

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего образования  
«Московский политехнический университет»

Кафедра «Инфокогнитивные технологии»  
Образовательная программа «Веб-технологии»

Отчет по курсовому проекту  
по дисциплине «Инженерное проектирование»

Тема: «Административная панель магазина продажи электроники»

**Выполнил:**

Студент группы 191-321

Мажаев В.С.

---

подпись, дата

**Принял:**

Старший преподаватель

Даньшина М.В.

---

подпись, дата

Москва 2020

## СОДЕРЖАНИЕ

### Оглавление

СОДЕРЖАНИЕ .....	2
ВВЕДЕНИЕ .....	3
Тип разработки .....	3
Задача .....	3
Аналоги .....	3
ОСНОВНАЯ ЧАСТЬ .....	7
Структура базы данных .....	7
Use cases .....	9
Этапы реализации проекта .....	9
ЗАКЛЮЧЕНИЕ .....	18
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	18

## ВВЕДЕНИЕ

### Тип разработки

Индивидуальный проект, направленный на отработку навыков Full-stack разработки. Разработка ведется поэтапно.

### Задача

Спроектировать, создать базу данных для магазина продажи электроники и настроить административную панель. Административная панель поможет управлять сайтом, регулярно его обновлять, редактировать и создавать новые товары, акции и все что нужно магазину.

### Аналоги

Начнем с М.Видео.

В магазине есть каталог, состоящий из категорий (смартфоны, телевизоры, техника для кухни и т.д.)

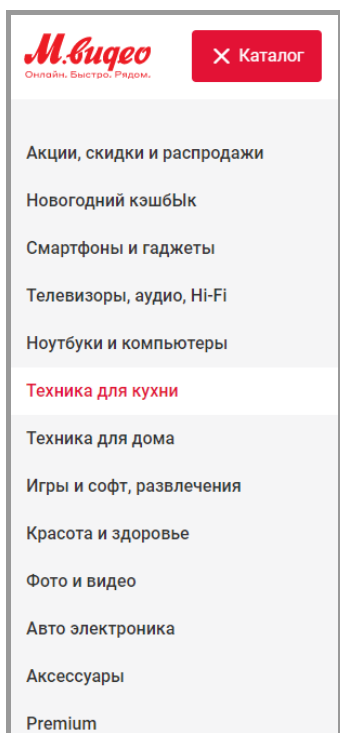


Рисунок 1

В категории находятся товары, которые обладают набором информации, а именно фотографии товара, краткая информация о товаре, его характеристики, отзывы и наличие в магазинах.

На картинке ниже раздел характеристик у смартфона.


## Характеристики

Экран .....	6.67"/2400x1080 Пикс
Оперативная память (RAM) ? .....	4 ГБ
Встроенная память (ROM) .....	128 ГБ
Основная камера МПикс .....	48/8/2/2
Фронтальная камера МПикс .....	8
Технология NFC ? .....	Да
Поддержка стандартов .....	2G/3G/4G LTE
SIM карта .....	2 nano-SIM
Поддержка быстрой зарядки .....	Да
Сканер отпечатка пальца под экраном ? .....	Да

Рисунок 2

Причем количество характеристик у каждой категории товаров разное.

Отзыв состоит из имени автора, даты, количества баллов (звездочек), и текста отзыва

**Марат** 05.12.2020  **Рекомендует**

**5** Мощность ★★★★★ Функциональность ★★★★★ Время работы ★★★★★

Симпатичный и удобный в руке аппарат, размер подходящий сделали, молодцы. Экран без рамки, яркий, картинка радует глаз. Есть NFS-ка, батарея долго держится, плюс зарядка быстрая.

Рисунок 3

Наличие в магазинах показывается таблицей, в которой есть адрес магазина, станция метро, наличие товара и режим работы магазина




Адрес	Станция метро	Наличие	Забрать	Режим работы
<b>24</b> м. «Алексеевская» Москва, пр-т Мира, д. 91, к. 1	Алексеевская		через 15 минут	Магазин открыт 24 часа
<b>24</b> м. «Славянский бульвар» Москва, Славянский б-р, д. 13, стр. 1	Славянский Бульвар		через 15 минут	Магазин открыт 24 часа
<b>24</b> 24 часа! м. «Улица 1905 года» Москва, ул. Красная Пресня, д. 23, к. Б, стр. 1	Улица 1905 года		через 15 минут	Магазин открыт 24 часа

Рисунок 4

Также в этом магазине есть личный кабинет, он содержит информацию о заказах, персональных скидках и личные данные пользователя, такие как телефон, email и фео.

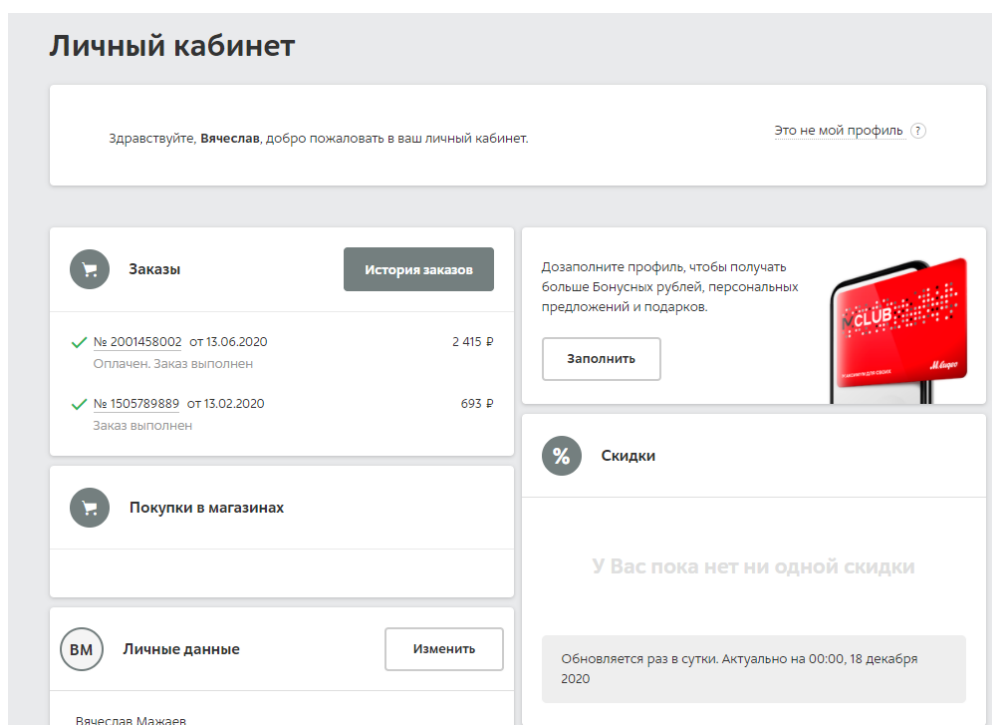


Рисунок 5

Также к пользователю прикреплены раздел “избранное” и корзина:

Теперь рассмотрим Ситилинк, он в общем отличается только дизайном, логические разделы те же самые.

У него тоже есть раздел с категориями

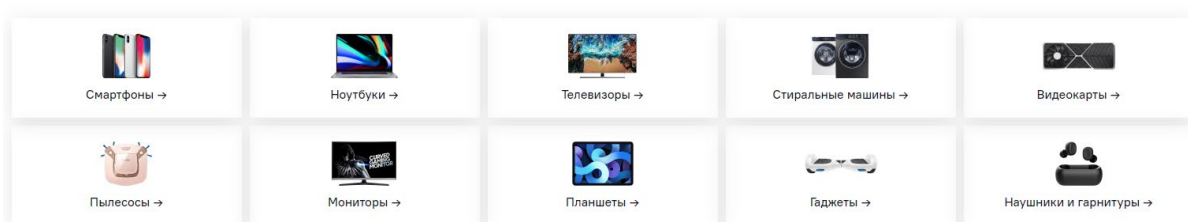


Рисунок 6

Характеристиками устройства

### Дисплей

Дисплей	6.1", IPS
Разрешение дисплея	1792x828
Число пикселей на дюйм	326PPI
Сенсорный экран ②	Multitouch
Защитное покрытие экрана	стекло
Смартфон с «Монобровью»	ДА

### Конфигурация

Процессор ②	Apple A13 Bionic,
Объем встроенной памяти ②	64 ГБ

### Камера

Двойная камера	есть
Основная камера ②	12Мп
Диафрагма, основная камера	F/2.4
Вторая основная камера	12 МП
Вспышка ②	есть

Рисунок 7

## Отзывы к товару


**Валера** ★ 5
 24 сентября 2019

**Достоинства**

Iphone 11 получился отличным. Перешел на него с бки и конечно разница колоссальная. Пересмотрел множество обзоров перед покупкой и был готов как к минусам так и к плюсам. Опять же минусы сугубо субъективные, все зависит с какой модели телефона вы пересаживаетесь. Что касается рамок — при сравнении на них делался постоянный упор обзорщиками, что они широкие и бросаются в глаза.. скажу так — для меня это совершенно не заметно, сам экран шикарный, ни о какой зернистой и речи быть не может, все прекрасно. Ночной режим съемки просто фантастический, в кромешной темноте получаются шикарные снимки ночного города. Быстродействие отличное.

**Недостатки**

пожалую один нашелся недостаток — маркая задняя панель, отпечатки оставляет. Решается покупкой чехла. но даже без него, при попадании телефона в карман джинс все «протирается».

**Комментарий**

прекрасный Iphone, своих денег безусловно стоит.

Рисунок 8

И личный кабинет имеет информацию о заказах, персональных скидках и личные данные пользователя, такие как телефон, email и ФИО. Разделы избранное и корзина также присутствуют.

## ОСНОВНАЯ ЧАСТЬ

### Структура базы данных

Следуя из анализа конкурентов, база данных и админ панель должны включать в себя сущности: товара, который имеет характеристики, относится к определенной категории товаров, имеется в наличии в магазинах в определенном количестве, имеет отзыв. Пользователь может сделать заказ, добавить товар в избранное, сделать отзыв, который включает в себя рейтинг. Рейтинг — это количество звездочек(баллов) за качество и функционал товара.

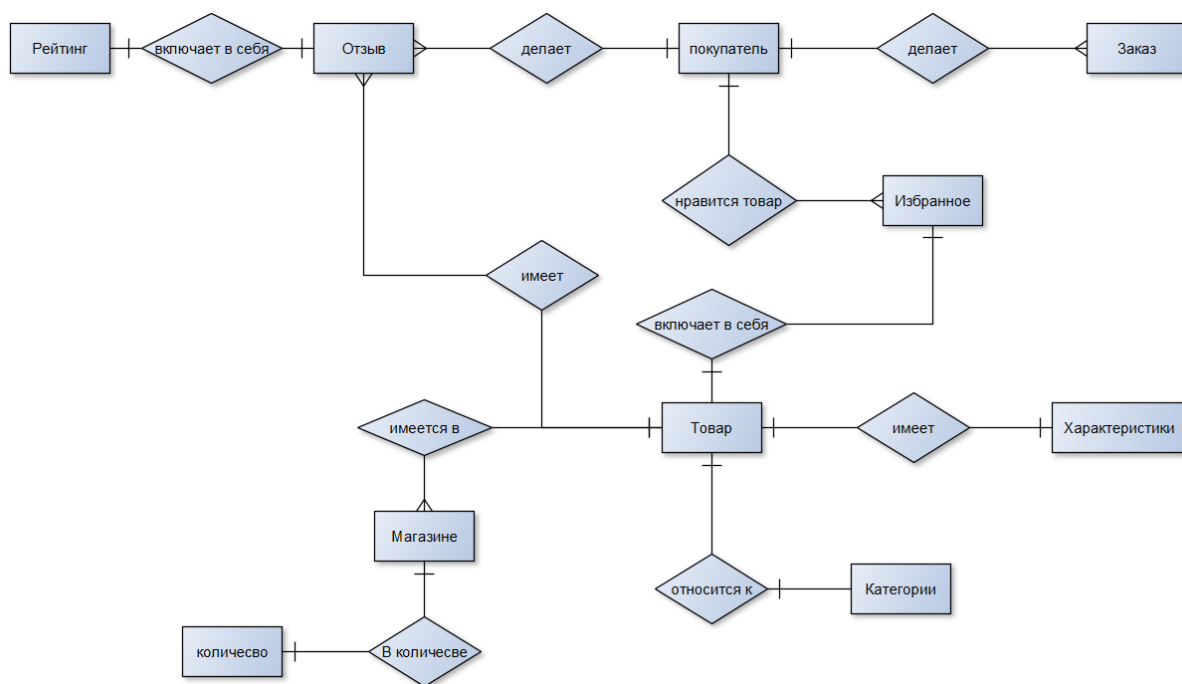


Рисунок 9 инфологическая схема БД

С помощью инфологической схемы можно составить подробную физическую схему БД. В ней находятся все те же сущности, но каждая из них имеет набор полей с определенным типом данных. Таблица `products` имеет связь по внешнему ключу с таблицами `categories` (связь один к одному) и `specifications` (связь один к одному). Таблица `reviews` имеет связь по внешнему ключу с `products` (связь один к одному), `ratings` (связь один к одному) и `users` (связь один ко многим). Таблицы `favourite_products` и `orders` связаны с `users` и с `products` по внешнему ключу и имеют связь один ко многим. Таблица `quantity` показывает количество товара в определенном магазине и связана внешним ключом с `stores` (один ко многим) и `products` (один к одному).



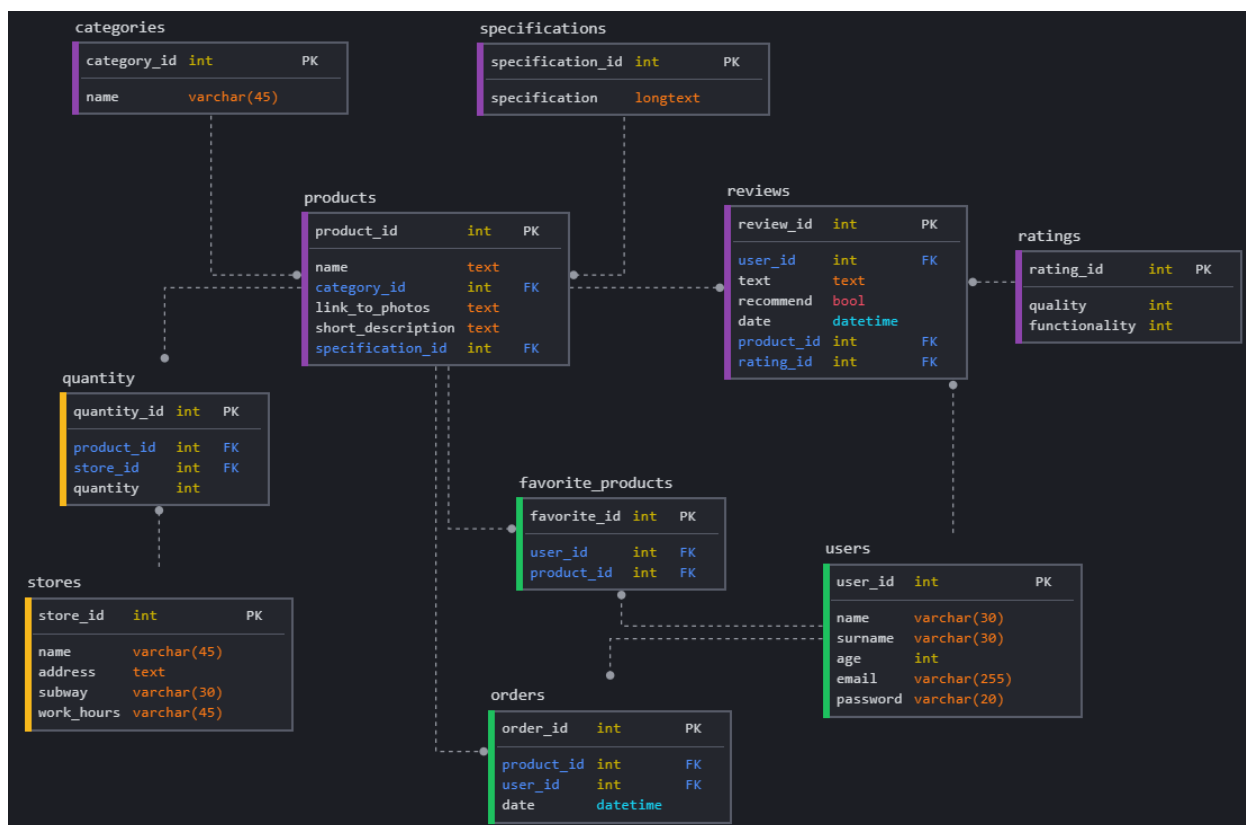


Рисунок 10 физическая схема БД

## Use cases

### Кейсы использования приложения

1. Администратор добавляет новую категорию товаров
2. Администратор добавляет новый товар магазина
3. Администратор добавляет описание и характеристики к новому товару
4. Администратор добавляет новый магазин
5. Администратор импортирует данные новых товаров
6. Администратор экспортирует список отзывов от пользователей
7. Покупатель добавляет товар в избранное
8. Покупатель добавляет товар в корзину
9. Покупатель пишет отзыв к товару
10. Администратор удаляет пользователя
11. Администратор удаляет товар
12. Администратор удаляет магазин
13. Менеджер в магазине смотрит наличие товара на складе
14. Менеджер в магазине смотрит характеристики товара, чтобы рассказать покупателю
15. Разработчик проверяет работу frontend части интернет-магазина
16. Отдел логистики составляет план завоза товаров
17. Руководство компании составления отчет

## Этапы реализации проекта

1. Анализ конкурентов

Для успешной реализации проекта необходимо разбираться в области продажи электронной техники, в этом мне помог анализ конкурентов. Я проанализировал несколько сайтов занимающихся продажей электроники, а именно их структуру и функционал, чтобы понять какую информацию нужно хранить в базе данных.

## 2. Проектирование инфологической модели предметной области

Основываясь на накопленной информации из анализа конкурентов, я составил схему «Сущность связь», которая показывает существующие сущности в проекте и их связь.

## 3. Проектирование физической структуры

На основе инфологической модели я создал физическую модель базы данных, в которой видно не только сущности и их характеристики, но и тип данных.

## 4. Создание репозитория проекта

На GitHub я создал репозиторий проекта и клонировал его на компьютер с помощью WebStorm, далее по мере добавления функционала я создавал ветки, в которых изучал и тестировал разные фишки и вел разработку нового функционала, который мог все поломать. Только после того, как приложение могло стабильно работать я сливал ветку в мастер.

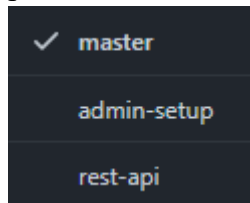


Рисунок 11 Пример веток

## 5. Создание Django-приложения

Для начала я изучил материалы по Django в интернете, просмотрел видео курсы и статьи, создав пару тестовых приложений. Создание основного приложения я начал с логического разделения всего проекта на приложения (products, stores, users). Они также отмечены разными цветами на физической модели базы данных. Далее я начал перенос базы данных в файлы models.py находящиеся в своих приложениях.

```
class Product(models.Model):
    name = models.CharField('name', max_length=200)
    category = models.ForeignKey(Category, on_delete=models.PROTECT)
    specification = models.ForeignKey(Specification, on_delete=models.PROTECT)
    link_to_photos = models.URLField('link', max_length=200)
    short_description = models.TextField('Short description')

    def __str__(self):
        return self.name
```

Рисунок 12 Пример сущности products

## 6. Настройка административного интерфейса Django

После первоначальной настройки я начал добавлять вывод сущностей в админ панель, добавил функции фильтрации и поиска

```
@admin.register(Product)
class ProductAdmin(ImportExportActionModelAdmin):
    list_display = ('name', 'category', 'link_to_photos', 'short_description')
    list_filter = ('category',)
    search_fields = ('name', 'category__name')
```

Рисунок 13 Пример регистрации сущности

Чтобы зайти в админ панель я создал два пользователя: admin/admin и user/user. Пользователя admin я создал командой «python manage.py createsuperuser», а user создал из админ панели. Для модели user доступен только просмотр таблиц. Вот как выглядит страница сущности products в админ панели, поиск производится по двум полям, name и category\_name причем category\_name это внешний ключ, и чтобы поиск работал с ним нужно указание таблицы, в которой находится информация, именно поэтому поле category из сущности product я вписал в search\_fields как category\_name. Фильтрация производится по категории товара.

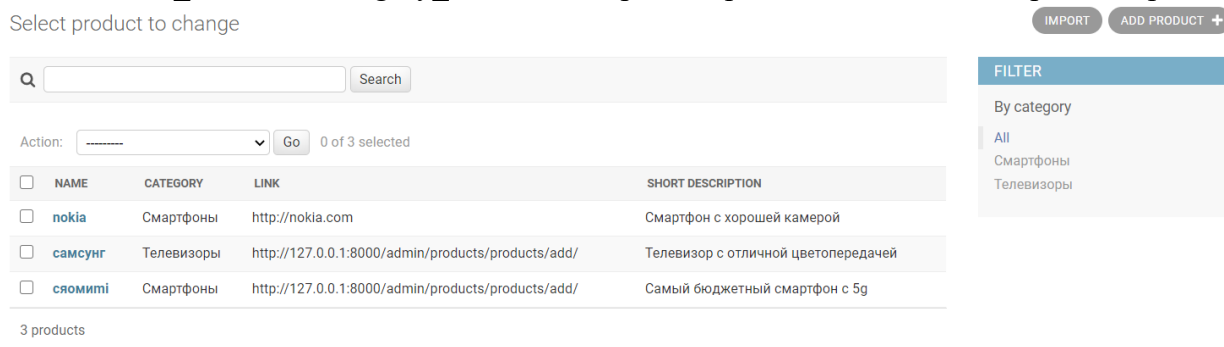


Рисунок 14

Также есть функция сортировки по возрастанию и убыванию.



Рисунок 15

Также я настроил admin actions. Я добавил функцию пополнения товаров на складе. Админ выбирает товары из списка и выполняет функцию, после этого количество выбранных товаров в магазинах становится равным 1000.

```
def apply_full_stock(modeladmin, request, queryset):
    for quantity in queryset:
        quantity.quantity = 1000
        quantity.save()

    apply_full_stock.short_description = 'Apply full stock'
```

Рисунок 16

Action: Apply full stock Go 2 of 2 selected

<input checked="" type="checkbox"/>	PRODUCT	STORE	QUANTITY
<input checked="" type="checkbox"/>	nokia	ТЦ домодедовский	12
<input checked="" type="checkbox"/>	самсунг	ТЦ домодедовский	1

2 quantitys

Рисунок 17

<input type="checkbox"/>	PRODUCT	STORE	QUANTITY
<input type="checkbox"/>	nokia	ТЦ домодедовский	1000
<input type="checkbox"/>	самсунг	ТЦ домодедовский	1000

Рисунок 18

## 7. Наполнение базы данных

Далее я заполнил базу данных тестовой информацией

## 8. Реализация REST API

Я добавил возможность управления данными приложения со стороны, реализовал REST API, а также разобрался с тестированием API через сервис POSTMAN.

Вот 3 примера URL созданные для API

```
app_name = 'products'
urlpatterns = [
    path('product/create/', ProductCreateView.as_view()),
    path('all/', ProductsListView.as_view()),
    path('product/detail/<int:pk>', ProductDetailView.as_view()),
]
```

Рисунок 19

Этот URL отвечает за создание нового товара

127.0.0.1:8000/api/v1/products/product/create/

Django REST framework

Product Create

Product Create

POST /api/v1/products/product/create/

HTTP 201 Created  
Allow: POST, OPTIONS  
Content-Type: application/json  
Vary: Accept

```
{
  "id": 6,
  "name": "nokia",
  "link_to_photos": "http://nokia.com",
  "short_description": "Смартфон с хорошей камерой",
  "category": 1,
  "specification": 1
}
```

Рисунок 20

Этот URL отвечает за вывод всех товаров

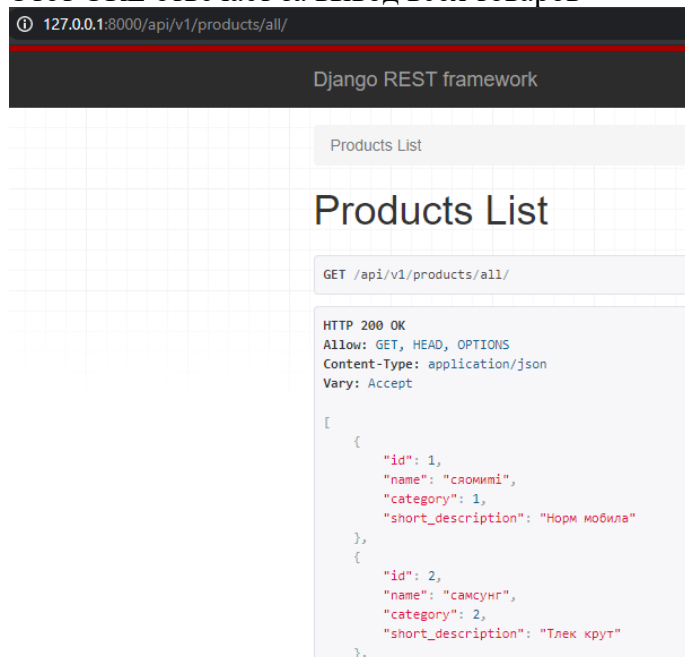


Рисунок 21

А этот за вывод товара по его id

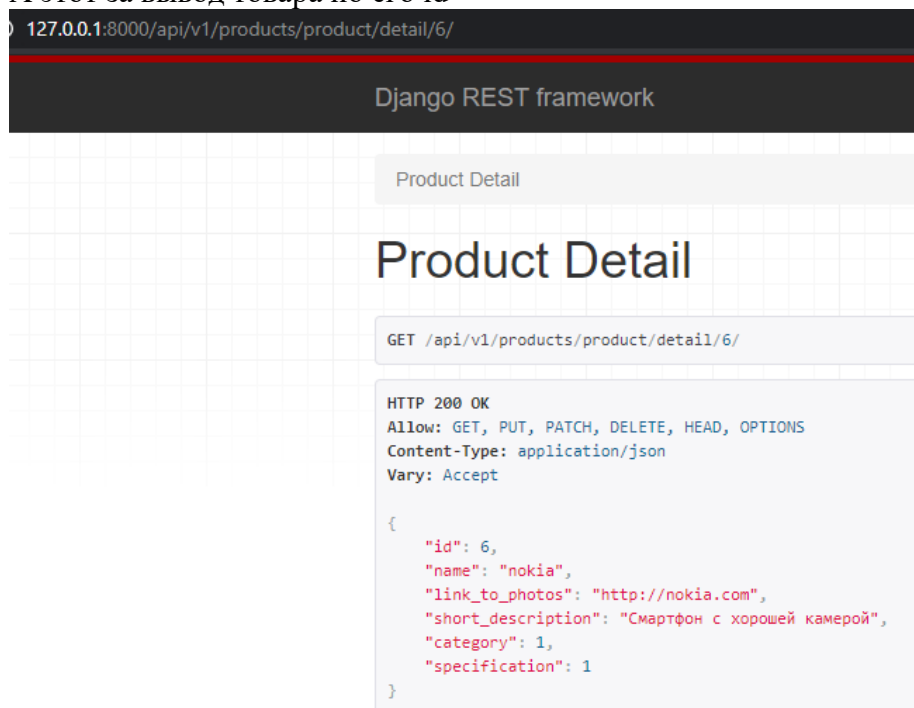


Рисунок 22

Далее тестирование этих 3 URL в сервисе Postman (Рис 23-25)

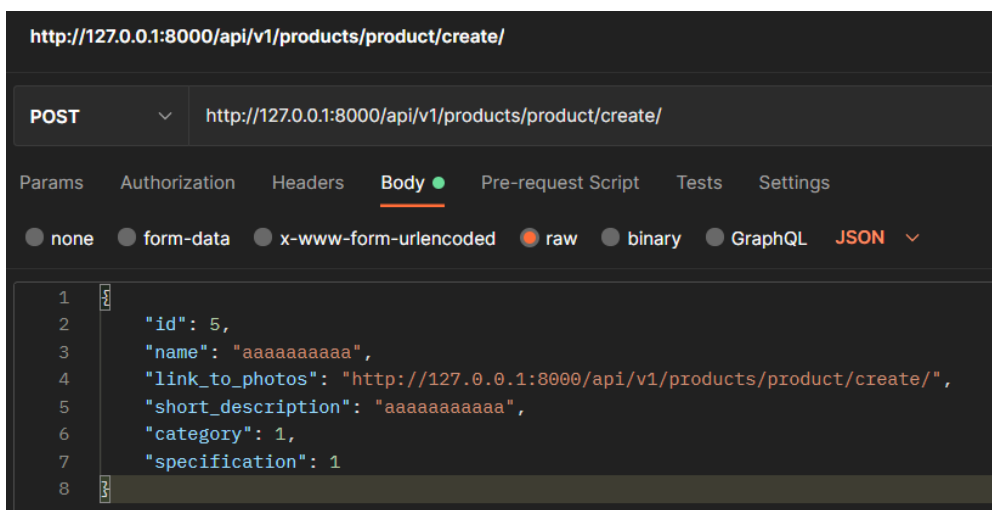


Рисунок 23

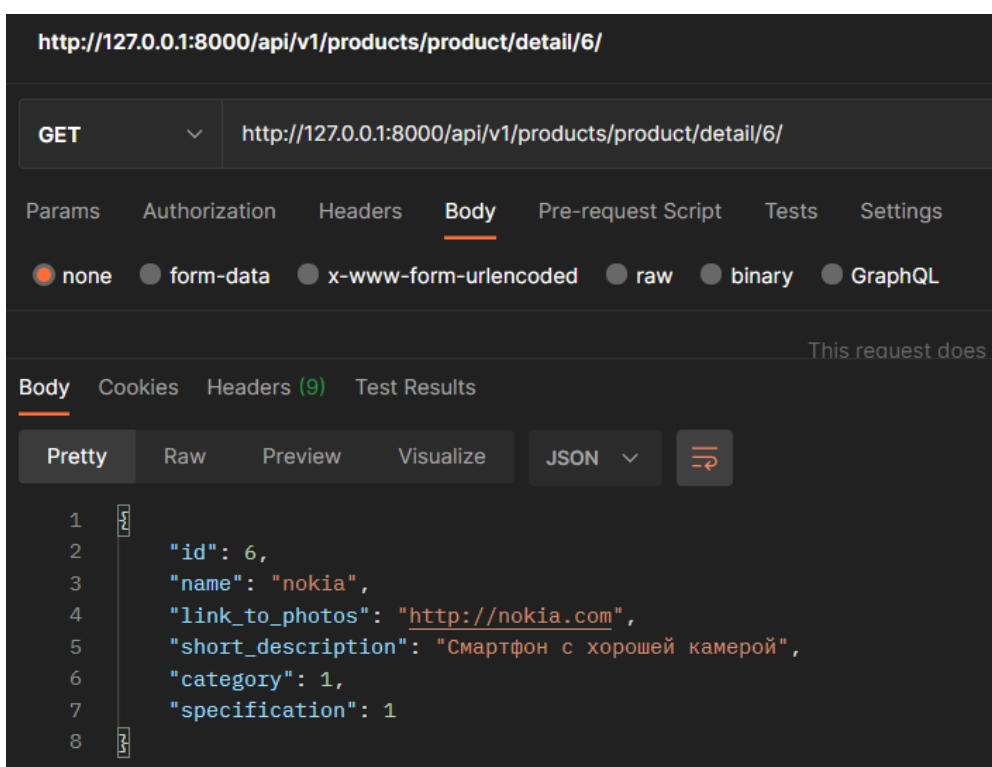


Рисунок 24

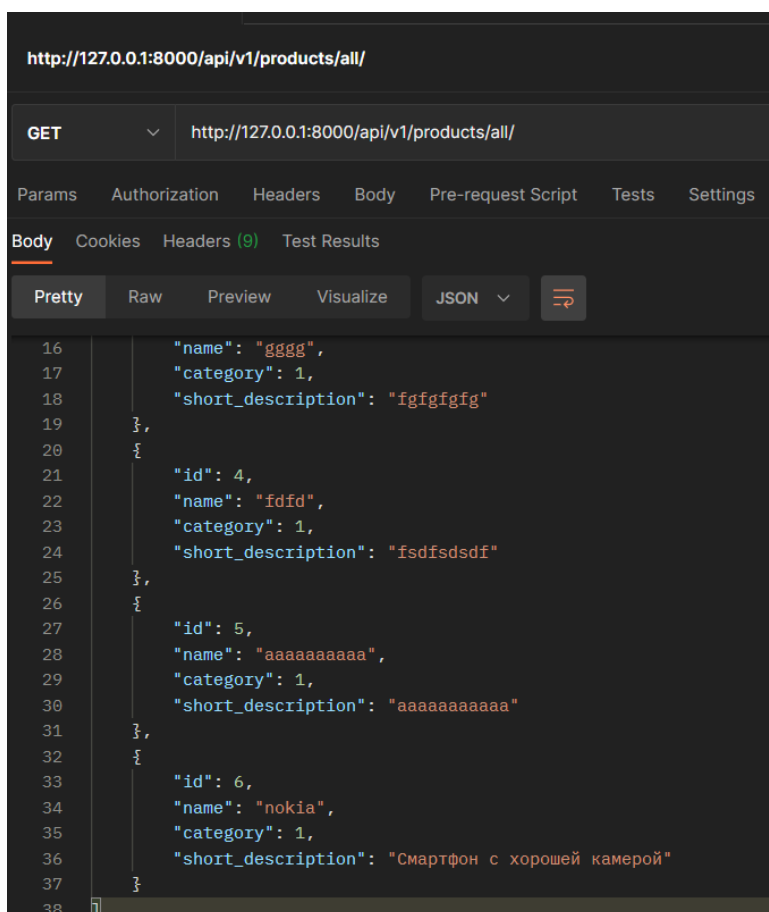


Рисунок 25

## 9. Реализация экспорта и импорта данных

С помощью `django-import-export` были реализованы возможности импорта и экспорта данных.

Экспорт находится в admin actions

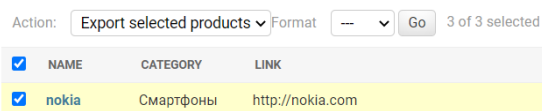


Рисунок 26

## Импорт

### Import

This importer will import the following fields: `id`, `name`, `category`, `specification`, `link_to_photos`, `short_description`

File to import:  Файл не выбран

Format:

Рисунок 27

Для создания этих функций при регистрации объектов админки мы наследуемся от класса `ImportExportActionModelAdmin`

```
@admin.register(Product)
class ProductAdmin(ImportExportActionModelAdmin):
    list_display = ('name', 'category', 'link_to_photos', 'short_description')
    list_filter = ('category',)
    search_fields = ('name', 'category__name')
```

Рисунок 28

## 10. Написание типовых запросов к базе данных

### Добавление объекта рейтинг

```
>>> from products.models import Rating, Review
>>> r = Rating(quality=5, functionality=5)
>>> r.save()
```

Select rating to change

Action:   0 of 3 selected

<input type="checkbox"/>	QUALITY	FUNCTIONALITY
<input type="checkbox"/>	5	5
<input type="checkbox"/>	5	4
<input type="checkbox"/>	4	4

3 ratings

### Изменение поля quality

```
>>> r.quality = 1
>>> r.save()
```

Select rating to change

Action:   0 of 3 selected

<input type="checkbox"/>	QUALITY	FUNCTIONALITY
<input type="checkbox"/>	1	5
<input type="checkbox"/>	5	4
<input type="checkbox"/>	4	4

3 ratings

Создание ссылки на объект рейтинга с id 3, ссылки на объект продукта с id 6  
И создание объекта отзыва, содержащего внешние ключи на вышеописанные объекты.

```
>>> ra = Rating.objects.get(id=3)
>>> from products.models import Product
>>> p = Product.objects.get(id=6)
>>> re = Review(text='отличный аппарат', recommendation=True, product=p, rating=ra)
>>> re.save()
```



Select review to change

IMPORT

Action:  Go 0 of 3 selected

<input type="checkbox"/>	REVIEW TEXT	RECOMMENDATION	DATE	PRODUCT	RATING
<input type="checkbox"/>	отличный аппарат	✓	Jan. 12, 2021, 9:08 a.m.	nokia	Rating object (3)
<input type="checkbox"/>	cool tv	✓	Jan. 8, 2021, 8:12 p.m.	самсунг	Rating object (1)
<input type="checkbox"/>	Работает плохо	✗	Jan. 8, 2021, 7:47 p.m.	сяоми1	Rating object (1)

### Вывод всех объектов продукт

```
>>> all_products=Product.objects.all()
>>> print(all_products)
<QuerySet [<Product: сяоми1>, <Product: самсунг>, <Product: nokia>]>
>>>
```

### Вывод продуктов начинающихся на с

```
>>> products=Product.objects.filter(name__startswith='c')
>>> print(products)
<QuerySet [<Product: сяоми1>, <Product: самсунг>]>
>>>
```

### Вывод продуктов начинающихся на с и принадлежащих категории 1

```
>>> products=Product.objects.filter(name__startswith='c').filter(category=1)
>>> print(products)
<QuerySet [<Product: сяоми1>]>
>>>
```

### Вывод всех объектов отзывов

```
>>> reviews=Review.objects.all()
>>> print(reviews)
<QuerySet [<Review: Работает плохо>, <Review: cool tv>, <Review: отличный аппарат>]>
>>>
```

### Фильтр отзывов по дню

```
>>> reviews=Review.objects.filter(date__day=8)
>>> print(reviews)
<QuerySet [<Review: Работает плохо>, <Review: cool tv>]>
>>>
```

<input type="checkbox"/>	REVIEW TEXT	RECOMMENDATION	DATE
<input type="checkbox"/>	отличный аппарат	✓	Jan. 12, 2021, 9:08 a.m.
<input type="checkbox"/>	cool tv	✓	Jan. 8, 2021, 8:12 p.m.
<input type="checkbox"/>	Работает плохо	✗	Jan. 8, 2021, 7:47 p.m.

### Обновление объекта отфильтрованного по дню

```
>>> reviews=Review.objects.filter(date__day=12)
>>> print(reviews)
<QuerySet [<Review: отличный аппарат>]>
>>> reviews.update(text='Супер отличный аппарат')
1
>>> print(reviews)
<QuerySet [<Review: Супер отличный аппарат>]>
>>> []
```

И его удаление

```
>>> reviews.delete()
(1, {'products.Review': 1})
>>> print(reviews)
<QuerySet []>
>>> []
```

11. Документирование
12. Заполнение оценочного листа

## ЗАКЛЮЧЕНИЕ

Результатом всех работ является готовая административная панель django-приложения, два типа пользователей и мешок знаний.

GitHub <https://github.com/slavocado/mvideo>

Приложение <http://mvideo.std-947.ist.mospolytech.ru/admin>

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

<https://docs.djangoproject.com/en/3.1/>

<https://webdevblog.ru/kak-prevratit-adminku-django-v-legkovesnuju-panel-instrumentov>

<https://djbook.ru/>

<https://django-import-export.readthedocs.io/en/latest/>

<https://www.django-rest-framework.org/>

<https://django.fun/tutorials/put-ot-request-do-response-v-django/>

<https://pythonist.ru/kastomizacziya-admin-paneli-django/>