class AppServiceProvider extends ServiceProvider
{
    public function register()
    {
        //
    }
    public function boot()
    {
        Paginator::useBootstrap();
        Blade::directive('routeactive', function ($route) {
            return "<?php echo Route::currentRouteNamed($route)
? 'class=\"active\"' : ' ' ?>";
        });
        Blade::if('admin', function () {
            return Auth::check() && Auth::user()->isAdmin();
        });
    }
}

```

RouteServiceProvider.php

```

<?php
namespace App\Providers;
use Illuminate\Cache\RateLimiting\Limit;
use Illuminate\Foundation\Support\Providers\RouteServiceProvider
as ServiceProvider;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\RateLimiter;
use Illuminate\Support\Facades\Route;

class RouteServiceProvider extends ServiceProvider
{
    public const HOME = '/home';
    protected $namespace = 'App\Http\Controllers';

```

```

public function boot()
{
    $this->configureRateLimiting();
    $this->routes(function () {
        Route::middleware('api')
            ->prefix('api')
            ->namespace($this->namespace)
            ->group(base_path('routes/api.php'));
        Route::middleware('web')
            ->namespace($this->namespace)
            ->group(base_path('routes/web.php'));
    });
}

protected function configureRateLimiting()
{
    RateLimiter::for('api', function (Request $request) {
        return Limit::perMinute(60)->by($request->user()?->id ?: $request->ip());
    });
}
}

```

master.blade.php

```

package ru.zderev.sport.base;

<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>Интернет Магазин: @yield('title')</title>

    <script src="/js/jquery.min.js"></script>

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 75 |

```

<script src="/js/bootstrap.min.js"></script>
<link href="/css/bootstrap.min.css" rel="stylesheet">
<link href="/css/starter-template.css" rel="stylesheet">
</head>
<body>
<nav class="navbar navbar-inverse navbar-fixed-top">
  <div class="container">
    <div class="navbar-header">
      <a class="navbar-brand"
href="{{route('index')}}">Интернет Магазин</a>
    </div>
    <div id="navbar" class="collapse navbar-collapse">
      <ul class="nav navbar-nav">
        <li @routeactive('index')><a
href="{{route('index')}}">Все товары</a></li>
        <li @routeactive('category*')><a
href="{{route('categories')}}">Категории</a>
</li>
        <li @routeactive('basket*')><a
href="{{route('basket')}}">В корзину</a></li>
      </ul>
      <ul class="nav navbar-nav navbar-right">
        @guest
          <li @routeactive('login')><a
href="{{route('login')}}">Вход</a></li>
          <li @routeactive('register')><a
href="{{route('register')}}">Зарегистрироваться</a></li>
        @endguest
        @auth
          @admin
            <li @routeactive('home')><a
href="{{route('home')}}">Панель администратора</a></li>
          @else
            <li @routeactive('home')><a

```

```

href="{{route('person.orders.index')}}">Мои заказы</a></li>
        @endadmin
        <li @routeactive('get-logout')><a
href="{{route('get-logout')}}">Выйти</a></li>
        @endauth
    </ul>
</div>
</div>
</nav>
<div class="container">
    <div class="starter-template">
        @if(session()->has('success'))
            <p class="alert alert-success">{{ session()-
>get('success') }}</p>
        @endif
        @if(session()->has('warning'))
            <p class="alert alert-warning">{{ session()-
>get('warning') }}</p>
        @endif
        @yield('content')
    </div>
</div>
</body>
</html>

```

card.blade.php

```

<div class="col-sm-6 col-md-4">
    <div class="thumbnail">
        
        <div class="caption">
            <h3>{{ $product->name }}</h3>
            <p>{{ $product->price }} руб.</p>
            <p>
                <form action="{{ route('basket-add', $product) }}"
method="POST">

```

```

        <button type="submit" class="btn btn-primary"
role="button">Добавить в корзину</button>

        <!-- <a href="{{ route('product', [$product-
>category->code, $product->name, $product->code]) }}" class="btn
btn-default" role="button">Подробнее</a> -->

        @csrf
    </form>
</p>
</div>
</div>
</div>

```

index.blade.php

```

@extends('layouts.master')
@section('title', 'Главная')
@section('content')
<h1 class="hr">Каталог</h1>
<div class="index">
    <div class="col-sm-3">
        <form method="GET" action="{{ route('index') }}">
            <div class="filters row">
                <h1 class="hr">Цена</h1>
                <div class="col-sm-12">
                    <label class="price" for="price_from">от
                        <input type="text" name="price_from"
id="price_from" size="6">
                    </label>
                    <label class="price" for="price_to">до
                        <input type="text" name="price_to"
id="price_to" size="6">
                    </label>
                </div>
            </div>
            <div>
                <h1 class="hr">Категории</h1>
                @foreach($categories as $category)

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 78 |

```

        <div>
            <label class="category-filter"
for="{{ $category->id }}">
                <input type="checkbox"
value="{{ $category->id }}" name="{{ $category->id }}"
id="{{ $category->id }}"> {{ $category->name }}</label>
            </div>
        @endforeach
    <div>
        <button type="submit" class="btn btn-
primary">Фильтр</button>
        <a href="{{ route('index') }}" class="btn
btn-warning">Сброс</a>
    </div>
</div>
</div>
</form>
</div>
<div class="col-sm-9">
    <div class="row">
        @foreach($products as $product)
            @include('layouts.card', compact('product'))
        @endforeach
    </div>
</div>
</div>
    {{ $products->links() }}
@endsection

order.blade.php

@extends('layouts.master')
@section('title', 'Оформление заказа')
@section('content')

    <h1>Подтвердите заказ:</h1>

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 79 |

```

<div class="container">
  <div class="row justify-content-center">
    <p>Общая стоимость: <b>{{ $order->getFullPrice() }}
руб.</b></p>
    <form action="{{ route('basket-confirm') }}"
method="POST">
      <div>
        <p>Укажите свои имя и номер телефона, чтобы
наш менеджер мог с вами связаться:</p>
        <div class="container">
          <div class="form-group">
            <label for="name" class="control-
label col-lg-offset-3 col-lg-2">Имя: </label>
            <div class="col-lg-4">
              <input type="text" name="name"
id="name" value="" class="form-control">
            </div>
          </div>
          <br>
          <br>
          <div class="form-group">
            <label for="phone" class="control-
label col-lg-offset-3 col-lg-2">Номер телефона: </label>
            <div class="col-lg-4">
              <input type="text" name="phone"
id="phone" value="" class="form-control">
            </div>
          </div>
        </div>
      </div>
      <br>
      @csrf
      <input type="submit" class="btn btn-success"
value="Подтвердите заказ">
    </div>
  </div>

```

```

        </form>
    </div>
</div>
@endsection

category.blade.php

@extends('layouts.master')
@section('title', 'Категория ' . $category->name)
@section('content')
    <h1>
        {{$category->name}}
    </h1>
    <h2>
        Найдено товара: {{$category->products->count()}}
    </h2>
    <p>
        {{$category->description}}
    </p>
    <div class="row">
        @foreach($category->products as $product)
            @include('layouts.card', compact('product'))
        @endforeach
    </div>
@endsection

```

```

categories.blade.php

@extends('layouts.master')
@section('title', 'Все категории')
@section('content')
    @foreach($categories as $category)
        <div class="panel">
            <a class="link" href="{{route('category', $category-
            >code)}}">
                <!--  -->
            <h2>{{$category->name}}</h2>
        </div>
    @endforeach

```

```

        </a>
    <p>
        {{ $category->description }}
    </p>
</div>
@endforeach
@endsection

```

basket.blade.php

```

@extends('layouts.master')
@section('title', 'Корзина')
@section('content')
    <h1>Корзина</h1>
    <p>Оформление заказа</p>
    <div class="panel">
        <table class="table table-striped">
            <thead>
                <tr>
                    <th>Название</th>
                    <th>Кол-во</th>
                    <th>Цена</th>
                    <th>Стоимость</th>
                </tr>
            </thead>
            <tbody>
                @foreach($order->products as $product)
                    <tr>
                        <td>
                            <a href="{{ route('product', [$product->category->code, $product->code]) }}">
                                
                                {{ $product->name }}
                            </a>
                        </td>

```

```

        <td><span class="badge">{{ $product->pivot-
>count}}</span>

        <div class="btn-group form-inline">
            <form action="{{ route('basket-
remove', $product) }}" method="POST">

                <button type="submit" class="btn
btn-danger" href=""><span class="glyphicon glyphicon-minus"
aria-hidden="true"></span></button>

                @csrf

            </form>

            <form action="{{ route('basket-add',
$product) }}" method="POST">

                <button type="submit" class="btn
btn-success" href=""><span class="glyphicon glyphicon-plus"
aria-hidden="true"></span></button>

                @csrf

            </form>

        </div>

    </td>

    <td>{{ $product->price }} руб.</td>
    <td>{{ $product->getPriceForCount() }}
руб.</td>

</tr>
@endforeach
<tr>
    <td colspan="3">Общая стоимость:</td>
    <td>{{ $order->getFullPrice() }} руб.</td>
</tr>
</tbody>
</table>
<br>
<div class="btn-group pull-right" role="group">
    <a type="button" class="btn btn-success" href="{{
route('basket-place') }}">Оформить заказ</a>

```

```

        </div>
    </div>
@endsection

register.blade.php

@extends('layouts.master')
@section('title', 'Регистраниця')
@section('content')
<div class="col-md-8">
    <div class="card">
        <div class="card-header">Регистраниця</div>
        <div class="card-body">
            <form method="POST" action="{{route('register')}}"
aria-label="Register">
                @csrf
                <div class="form-group row">
                    <label for="name" class="col-md-4 col-form-
label text-md-right">Имя</label>
                    <div class="col-md-6">
                        <input id="name" type="text"
class="form-control" name="name" value="" required autofocus>
                    </div>
                </div>
                <div class="form-group row">
                    <label for="email" class="col-md-4 col-form-
label text-md-right">E-Mail</label>
                    <div class="col-md-6">
                        <input id="email" type="email"
class="form-control" name="email" value="" required>
                    </div>
                </div>
                <div class="form-group row">
                    <label for="password" class="col-md-4 col-
form-label text-md-right">Пароль</label>
                    <div class="col-md-6">

```

```

        <input id="password" type="password"
class="form-control" name="password" required>

        </div>
    </div>
    <div class="form-group row">
        <label for="password-confirm" class="col-md-
4 col-form-label text-md-right">Подтвердите
        пароль</label>
        <div class="col-md-6">
            <input id="password-confirm"
type="password" class="form-control"
name="password_confirmation" required>
        </div>
    </div>
    <div class="form-group row mb-0">
        <div class="col-md-6 offset-md-4">
            <button type="submit" class="btn btn-
primary">
                Зарегистрироваться
            </button>
        </div>
    </div>
</form>
</div>
</div>
@endsection

```

login.blade.php

```

@extends('layouts.master')
@section('title', 'Авторизация')
@section('content')
<div class="col-md-8">
    <div class="card">

```

```

<div class="card-header">Авторизация</div>
<div class="card-body">
    <form method="POST" action="{{route('login')}}"
aria-label="Login">
        @csrf
        <div class="form-group row">
            <label for="email" class="col-sm-4 col-form-
label text-md-right">E-Mail</label>
            <div class="col-md-6">
                <input id="email" type="email"
class="form-control" name="email" value="" required autofocus>
            </div>
        </div>
        <div class="form-group row">
            <label for="password" class="col-md-4 col-
form-label text-md-right">Пароль</label>
            <div class="col-md-6">
                <input id="password" type="password"
class="form-control" name="password" required>
            </div>
        </div>
        <div class="form-group row mb-0">
            <div class="col-md-8 offset-md-4">
                <button type="submit" class="btn btn-
primary">
                    Войти
                </button>
            </div>
        </div>
    </form>
</div>
</div>
@endsection

```

auth\master.blade.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>Админка: @yield('title')</title>
    <!-- Scripts -->
    <script src="/js/app.js" defer></script>
    <!-- Styles -->
    <link href="/css/app.css" rel="stylesheet">
    <link href="/css/bootstrap.min.css" rel="stylesheet">
    <link href="/css/admin.css" rel="stylesheet">
</head>
<body>
    <div id="app">
        <nav class="navbar navbar-default navbar-expand-md
navbar-light navbar-laravel">
            <div class="container">
                <a class="navbar-brand"
href="{{route('index')}}">
                    Вернуться на сайт
                </a>
                <ul class="nav navbar-nav">
                    @admin
                        <li><a
href="{{route('categories.index')}}">Категории</a></li>
                        <li><a
href="{{route('products.index')}}">Товары</a></li>
                        <li><a
href="{{route('basket')}}">Заказы</a></li>
                    @endadmin
                </ul>
            </div>
        </nav>
    </div>
</body>
</html>
```

```

</ul>
<div id="navbar" class="collapse navbar-
collapse">
    @guest
        <ul class="nav navbar-nav navbar-right">
            <li class="nav-item">
                <a class="nav-link" href="{{
route('login') }}">Войти</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{
route('register') }}">Зарегистрироваться</a>
            </li>
        </ul>
    @endguest
    @auth
        <ul class="nav navbar-nav navbar-right">
            <li class="nav-item dropdown">
                <a id="navbarDropdown"
class="nav-link dropdown-toggle" href="#" role="button"
                data-toggle="dropdown"
                aria-haspopup="true" aria-
expanded="false" v-pre>
                    @admin
                        Администратор
                    @else
                        Пользователь
                    @endadmin
                </a>
                <div class="dropdown-menu
dropdown-menu-right" aria-labelledby="navbarDropdown">
                    <a class="dropdown-item"
href="{{ route('logout') }}"

```

```

onclick="event.preventDefault();

document.getElementById('logout-form').submit();">
        Выйти
    </a>
    <form id="logout-form"
action="{{ route('logout')}}" method="POST"
        style="display: none;">
        @csrf
    </form>
</div>
</li>
</ul>
@endauth
</div>
</div>
</nav>
<div class="py-4">
    <div class="container">
        <div class="row justify-content-center">
            @yield('content')
        </div>
    </div>
</div>
</div>
</body>
</html>

```

auth\product\index.blade.php

```

@extends('auth.layouts.master')
@section('title', 'Товары')
@section('content')
    <div class="col-md-12">
        <h1>Товары</h1>
        <table class="table">

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 89 |

```

<tbody>
<tr>
<th>
#
</th>
<th>
Код
</th>
<th>
Название
</th>
<th>
Категория
</th>
<th>
Кол-во товарных предложений
</th>
<th>
Действия
</th>
</tr>
@foreach($products as $product)
<tr>
<td>{{ $product->id }}

```

```

                                href="{{
route('products.show', $product) }}">Открыть</a>
                                <a class="btn btn-warning"
type="button"
                                href="{{
route('products.edit', $product) }}">Редактировать</a>
                                @csrf
                                @method('DELETE')
                                <input class="btn btn-danger"
type="submit" value="Удалить"></form>
                                </div>
                                </td>
                                </tr>
                                @endforeach
                                </tbody>
                                </table>
                                {{ $products->links() }}
                                <a class="btn btn-success" type="button" href="{{
route('products.create') }}">Добавить товар</a>
                                </div>
                                @endsection

```

auth\product\show.blade.php

```

@extends('auth.layouts.master')
@section('title', 'Продукт ' . $product->name)
@section('content')
    <div class="col-md-12">
        <h1>{{ $product->name }}</h1>
        <table class="table">
            <tbody>
                <tr>
                    <th>
                        Поле
                    </th>
                    <th>

```

```

                Значение
            </th>
        </tr>
        <tr>
            <td>ID</td>
            <td>{{ $product->id }}</td>
        </tr>
        <tr>
            <td>Код</td>
            <td>{{ $product->code }}</td>
        </tr>
        <tr>
            <td>Название</td>
            <td>{{ $product->name }}</td>
        </tr>
        <tr>
            <td>Описание</td>
            <td>{{ $product->description }}</td>
        </tr>
        <tr>
            <td>Картинка</td>
            <td></td>
        </tr>
        <tr>
            <td>Категория</td>
            <td>{{ $product->category->name }}</td>
        </tr>
    </tbody>
</table>
</div>
@endsection

auth\product\form.blade.php

@extends('auth.layouts.master')

```

```

@isset($product)
    @section('title', 'Редактировать товар ' . $product->name)
@else
    @section('title', 'Создать товар')
@endisset

@section('content')
    <div class="col-md-12">
        @isset($product)
            <h1>Редактировать товар <b>{{ $product->name
        }}</b></h1>
        @else
            <h1>Добавить товар</h1>
        @endisset
        <form method="POST" enctype="multipart/form-data"
            @isset($product)
                action="{{ route('products.update', $product) }}"
            @else
                action="{{ route('products.store') }}"
            @endisset
        >

        <div>
            @isset($product)
                @method('PUT')
            @endisset
            @csrf
            <div class="input-group row">
                <label for="code" class="col-sm-2 col-form-
label">Код: </label>
                <div class="col-sm-6">
                    @error('code')
                        <div class="alert alert-danger">{{
$message }}</div>

```

```

        @enderror
        <input type="text" class="form-control"
name="code" id="code"
        value="@isset($product){{$product->code }}@else{{ old('code') }}@endisset">
    </div>
</div>
<br>
<div class="input-group row">
    <label for="name" class="col-sm-2 col-form-label">Название: </label>
    <div class="col-sm-6">
        @error('name')
            <div class="alert alert-danger">{{
$message }}</div>
        @enderror
        <input type="text" class="form-control"
name="name" id="name"
        value="@isset($product){{$product->name }}@else{{ old('code') }}@endisset">
    </div>
</div>
<br>
<div class="input-group row">
    <label for="category_id" class="col-sm-2 col-form-label">Категория: </label>
    <div class="col-sm-6">
        <select name="category_id"
id="category_id" class="form-control">
            @foreach($categories as $category)
                <option value="{{ $category->id
}} "
            @isset($product)
            @if($product->

```

```

>category_id == $category->id)

                                selected
                                @endif
                                @endisset
                                >{{ $category->name }}</option>
                                @endforeach
                                </select>
                                </div>
                                </div>
                                <br>
                                <div class="input-group row">
                                    <label for="price" class="col-sm-2 col-form-
label">Цена: </label>
                                    <div class="col-sm-6">
                                        @error('price')
                                            <div class="alert alert-danger">{{
$message }}</div>
                                        @enderror
                                        <input type="text" class="form-control"
name="price" id="price"
                                            value="@isset($product){{
$product->price }}@endisset{{ old('price') }}">
                                        </div>
                                    </div>
                                    <br>
                                    <div class="input-group row">
                                        <label for="description" class="col-sm-2
col-form-label">Описание: </label>
                                        <div class="col-sm-6">
                                            @error('description')
                                                <div class="alert alert-danger">{{
$message }}</div>
                                            @enderror
                                            <textarea name="description"

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 95 |

```

id="description" cols="72"
                                rows="7">@isset($product){{
$product->description }}@endisset{{ old('description')
}}</textarea>
                                </div>
                                </div>
                                <br>
                                <div class="input-group row">
                                    <label for="image" class="col-sm-2 col-form-
label">Картинка: </label>
                                    <div class="col-sm-10">
                                        <label class="btn btn-default btn-file">
                                            Загрузить <input type="file"
style="display: none;" name="image" id="image">
                                        </label>
                                    </div>
                                </div>
                                <br>
                                <button class="btn btn-
success">Сохранить</button>
                                </div>
                                </form>
                                </div>
@endsection

```

auth\orders\index.blade.php

```

@extends('auth.layouts.master')
@section('title', 'Заказы')
@section('content')
    <div class="col-md-12">
        <h1>Заказы</h1>
        <table class="table">
            <tbody>
                <tr>
                    <th>

```

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 96 |

```

        #
    </th>
    <th>
        Имя
    </th>
    <th>
        Телефон
    </th>
    <th>
        Когда отправлен
    </th>
    <th>
        Сумма
    </th>
    <th>
        Действия
    </th>
</tr>
@foreach($orders as $order)
    <tr>
        <td>{{ $order->id }}</td>
        <td>{{ $order->name }}</td>
        <td>{{ $order->phone }}</td>
        <td>{{ $order->created_at->format('H:i
d/m/Y') }}</td>
        <td>{{ $order->getFullPrice() }} руб.</td>
        <td>
            <div class="btn-group" role="group">
                <a class="btn btn-success"
type="button"
                @admin
                    href="{{ route('orders.show',
$order) }}"
                @else

```

```

                                href="{{
route('person.orders.show', $order) }}"
                                @endadmin >
                                Открыть</a>
                            </div>
                        </td>
                    </tr>
                @endforeach
            </tbody>
        </table>
    </div>
@endsection

```

auth/orders/show.blade.php

```

@extends('auth.layouts.master')
@section('title', 'Заказ ' . $order->id)
@section('content')
    <div class="py-4">
        <div class="container">
            <div class="justify-content-center">
                <div class="panel">
                    <h1>Заказ №{{ $order->id }}</h1>
                    <p>Заказчик: <b>{{ $order->name }}</b></p>
                    <p>Номер телефона: <b>{{ $order->phomne
}}</b></p>

                    <table class="table table-striped">
                        <thead>
                            <tr>
                                <th>Название</th>
                                <th>Кол-во</th>
                                <th>Цена</th>
                                <th>Стоимость</th>
                            </tr>
                        </thead>
                        <tbody>

```

```

        @foreach ($order->products as $product)
            <tr>
                <td>
                    <a href="{{ route('product',
$product) }}">

                            {{ $product->name }}
                    </a>
                </td>
                <td><span
class="badge">1</span></td>
                <td>{{ $product->price }}
руб.</td>
                <td>{{ $product->
getPriceForCount() }} руб.</td>
            </tr>
        @endforeach
        <tr>
            <td colspan="3">Общая
стоимость:</td>
            <td>{{ $order->getFullPrice() }}
руб.</td>
        </tr>
    </tbody>
</table>
<br>
</div>
</div>
</div>
</div>
@endsection

```

auth\categories\index.blade.php

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 99 |

```

@extends('auth.layouts.master')
@section('title', 'Создание категории')
@section('content')
    <div class="col-md-12">
        <h1>Категории</h1>
        <table class="table">
            <tbody>
                <tr>
                    <th>
                        #
                    </th>
                    <th>
                        Код
                    </th>
                    <th>
                        Название
                    </th>
                    <th>
                        Действия
                    </th>
                </tr>
                @foreach($categories as $category)
                    <tr>
                        <td>{{ $category->id }}</td>
                        <td>{{ $category->code }}</td>
                        <td>{{ $category->name }}</td>
                        <td>
                            <div class="btn-group" role="group">
                                <form action="{{
route('categories.destroy', $category) }}" method="POST">
                                    <a class="btn btn-success"
type="button"
                                href="{{
route('categories.show', $category) }}">Открыть</a>

```

```

<a class="btn btn-warning"
type="button"
href="{{
route('categories.edit', $category) }}">Редактировать</a>

@csrf
@method('DELETE')
<input class="btn btn-
danger" type="submit" value="Удалить">
</form>
</div>
</td>
</tr>
@endforeach
</tbody>
</table>
<a class="btn btn-success" type="button" href="{{
route('categories.create') }}">Добавить категорию</a>
</div>
@endsection

```

auth\categories\show.blade.php

```

@extends('auth.layouts.master')
@section('title', 'Категория ' . $category->name)
@section('content')
<div class="col-md-12">
<h1>Категория: <b>{{ $category->name }}</b></h1>
<table class="table">
<tbody>
<tr>
<th>
Поле
</th>
<th>
Значение

```

```

        </th>
    </tr>
    <tr>
        <td>ID</td>
        <td>{{ $category->id }}</td>
    </tr>
    <tr>
        <td>Код</td>
        <td>{{ $category->code }}</td>
    </tr>
    <tr>
        <td>Название</td>
        <td>{{ $category->name }}</td>
    </tr>
    <tr>
        <td>Описание</td>
        <td>{{ $category->description }}</td>
    </tr>
    <tr>
        <td>Картинка</td>
        <td></td>
    </tr>
    <tr>
        <td>Кол-во товаров</td>
        <td>{{ $category->products->count() }}</td>
    </tr>
</tbody>
</table>
</div>
@endsection

auth\categories\form.blade.php

@extends('auth.layouts.master')

```

```

@isset($category)
    @section('title', 'Редактировать категорию ' . $category->name)
@else
    @section('title', 'Создать категорию')
@endisset
@section('content')
    <div class="col-md-12">
        @isset($category)
            <h1>Редактировать категорию: <b>{{ $category->name
        }}</b></h1>
        @else
            <h1>Добавить категорию</h1>
        @endisset
        <form method="POST" enctype="multipart/form-data"
            @isset($category)
                action="{{ route('categories.update', $category)
            }}"
            @else
                action="{{ route('categories.store') }}"
            @endisset
        >

        <div>
            @isset($category)
                @method('PUT')
            @endisset
            @csrf
            <div class="input-group row">
                <label for="code" class="col-sm-2 col-form-label">Код: </label>
                <div class="col-sm-6">
                    @error('code')
                        <div class="alert alert-danger">{{
                            $message }}</div>
                    @enderror
                </div>
            </div>
        </div>
    </div>

```

```

        <input type="text" class="form-control"
name="code" id="code"
        value="@isset($category){{$category->code }}@endisset{{ old('code') }}">
    </div>
</div>
<br>
<div class="input-group row">
    <label for="name" class="col-sm-2 col-form-label">Название: </label>
    <div class="col-sm-6">
        @error('name')
            <div class="alert alert-danger">{{
$message }}</div>
        @enderror
        <input type="text" class="form-control"
name="name" id="name"
        value="@isset($category){{$category->name }}@endisset{{ old('name') }}">
    </div>
</div>
<br>
<div class="input-group row">
    <label for="description" class="col-sm-2 col-form-label">Описание категории: </label>
    <div class="col-sm-6">
        @error('description')
            <div class="alert alert-danger">{{
$message }}</div>
        @enderror
        <textarea name="description"
id="description" cols="72" rows="7">
            @isset($category){{$category->description }}@endisset{{
old('description') }}

```

```

        </textarea>
    </div>
</div>
<br>
<div class="input-group row">
    <label for="image" class="col-sm-2 col-form-
label">Картинка: </label>
    <div class="col-sm-10">
        <label class="btn btn-default btn-file">
            Загрузить <input type="file"
style="display: none;" name="image" id="image">
        </label>
    </div>
</div>
<button class="btn btn-
success">Сохранить</button>
    </div>
</form>
</div>
@endsection

```

Приложение Д

(обязательное)

Сетевое приложение «бизнес-потребители» для электронного магазина

Презентация

| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 106 |

Выпускная квалификационная работа

На тему: «Сетевое приложение «Бизнес-потребители»
для электронного магазина»

Автор работы: студент группы 18BB2 Ткачев А.М.

Руководитель: к.т.н., доцент Карамышева Н.С.

Введение

На сегодняшний день развитие Интернета вышло на высокий уровень и все больше пользователей появляется в сети. Одним из больших плюсов Интернета, является электронная коммерция или другими словами – онлайн торговля.

Целью создания сетевого приложения является предоставление каталога продуктов японской литературы и возможности её покупки.

Практическая значимость данной работы заключается в получении завершённого и автономного сетевого продукта, готового к использованию.



| | | | | | | |
|------|------|----------|---------|------|-----------------------|------|
| | | | | | ПГУ1 090301.40.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 107 |

Цель работы

Цель работы – разработать сетевое приложение, позволяющее пользователям покупать товары.

Разработанный электронный магазин предназначен для предоставления каталога продуктов японской литературы под категорией «Манга» и возможности её покупки.

Анализ аналогов

- AliExpress, WildBerries, OZON - интернет-магазины, занимающиеся продажей одежды, обуви, электроники, детских товаров, товаров для дома и множества других;
- «Читай город» и «Буквоед» – это магазины, специализирующиеся на продаже литературы;
- «Fast Anime» - интернет-магазин по продаже аниме атрибутики

- 1) Множество категорий
- 2) Фильтрация товара
- 3) Дизайн
- 4) Простота в использовании

Стек технологий

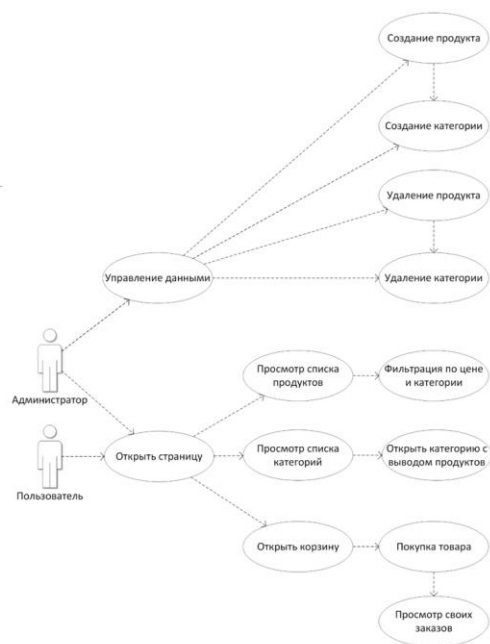
- Язык программирования PHP
- Framework Laravel
- HTML, CSS и JavaScript
- СУБД MySQL
- Visual Studio Code



Функциональные блоки архитектуры

- Класс-маршрутизатор «web.php», в котором заключены все маршруты сайта
- Классы-посредники в директории определяют права доступа у пользователя,
- Классы-контроллеры - это класс является, в которых задана основная логика программы
- Мастер-класс «master.blade.php» содержит в себе часть кода, которая может повторяться в других классах
- Директория \resources, содержит все языковые файлы, в которых задается разметка страницы;

Диаграмма прецедентов



Физическая модель базы данных

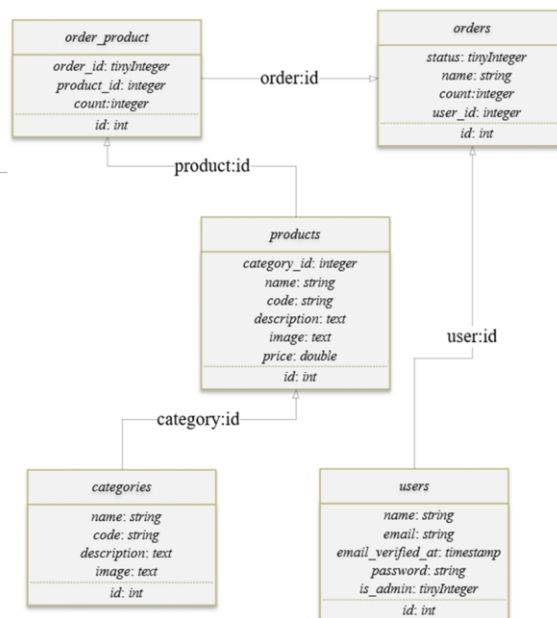
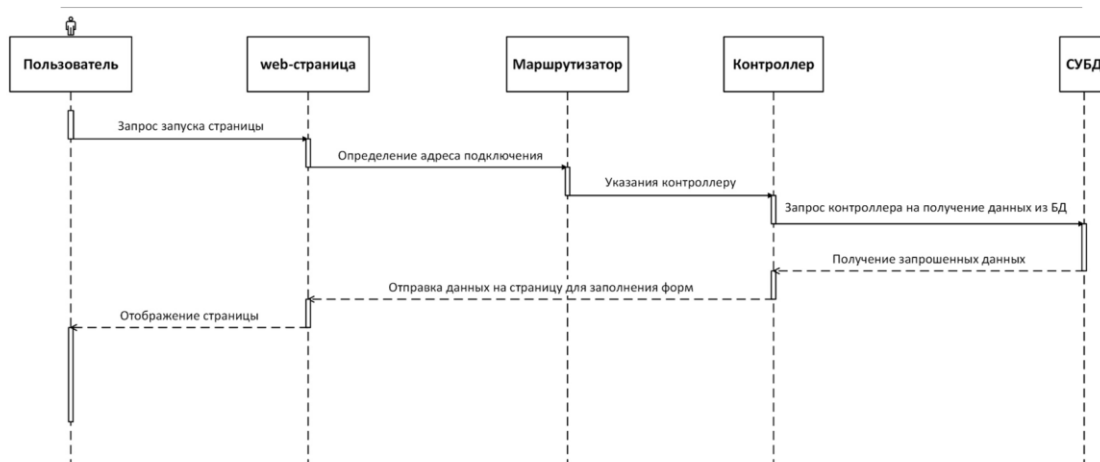


Диаграмма последовательностей



Возможности администратора

Категории

| # | Код | Название | Действия |
|----|---------|----------|--|
| 1 | senen | Сёнен | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 2 | mekha | Меха | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 3 | fentezi | Фэнтези | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 10 | drama | Драма | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |

Добавить категорию

Товары

| # | Код | Название | Категория | Кол-во товарных предложений | Действия |
|----|--------------------------------|----------------|-----------|-----------------------------|--|
| 6 | Kidou_Senshi_Gundam_The_Origin | Воин Gundam | Меха | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 8 | Berserk | Берсерк | Фэнтези | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 9 | Mushishi | Мастер мист | Фэнтези | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 18 | One_Piece | Ван-Пас | Сёнен | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 19 | Ashita_no_Joe | Завтрашний Джо | Сёнен | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 20 | Guren_Lagann | Гуррен-Лагган | Меха | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 21 | Sidonia_no_Kishi | Рыцари Сирионы | Меха | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |
| 22 | Gintama | Гинтама | Сёнен | | <button>Открыть</button> <button>Редактировать</button> <button>Удалить</button> |

Добавить товар

Сайт для клиента

Интернет-Магазин

Все товары

Категории

В корзину

Мои заказы

Выйти

КАТАЛОГ

ЦЕНА

от до

КАТЕГОРИИ

☐ Сёнен

☐ Меха

☐ Фэнтези

☐ Драма

Фильтры

Сброс



Воин Гандам

700 руб.

Добавить в корзину



Берсерк

1100 руб.

Добавить в корзину



Мастер муси

580 руб.

Добавить в корзину



Один Piece

Добавить в корзину



Драгон Болл

Добавить в корзину



Гинтам

Добавить в корзину

Заключение

В ходе выполнения данной работы была изучена методология проектирования сетевого приложения «бизнес-потребители» для электронного магазина и приобретены навыки работы с фрейворком Laravel. При реализации поставленной задачи были получены следующие результаты:

- разработана структура базы данных для СУБД MySQL;
- разработан фронтенд электронного магазина;
- разработан и отлажен бэкенд для электронного магазина

Сетевое приложение обладает адаптивной версткой и кроссбраузерностью, для запуска на любых устройствах.