**СТиВПП 3 - Проект каталог. Начало**

Чек-лист требований заказчика

* Красиво – Использован Bootstrap для стилизации, адаптивная вёрстка.
* Поиск – Реализован поиск по бренду, модели и артикулу.
* Фильтрация – Фильтрация по интерфейсу (SATA/NVMe) и диапазону цен.
* CRUD для товаров – Добавление, редактирование, удаление через админ-панель Django.
* Три страницы – Главная (список товаров), детальное описание, страница "О сервисе".

В рамках проекта была разработана пользовательская часть, представляющая собой удобный интерфейс для просмотра каталога SSD. Главная страница содержит список товаров, представленных в виде карточек, которые можно быстро искать и фильтровать по различным параметрам. Пользователи могут осуществлять поиск по бренду, модели и артикулу (SKU). Также реализованы фильтры по интерфейсу подключения (SATA или NVMe) и по цене (с возможностью задания минимального и максимального значения).

При переходе на детальную страницу товара пользователь получает полное описание, включая бренд, модель, характеристики и цену.

На бэкенде зарегистрирована модель SSD в административной панели Django. В списке товаров отображаются ключевые поля: бренд, модель, артикул и цена. Для удобства навигации реализованы фильтры по интерфейсу, форм-фактору и наличию товара. Также добавлена возможность поиска по бренду, модели и артикулу.

В форме редактирования информация о товаре удобно сгруппирована по категориям. Доступ к админ-панели ограничен — только пользователи с правами администратора (is\_staff) могут вносить изменения. Для администрирования создан суперпользователь.

В проекте настроены маршруты: главная страница доступна по пути /, страница с детальной информацией о товаре — по /ssd/<id>/, страница «О сервисе» — по /about/, а административная панель — по /admin/.

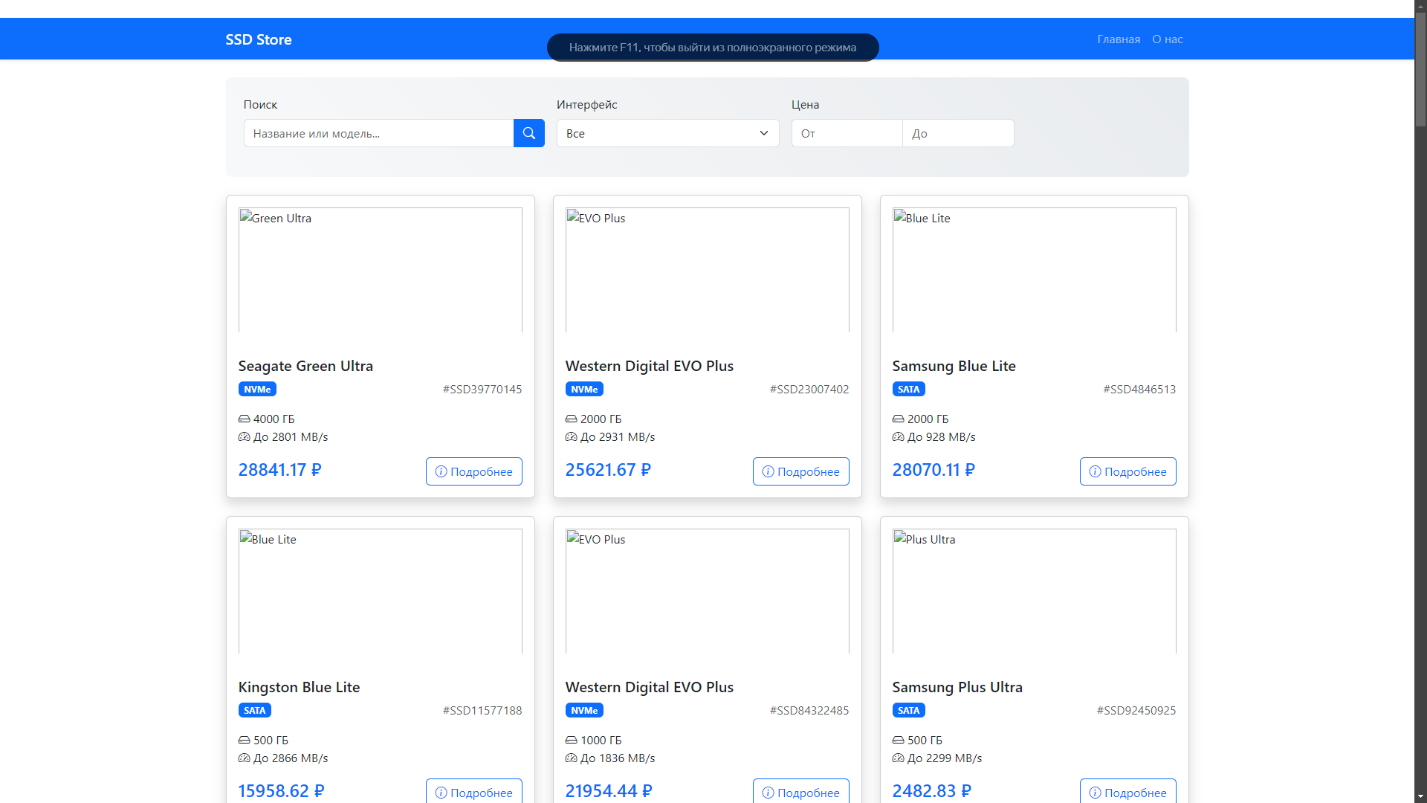
Проект использует базу данных SQLite, настроенную по умолчанию в Django. Все необходимые миграции были выполнены с помощью команд python manage.py makemigrations и python manage.py migrate.

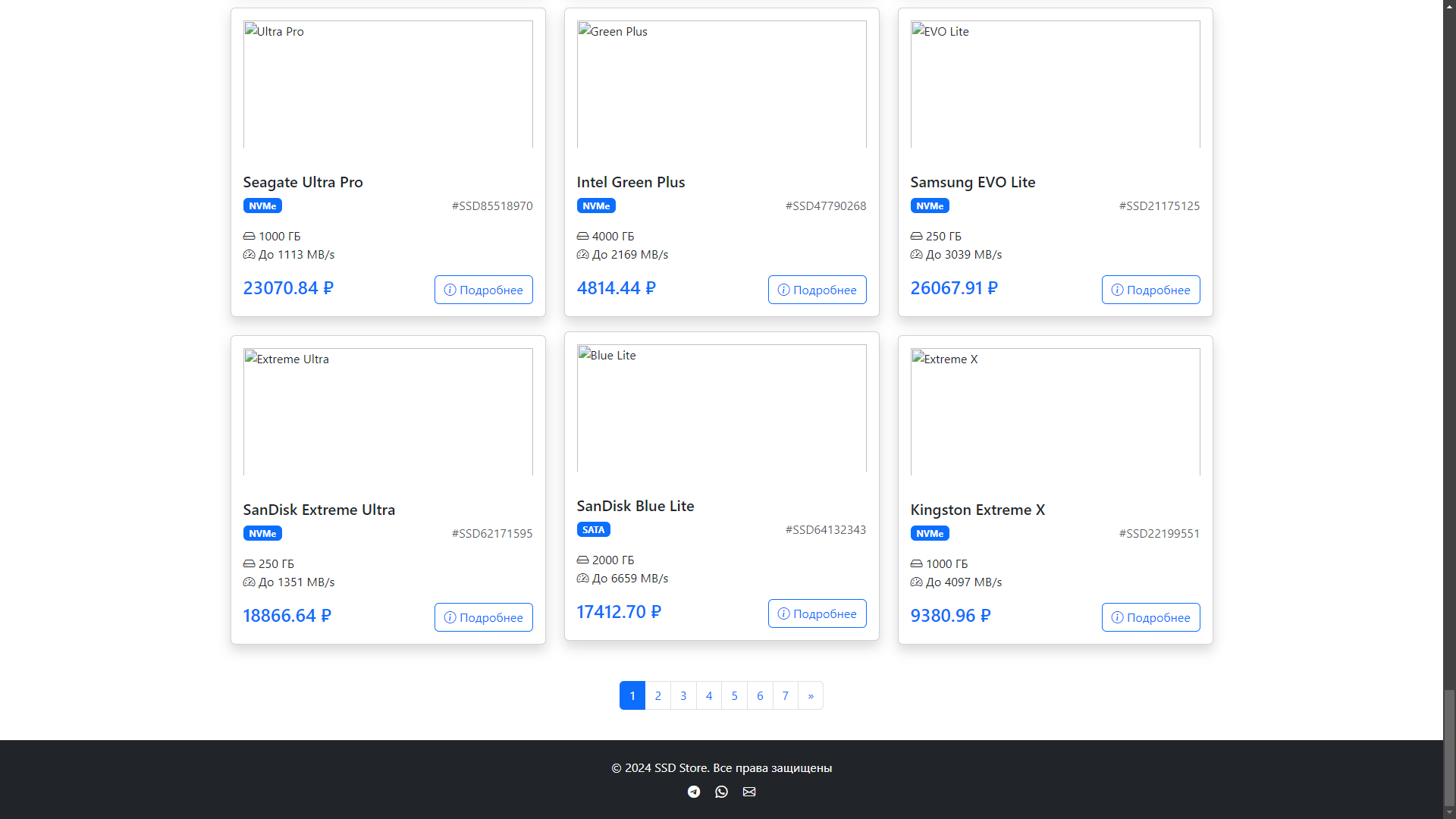
Таким образом, пользовательская часть предоставляет удобный просмотр каталога, быстрый поиск и гибкую фильтрацию. Пользователи могут только просматривать товары без возможности изменения данных. Администраторы через админ-панель получают полный контроль над товарами и могут легко управлять контентом.

Архитектура проекта позволяет без проблем добавлять новые функции, такие как реализация корзины или авторизация пользователей.

**Приложение**

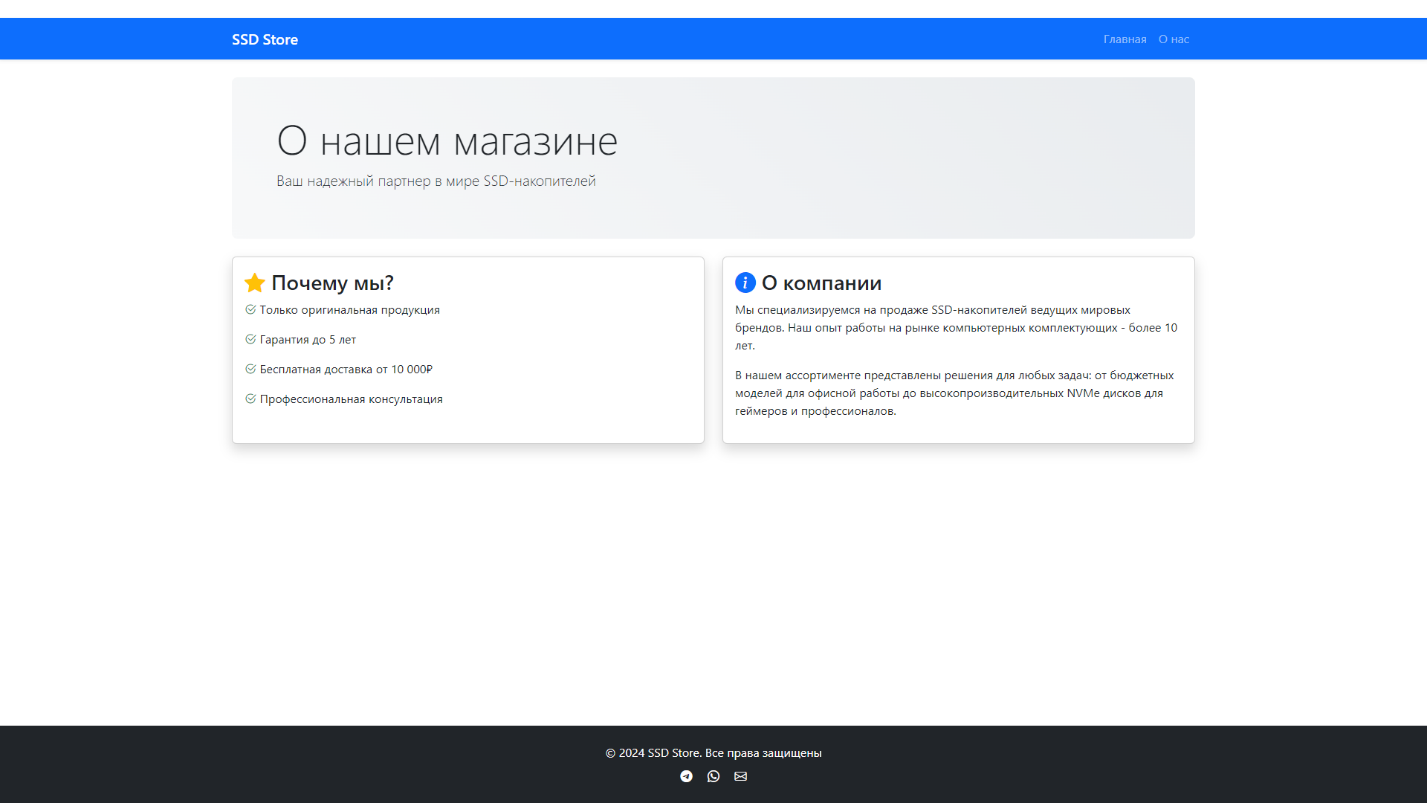
**Приложение А. Скриншоты**

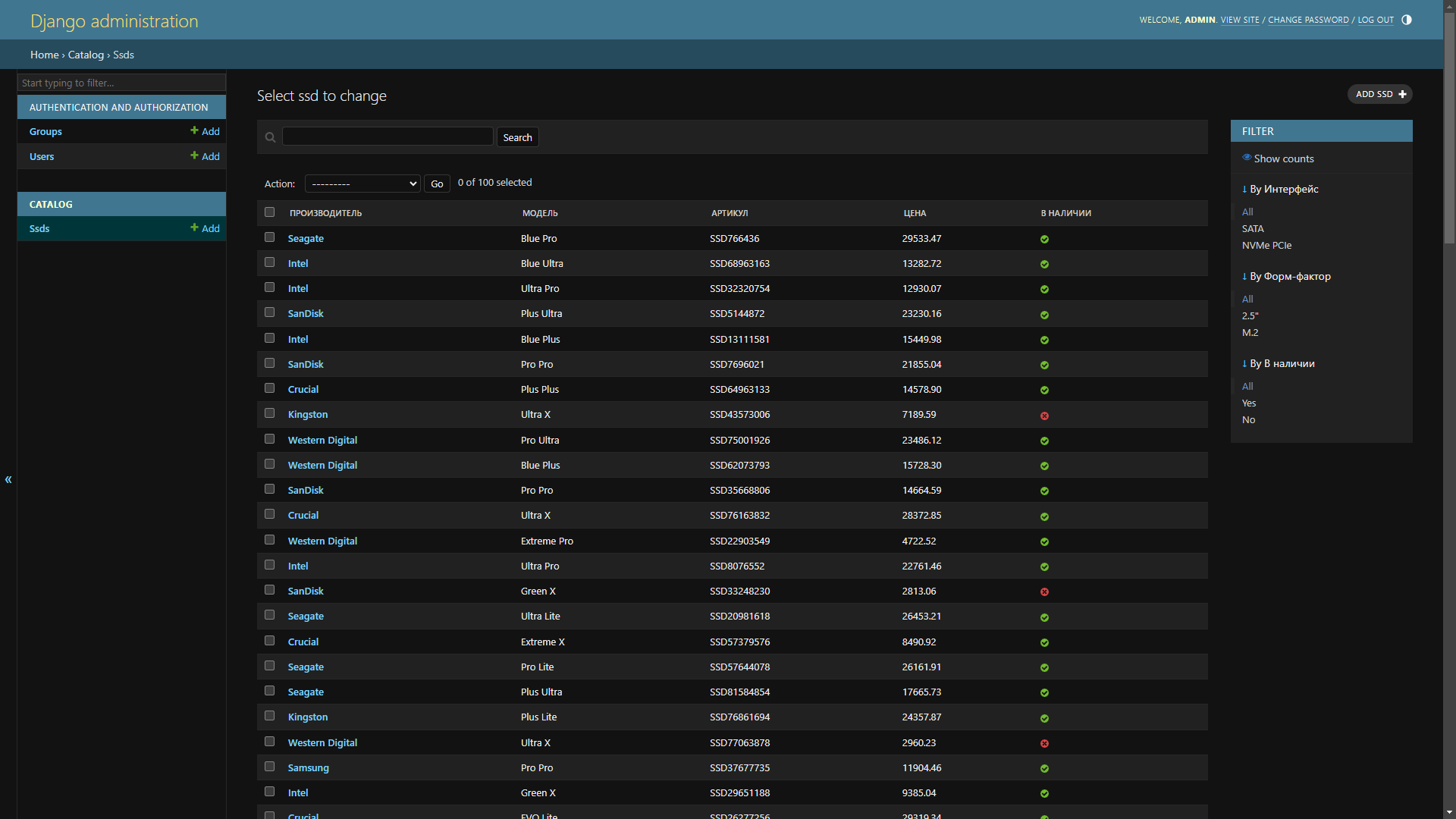
****

****

**Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Контент, сгенерированный ИИ, может содержать ошибки.**

****

****

**Приложение Б. Листинг**

|  |
| --- |
| from django.db.models import Q from django.views.generic import ListView, TemplateView, DetailView  from .models import SSD   class HomeView(ListView):  model = SSD  template\_name = "catalog/home.html"  context\_object\_name = "ssd\_list"  paginate\_by = 48   def get\_queryset(self):  queryset = super().get\_queryset()  self.params = self.request.GET.copy()   search\_query = self.params.get('search')  interface = self.params.get('interface')  min\_price = self.params.get('min\_price')  max\_price = self.params.get('max\_price')   if search\_query:  queryset = queryset.filter(  Q(brand\_\_icontains=search\_query) |  Q(model\_\_icontains=search\_query) |  Q(sku\_\_icontains=search\_query)  )   if interface:  queryset = queryset.filter(interface=interface)   if min\_price:  queryset = queryset.filter(price\_\_gte=min\_price)   if max\_price:  queryset = queryset.filter(price\_\_lte=max\_price)   return queryset   def get\_context\_data(self, \*\*kwargs):  context = super().get\_context\_data(\*\*kwargs)  context['params'] = self.params  return context   class SSDDetailView(DetailView):  model = SSD  template\_name = "catalog/detail.html"   class AboutView(TemplateView):  template\_name = "catalog/about.html" |

Листинг 1. catalog/views

|  |
| --- |
| from django.urls import path  from . import views  urlpatterns = [  path('', views.HomeView.as\_view(), name='home'),  path('ssd/<int:pk>/', views.SSDDetailView.as\_view(), name='ssd\_detail'),  path('about/', views.AboutView.as\_view(), name='about'), ] |

Листинг 2. catalog/urls

|  |
| --- |
| from django.db import models   class SSD(models.Model):  INTERFACE\_TYPES = [  ('SATA', 'SATA'),  ('NVMe', 'NVMe PCIe'),  ]  FORM\_FACTORS = [  ('2.5"', '2.5"'),  ('M.2', 'M.2'),  ]   sku = models.CharField(max\_length=50, unique=True, verbose\_name="Артикул")  brand = models.CharField(max\_length=100, verbose\_name="Производитель")  model = models.CharField(max\_length=100, verbose\_name="Модель")  capacity\_gb = models.PositiveIntegerField(verbose\_name="Ёмкость (ГБ)")  interface = models.CharField(max\_length=20, choices=INTERFACE\_TYPES, verbose\_name="Интерфейс")  form\_factor = models.CharField(max\_length=20, choices=FORM\_FACTORS, verbose\_name="Форм-фактор")  read\_speed = models.PositiveIntegerField(verbose\_name="Скорость чтения (МБ/с)")  write\_speed = models.PositiveIntegerField(verbose\_name="Скорость записи (МБ/с)")  warranty\_years = models.PositiveIntegerField(null=True, blank=True, verbose\_name="Гарантия (лет)")  price = models.DecimalField(max\_digits=10, decimal\_places=2, verbose\_name="Цена")  in\_stock = models.BooleanField(default=True, verbose\_name="В наличии")   def \_\_str\_\_(self):  return f"{self.brand} {self.model}" |

Листинг 3. catalog/models

|  |
| --- |
| from django.apps import AppConfig   class CatalogConfig(AppConfig):  default\_auto\_field = 'django.db.models.BigAutoField'  name = 'catalog' |

Листинг 4. catalog/apps

|  |
| --- |
| from django.contrib import admin  from .models import SSD   @admin.register(SSD) class SSDAdmin(admin.ModelAdmin):  list\_display = ('brand', 'model', 'sku', 'price', 'in\_stock')  list\_filter = ('interface', 'form\_factor', 'in\_stock')  search\_fields = ('brand', 'model', 'sku')  fieldsets = (  ('Основное', {  'fields': ('sku', 'brand', 'model', 'price')  }),  ('Характеристики', {  'fields': ('capacity\_gb', 'interface', 'form\_factor')  }),  ('Дополнительно', {  'fields': ('warranty\_years', 'in\_stock')  }),  ) |

Листинг 5. admin

|  |
| --- |
| from decimal import Decimal  from django.core.management.base import BaseCommand from django\_seed import Seed from faker import Faker  from catalog.models import SSD   class Command(BaseCommand):  help = "Генерирует тестовые данные для SSD"  fake = Faker(['en\_US'])   def add\_arguments(self, parser):  parser.add\_argument(  '--number',  type=int,  default=50,  help='Количество SSD для генерации'  )  parser.add\_argument(  '--clear',  action='store\_true',  help='Очистить базу перед генерацией'  )   def handle(self, \*args, \*\*options):  if options['clear']:  SSD.objects.all().delete()  self.stdout.write(self.style.SUCCESS("База данных очищена"))   seeder = Seed.seeder()  seeder.faker = self.fake   brands = ["Samsung", "Western Digital", "Kingston", "Crucial", "SanDisk", "Intel", "Seagate"]  models = ["EVO", "Pro", "Blue", "Green", "Ultra", "Plus", "Extreme"]  interface\_choices = [choice[0] for choice in SSD.INTERFACE\_TYPES]  form\_factor\_choices = [choice[0] for choice in SSD.FORM\_FACTORS]   try:  seeder.add\_entity(SSD, options['number'], {  'sku': lambda x: f"SSD{seeder.faker.unique.random\_number(digits=8)}", *# Увеличиваем до 8 цифр* 'brand': lambda x: seeder.faker.random\_element(brands),  'model': lambda x: (  f"{seeder.faker.random\_element(models)} "  f"{seeder.faker.random\_element(['Pro', 'Plus', 'Ultra', 'X', 'Lite'])}"  )[:50], *# Обрезаем до 50 символов* 'capacity\_gb': lambda x: seeder.faker.random\_element([250, 500, 1000, 2000, 4000]),  'interface': lambda x: seeder.faker.random\_element(interface\_choices),  'form\_factor': lambda x: seeder.faker.random\_element(form\_factor\_choices),  'read\_speed': lambda x: seeder.faker.random\_int(500, 7000),  'write\_speed': lambda x: seeder.faker.random\_int(400, 6000),  'warranty\_years': lambda x: seeder.faker.random\_int(1, 5),  'price': lambda x: Decimal( *# Исправляем для DecimalField* seeder.faker.random\_int(2000, 30000) +  Decimal(seeder.faker.random\_int(0, 99)) / 100  ),  'in\_stock': lambda x: seeder.faker.boolean(chance\_of\_getting\_true=75),  })   inserted\_pks = seeder.execute()  self.stdout.write(self.style.SUCCESS(  f"Успешно создано {options['number']} SSD-дисков"  ))   except Exception as e:  self.stdout.write(self.style.ERROR(  f"Ошибка при генерации данных: {str(e)}"  )) |

Листинг 6. catalog/management/commands/seed\_catalog

|  |
| --- |
| {% extends 'catalog/base.html' %}  {% block title %}О нас{% endblock %}  {% block content %} <div class="jumbotron bg-light p-5 mb-4 rounded-3">  <div class="container">  <h1 class="display-4">О нашем магазине</h1>  <p class="lead">Ваш надежный партнер в мире SSD-накопителей</p>  </div> </div>  <div class="row g-4">  <div class="col-md-6">  <div class="card shadow h-100">  <div class="card-body">  <h3><i class="bi bi-star-fill text-warning"></i> Почему мы?</h3>  <ul class="list-unstyled">  <li class="mb-3"><i class="bi bi-check2-circle text-success"></i> Только оригинальная продукция</li>  <li class="mb-3"><i class="bi bi-check2-circle text-success"></i> Гарантия до 5 лет</li>  <li class="mb-3"><i class="bi bi-check2-circle text-success"></i> Бесплатная доставка от 10 000₽</li>  <li class="mb-3"><i class="bi bi-check2-circle text-success"></i> Профессиональная консультация</li>  </ul>  </div>  </div>  </div>   <div class="col-md-6">  <div class="card shadow h-100">  <div class="card-body">  <h3><i class="bi bi-info-circle-fill text-primary"></i> О компании</h3>  <p>Мы специализируемся на продаже SSD-накопителей ведущих мировых брендов. Наш опыт работы на рынке компьютерных комплектующих - более 10 лет.</p>  <p>В нашем ассортименте представлены решения для любых задач: от бюджетных моделей для офисной работы до высокопроизводительных NVMe дисков для геймеров и профессионалов.</p>  </div>  </div>  </div> </div> {% endblock %} |

Листинг 7. catalog/templates/catalog/about.html

|  |
| --- |
| <!DOCTYPE html> <html lang="ru"> <head>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width, initial-scale=1">  <title>SSD Store - {% block title %}{% endblock %}</title>  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.1/font/bootstrap-icons.css">  <style>  .navbar-brand { font-weight: 600; }  .card:hover { transform: translateY(-5px); transition: all 0.3s ease; }  .jumbotron { background: linear-gradient(45deg, #f8f9fa, #e9ecef); }  .product-image { height: 200px; object-fit: contain; }  </style> </head> <body class="d-flex flex-column min-vh-100"> *<!-- Навигация -->* <nav class="navbar navbar-expand-lg navbar-dark bg-primary shadow-sm">  <div class="container">  <a class="navbar-brand" href="{% url 'home' %}">SSD Store</a>  <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav">  <span class="navbar-toggler-icon"></span>  </button>  <div class="collapse navbar-collapse" id="navbarNav">  <ul class="navbar-nav ms-auto">  <li class="nav-item"><a class="nav-link" href="{% url 'home' %}">Главная</a></li>  <li class="nav-item"><a class="nav-link" href="{% url 'about' %}">О нас</a></li>  </ul>  </div>  </div> </nav>  *<!-- Основной контент -->* <main class="flex-grow-1 my-4">  <div class="container">  {% block content %}{% endblock %}  </div> </main>  *<!-- Футер -->* <footer class="bg-dark text-white mt-auto py-4">  <div class="container text-center">  <p class="mb-0">&copy; 2024 SSD Store. Все права защищены</p>  <div class="mt-2">  <a href="#" class="text-white me-3"><i class="bi bi-telegram"></i></a>  <a href="#" class="text-white me-3"><i class="bi bi-whatsapp"></i></a>  <a href="#" class="text-white"><i class="bi bi-envelope"></i></a>  </div>  </div> </footer>  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script> </body> </html> |

Листинг 8. catalog/templates/catalog/base.html

|  |
| --- |
| {% extends 'catalog/base.html' %}  {% block title %}{{ ssd.brand }} {{ ssd.model }}{% endblock %}  {% block content %} <div class="row g-5">  *<!-- Изображение -->* <div class="col-md-6">  <div class="card shadow">  <img src="https://via.placeholder.com/600x400" class="card-img-top product-image p-4" alt="{{ ssd.model }}">  </div>  </div>   *<!-- Характеристики -->* <div class="col-md-6">  <div class="card shadow">  <div class="card-body">  <h1 class="card-title mb-4">{{ ssd.brand }} {{ ssd.model }}</h1>  <div class="d-flex justify-content-between align-items-center mb-4">  <h2 class="text-primary">{{ ssd.price }} ₽</h2>  <span class="badge bg-success">В наличии</span>  </div>   <dl class="row">  <dt class="col-sm-4">Артикул:</dt>  <dd class="col-sm-8">{{ ssd.sku }}</dd>   <dt class="col-sm-4">Интерфейс:</dt>  <dd class="col-sm-8"><span class="badge bg-primary">{{ ssd.interface }}</span></dd>   <dt class="col-sm-4">Объем:</dt>  <dd class="col-sm-8">{{ ssd.capacity\_gb }} ГБ</dd>   <dt class="col-sm-4">Скорость чтения:</dt>  <dd class="col-sm-8">{{ ssd.read\_speed }} MB/s</dd>   <dt class="col-sm-4">Скорость записи:</dt>  <dd class="col-sm-8">{{ ssd.write\_speed }} MB/s</dd>  </dl>   <div class="d-grid gap-2 d-md-flex mt-4">  <a href="{% url 'home' %}" class="btn btn-outline-secondary">  <i class="bi bi-arrow-left"></i> Назад  </a>  <button class="btn btn-primary">  <i class="bi bi-cart"></i> В корзину  </button>  </div>  </div>  </div>  </div> </div> {% endblock %} |

Листинг 9. catalog/templates/catalog/detail.html

|  |
| --- |
| {% extends 'catalog/base.html' %} {% load custom\_tags %} {% load static param\_replace %}  {% block title %}Главная{% endblock %} {% block content %} *<!-- Фильтры -->* <div class="jumbotron p-4 mb-4 rounded-3">  <form method="get">  <div class="row g-3 align-items-end">  <div class="col-md-4">  <label class="form-label">Поиск</label>  <div class="input-group">  <input type="text" name="search" class="form-control" placeholder="Название или модель..."  value="{{ request.GET.search }}">  <button type="submit" class="btn btn-primary">  <i class="bi bi-search"></i>  </button>  </div>  </div>  <div class="col-md-3">  <label class="form-label">Интерфейс</label>  <select name="interface" class="form-select">  <option value="">Все</option>  <option value="SATA" {% if request.GET.interface == "SATA" %}selected{% endif %}>SATA</option>  <option value="NVMe" {% if request.GET.interface == "NVMe" %}selected{% endif %}>NVMe</option>  </select>  </div>  <div class="col-md-3">  <label class="form-label">Цена</label>  <div class="input-group">  <input type="number" name="min\_price" class="form-control" placeholder="От"  value="{{ request.GET.min\_price }}">  <input type="number" name="max\_price" class="form-control" placeholder="До"  value="{{ request.GET.max\_price }}">  </div>  </div>  </div>  </form> </div>  *<!-- Список товаров -->* <div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4">  {% for ssd in page\_obj %}  <div class="col">  <div class="card h-100 shadow">  <img src="https://unsplash.com/photos/random" class="card-img-top product-image p-3" alt="{{ ssd.model }}">  <div class="card-body">  <h5 class="card-title">{{ ssd.brand }} {{ ssd.model }}</h5>  <div class="d-flex justify-content-between align-items-center mb-3">  <span class="badge bg-primary">{{ ssd.interface }}</span>  <span class="text-muted">#{{ ssd.sku }}</span>  </div>  <p class="card-text">  <i class="bi bi-hdd"></i> {{ ssd.capacity\_gb }} ГБ<br>  <i class="bi bi-speedometer2"></i> До {{ ssd.read\_speed }} MB/s  </p>  <div class="d-flex justify-content-between align-items-center">  <h4 class="text-primary">{{ ssd.price }} ₽</h4>  <a href="{% url 'ssd\_detail' ssd.pk %}" class="btn btn-outline-primary">  <i class="bi bi-info-circle"></i> Подробнее  </a>  </div>  </div>  </div>  </div>  {% empty %}  <div class="col-12">  <div class="alert alert-info text-center">По вашему запросу товаров не найдено</div>  </div>  {% endfor %} </div>  *<!-- Пагинация -->* {% if page\_obj.paginator.num\_pages > 1 %} <nav class="mt-5" aria-label="Page navigation">  <ul class="pagination justify-content-center">  {% if page\_obj.has\_previous %}  <li class="page-item">  <a class="page-link"  href="?{% url\_replace page=page\_obj.previous\_page\_number %}"  aria-label="Previous">  <span aria-hidden="true">&laquo;</span>  </a>  </li>  {% endif %}   {% for num in page\_obj.paginator.page\_range %}  {% if page\_obj.number == num %}  <li class="page-item active">  <span class="page-link">{{ num }}</span>  </li>  {% else %}  <li class="page-item">  <a class="page-link"  href="?{% url\_replace page=num %}">{{ num }}</a>  </li>  {% endif %}  {% endfor %}   {% if page\_obj.has\_next %}  <li class="page-item">  <a class="page-link"  href="?{% url\_replace page=page\_obj.next\_page\_number %}"  aria-label="Next">  <span aria-hidden="true">&raquo;</span>  </a>  </li>  {% endif %}  </ul> </nav> {% endif %} {% endblock %} |

Листинг 10. catalog/templates/catalog/home.html

|  |
| --- |
| from urllib.parse import urlencode  from django import template  register = template.Library()   @register.simple\_tag(takes\_context=True) def url\_replace(context, \*\*kwargs):  query = context['request'].GET.dict()  query.update(kwargs)  return urlencode(query) |

Листинг 11. catalog/templatetags/custom\_tags.py

|  |
| --- |
| from django import template  register = template.Library()   @register.simple\_tag def param\_replace(request, \*\*kwargs):  params = request.GET.copy()  for key, value in kwargs.items():  params[key] = value  return params.urlencode() |

Листинг 12. catalog/templatetags/param\_replace.py