Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)

Факультет информационных технологий

Кафедра Инфокогнитивные технологии

09.03.01 «Информатика и вычислительная техника»

Образовательная программа (профиль) «Веб-технологии»

**Курсовая работа**

По курсу Основы программирования

Выполнили:  
Колотыгин Даниил  
Реброва Анастасия

Группа: 241-3210

Проверил: Красникова И.Н.

МОСКВА 2024

**Цель работы**

Получить практические навыки самостоятельной реализации программного продукта. Закрепление теоретических знаний по дисциплине «Основы программирования», а также практических навыков по программированию на языке Python.

**Постановка задачи**

Студенты должны распределиться на команды из 4 человек. 1 команда выбирает 1 тему. Внутри команды студенты разбиваются на 2 подкоманды и разрабатывают проект на конкурентной основе.

В данной курсовой будет разрабатываться приложение с веб-интерфейсом с локальной базой данных отслеживаемых товаров с маркетплейса Ozon.

**Теоретическая часть**

Приложение позволяет отслеживать цену на конкретные позиции и автоматически добавлять в корзину товары, при определенном значении % снижении цены.

Основной функционал:

* Вывод карточек с товарами.
* Просмотр изменения цены.
* Удаление карточек.
* Изменение отслеживаемых параметров(% изменения цены).
* Автоматическое добавление товара в корзину.

Технические детали:

* Фреймворк: Django.
* Язык программирования: Python.
* База данных: SQLite.
* Фронтенд: HTML, CSS, JS.

Структура проекта:

* Приложение main.py: отвечает за обработку данных о птицах.
* Панель администрирования Django: позволяет добавлять и редактировать данные.
* Шаблоны: представляют карточки с информацией.
* Статические файлы: стили, html-шаблоны.

**Программа**

Модель данных на языке Python (models.py).

from datetime import date

from django import utils

from django.db import models

class Product(models.Model):

url = models.URLField(unique=True, primary\_key=True)

name = models.CharField(max\_length=255)

price = models.DecimalField(max\_digits=10, decimal\_places=2, default=0.00) # DecimalField

start\_price = models.DecimalField(max\_digits=10, decimal\_places=2, default=0.00)  # DecimalField

    percent = models.IntegerField(default=100)

cur\_percent = models.DecimalField(max\_digits=10, decimal\_places=2, default=0.00)  created = models.DateTimeField(default=utils.timezone.now)

    def \_\_str\_\_(self):

        return self.name

    def to\_dict(self):

        return {

            'url': self.url,

            'price': str(self.price),

            'percent': self.percent,

            'cur\_percent': self.cur\_percent

        }

Представление (views.py).

import json

from django.shortcuts import redirect, render

from django.http import JsonResponse

from django.views.decorators.csrf import csrf\_exempt

from .classes.parser import Parser

from .classes.db import DB

from decimal import Decimal # Для преобразования цены в Decimal

from .models import Product # Импортируем модель

parser = Parser()

db = DB()

@csrf\_exempt

def update(request):

if request.method == "POST":

url = request.POST.get("url")

price = request.POST.get("price")

if url and price:

price = float(price.replace(",", "."))

db.update\_price(url, price)

return JsonResponse({'result': True})

def get\_all(request):

products = db.get\_all()

# Преобразуем каждый объект Product в словарь

products\_list = [product.to\_dict() for product in products]

return JsonResponse({'data': products\_list})

def get(request):

if request.method != "GET": return

url = request.GET.get("url")

if url is None: return

product = db.get(url)

if product is None: return

return JsonResponse({'data': product.to\_dict()})

def delete(request):

if request.method != "GET": return

url = request.GET.get("url")

if url is None: return

db.delete(url)

return JsonResponse({'result': True})

def products(request):

# Сначала получаем все продукты из базы данных

all\_products = db.get\_all()

if request.method == 'POST':

method = request.POST.get("\_method", "").upper()

if method == "POST":

url = request.POST.get('ozon')

if url:

resp = parser.url(url)

if isinstance(resp, Exception):

return JsonResponse({'error': f"Ошибка при парсинге данных: {str(resp)}"})

price, name = resp # Распаковываем ответ парсера

db.create(name, price, url)

return redirect("products")

if method == "UPDATE":

percent = request.POST.get('procent')

url = request.POST.get('url')

if url and percent:

db.update\_percent(url, int(percent.replace("%", "")))

return redirect("products")

elif method == 'DELETE':

url = request.POST.get('url')

if url:

db.delete(url)

return redirect("products")

return render(request, 'products.html', {'all\_products': all\_products})

Шаблон (products.html).

<div class="product">

   <strong class="product\_name" title="{{ product.name }}"><a href="{{ product.url }}" target="\_blank">{{ product.name }}</a></strong>

   <br>

  <strong class="align\_start">Начальная цена:</strong>

       <p>{{ product.start\_price }}₽</p>

  <strong class="align\_start">При последней проверке:</strong>

       <p>{{ product.price }}₽</p>

 <strong>Купить при падении на:</strong>

 <form id="add\_procent\_form" method="POST">

                        {% csrf\_token %}

    <input type="hidden" name="\_method" value="UPDATE">

     <input type="hidden" name="url" value="{{ product.url }}">

      <label for="procent">

    <input id="add\_procent" type="text" name="procent" id="procent" placeholder="%" value="{{product.percent}}%"> </label>

    <button type="submit" id="accept"><svg width="55" height="55" viewBox="0 0 55 55" fill="none" xmlns="http://www.w3.org/2000/svg"><path d="M5 26.5L25 46.5L50 5" stroke="white" stroke-width="10" stroke-linecap="round"/></svg>

</button>

</form>

<strong class="{% if product.cur\_percent > 0 %} procent\_plus {% endif %} {% if product.cur\_percent < 0 %} procent\_minus {% endif %} {% if product.cur\_percent == 0 %} zero\_procent {% endif %}">

                        {{product.cur\_percent}}%

</strong>

<form id="del\_product" method="POST"> {% csrf\_token %}

   <input type="hidden" name="\_method" value="DELETE">

   <input type="hidden" name="url" value="{{ product.url }}">

  <button type="submit" id="delete">Удалить</button>

</form>

</div>

**Блок-схема**

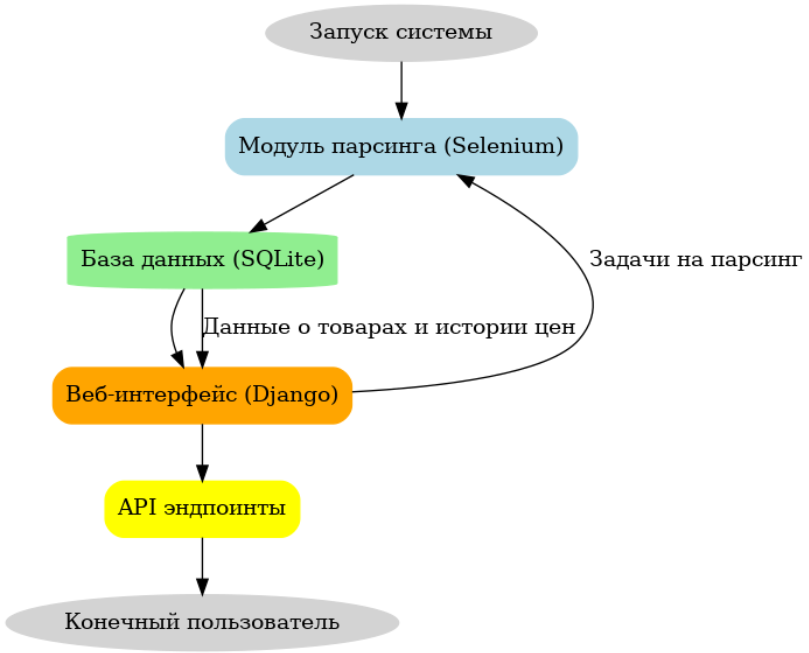
****

Рисунок 1 – Блок-схема работы сайта

**Результат работы**

Результатом работы является сайт, в котором можно посмотреть карточки добавленных и отслеживаемых товаров, удалить их и наблюдать за колебаниями цены. Также можно добавлять новые карточки. Сайт можно посмотреть на Рисунках 2-3.

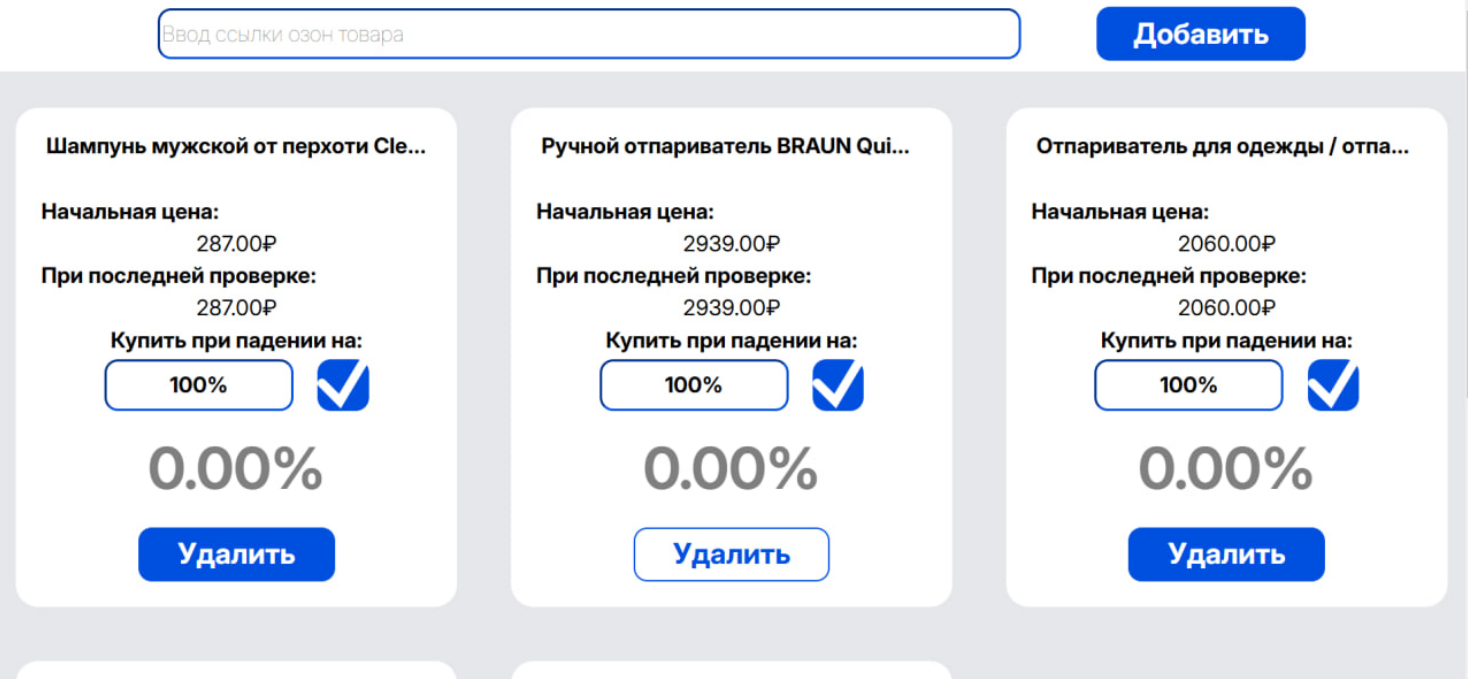


Рисунок 2 – Главная страница сайта

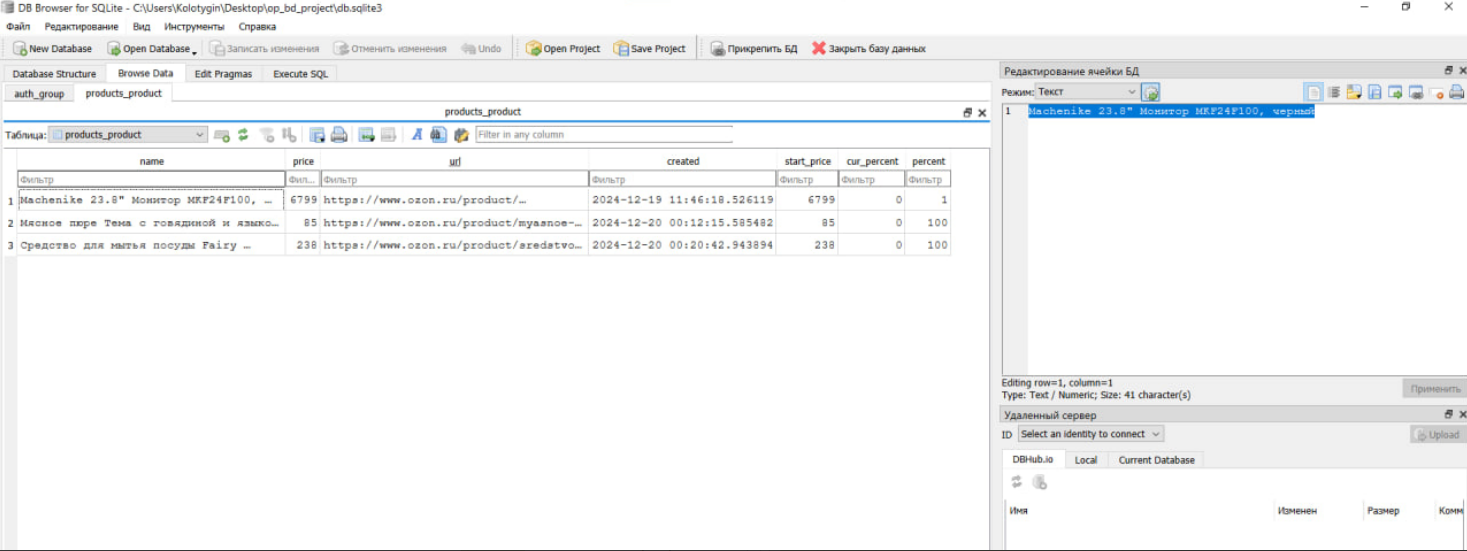


Рисунок 3 – Локальная база данных

Вывод: в ходе выполнения данной курсовой работы мы получили практические навыки самостоятельной реализации программного продукта. Закрепили теоретические знания по дисциплине «Основы программирования», а также практические навыки по программированию на языке Python.

**Список литературы:**

1. Официальная документация Django: <https://docs.djangoproject.com/>
2. Документация Python: <https://www.python.org/doc/>

Состав команды:

* Колотыгин Даниил.
* Реброва Анастасия.

Команда противников:

* Ильин Артем.
* Фролова Ксения.

Ссылка на GitHub: https://github.com/Daniil888-m/OzonParser.