МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное автономное образовательное
учреждение высшего образования
ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
Институт высоких технологий и пьезотехники
Направление подготовки 27.03.03 — «Системный анализ и управление»

Торосян Артем Каренович, 3 курс, 1 группа

Курсовая работа

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ БАЗЫ ДАННЫХ СРЕДСТВАМИ ЯЗЫКА ПРОГРАММИРОВАНИЯ PYTHON

Научный руководитель:

Е. В. Корохова

_____ _____

оценка(рейтинг) подпись руководителя

Задание на написание курсового проекта студента бакалавриата

Направление подготовки: Системный анализ и управление

Студент: Торосян Артем Каренович

Научный руководитель: Корохова Е. В.

Год защиты: 2021

Тема работы (курсовой): «Разработка веб-приложений с использованием базы данных средствами языка программирования Python»

Цель работы: обосновать возможности языка программирования Python для разработки веб-приложений с использованием базы данных и представить элементы реализации веб-сайта с использованием базы данных средствами языка.

Задачами курсового проекта являются:

- обзор и анализ существующих веб-сервисов, разработанных с помощью языка Python;
- обоснование выбора инструментальных средств (фрэймворк, СУБД, хостинг и т.п.) и языка основного разработки веб-сервиса (Python); особенности разработки с помощью языка Python, преимущества и недостатки;
- описание и характеристика основных этапов разработки веб-сайта: планирование структуры сайта, включающей личный кабинет пользователя с историей заказов, и административную панель; описание схемы базы данных;
- реализация одного из компонентов веб-сайта выбранными инструментальными средствами (фрагмент листинга).

Введение

На сегодняшний день язык программирования Python является одним из самых популярных в мире среди разработчиков. Для того, чтобы объяснить такую популярность языка, необходимо провести анализ сайтов, написанных на Python, и понять, какой функционал может быть реализован средствами языка. Также необходимо определиться со стеком используемых технологий - фреймворком, СУБД и хостингом для будущего пробного веб-приложения.

В качестве тематики будущего ресурса была выбрана сфера изготовления шоколада и конфет ручной работы (хобби автора данной работы). Изначальной задумкой было сделать сайт-галерею собственных работ с возможностью ознакомления сразу со всей информацией о каждом продукте. Сайт также должен иметь возможности авторизации пользователя для отслеживания заказов и участия в программе лояльности.

Целью данной работы является обоснование возможности использования языка программирования Python в разработке сайта с использованием базы данных и реализация его компонентов с помощью выбранных технологий.

Основные задачи:

- обзор и анализ существующих веб-сервисов, разработанных с помощью языка Python;
- характеристика веб-разработки с помощью Python, преимущества и недостатки;
- выбор фреймворка, который позволит реализовать весь необходимый функционал веб-приложения;
- выбор СУБД, которая обеспечит быстрый доступ к данным и надежное хранение данных;
- реализация компонентов сайта выбранным инструментальными средствами

Глава 1. АНАЛИЗ ЯЗЫКА ПРОГРАММИРОВАНИЯ PYTHON И ВЫБОР ПОДХОДЯЩИХ ТЕХНОЛОГИЙ

1.1 Примеры веб-приложений, написанных с использованием Python

Многие веб-сайты, которые используются ежедневно, а также серверная часть их мобильных приложений написаны с помощью Python. Это неудивительно, ведь у языка много преимуществ: он прост в изучении и работе, имеет множество библиотек и фреймворков для обширного числа задач в различных областях, что сильно упрощает разработку любых проектов с его помощью. Ниже описаны самые популярные из веб-сайтов, которые написаны с использованием Python:

Google search и YouTube - компания Google активно использует Python в разработке своих продуктов. Одним из поводов послужила работа в компании создателя языка Гвидо ван Россума с 2005 по 2012 год [1]. Однако компания начала его использовать еще на заре появления, так как язык хорошо справляется с задачами обработки трафика и вычислительными потребностями поисковой системы и связанных с ней приложений. Сегодня Python является основным языком серверной разработки компании [2], а бэк-энд YouTube почти полностью написан на Python.[3]

Spotify - компания также является большим поклонником Python и использует его в первую очередь для анализа данных и рекомендаций для пользователей. Для построения сложных последовательностей по выполнению зависимых задач они создали собственный фреймворк Luigi, с 2012 года являющийся open-source инструментом.[4]

Uber - выбирая между Python и Ruby, компания выбрала последнее, потому что платформе Uber нужно выполнять множество математических вычислений, для которых написано множество библиотек на Python, так

как серверная часть приложения должна прогнозировать время прибытия, спрос, трафик и многое другое. Еще одной причиной стала простота изучения Python и отсутствие сложностей с наймом сотрудников.[5]

Также в список сайтов, написанных с помощью Python, можно включить облачное хранилище *DropBox*, новостной сайт *Reddit*, социальную сеть *Pinterest*, стриминговый сервис *Netflix* и множество других.

Таким образом, с помощью Python действительно можно разрабатывать веб-сайты, причем по отзывам разработчиков этот процесс довольно быстро и просто.

1.2. Характеристика языка программирования Python

Из содержания предыдущей главы понятно, что Python довольно активно используется в разработке веб-сайтов даже крупными компаниями. На данный момент он является самым популярным языком программирования.[6] Рассмотрим достоинства и недостатки использования языка.

Python - это мультипарадигменный язык программирования, имеющий утиную типизацию. Его философия направлена на улучшение читаемости кода и его качества, а также повышение удобства и производительности разработчика. Является кроссплатформенным, что позволяет вести разработку с его использованием в любой ОС. [7]

Главное достоинство Python, благодаря которому он обрел такую популярность, - его простота во всем - от написания кода до отладки, что повышает скорость разработки. Пример базовой программы на языках Python и C - втором по популярности языке программирования - соответственно:

```
print('Hello, World!')
```

```
#include <stdio.h>
int main() {
   printf("Hello, World!");
   return 0;
}
```

Python имеет множество фреймворков и библиотек, которые позволяют найти ему применение практически в любой области. С помощью него можно создать мобильные приложения (фреймворк Kivy), десктопные приложения (библиотека pyQt), игры (библиотека PyGame), даже системы для цифровых устройств (пример: Raspberry Pi). Основными направлениями, в которых Python стал практически

незаменим, стали веб-разработка (фреймворки Django и Flask) и машинное обучение и работа с данными (фреймворки scikit-learn и TensorFlow).

Главным и единственным недостатком Python является его производительность. По сравнению со многими другими языками он считается крайне медленным. Вот основные причины, которые можно выделить:

- Global Interpreter Lock (глобальная блокировка интерпретатора) - интерпретатор может выполнять только одну операцию единовременно, независимо от того, как много потоков имеется в программе. Эта функция автоматически решает проблему использования одной переменной разными потоками, но в то жевремя сильно замедляет выполнение многопоточных программ. [8]
- Python - интерпретируемый язык. В отличие от многих компилируемых языков Python не использует JIT-компилятор, который позволял бы выявлять нагруженные куски кода и оптимизировать их. Это полезная технология, но из-за неё в несколько раз увеличилось бы время запуска программы, так как в JIT-компиляторах используется промежуточный язык.
- Динамическая типизация — это является проблемой, так как каждый раз, мы обращаемся к переменной для чтения или записи, производится проверка и преобразование типа, а это довольно тяжелые и долгие операции.

Но все-таки этот единственный недостаток Python не может перевесить все его плюсы, поэтому он остается ведущим языком программирования.

Рассмотрим подробнее Python в веб-разработке. Все вышеперечисленные плюсы языка позволяют ему конкурировать и выигрывать на фоне таких классических языков как PHP, Ruby, Java и других.

1.3 Сравнение и выбор инструментов для реализации проекта

Существуют множество фреймворков Python для разработки веб-приложений. Фреймворк для данного проекта был выбран по таким критериям:

- наличие подробной и понятной документации;
- наличие готовых решений для аутентификации пользователя;
- оптимальность изучения для работы в будущем.

Проанализируем четыре самых популярных full-stack фреймворка[9]:

- Django - самый популярный (350 тыс. репозиториев на GitHub против 160 тыс. репозиториев у второго по популярности Python-фреймворка - Flask[10][11]) и объемный веб-фреймворк Python. Предоставляет готовые решения для множества задач, что делает разработку с его помощью быстрой. Имеет поддержку авторизации, сессий, встроенную административную панель, шаблонизатор и поддержку форм. Фреймворк использует MTV (Model-Template-View) модель, которая незначительно отличается отстандартной MVC. Весь проект здесь разделяют на несколько “приложений”, что позволяет вести независимую разработку и при необходимости переносить эти части-приложения в другие проекты.К минусам можно отнести не самую гибкую ORM и медленное развитие (так как Django - монолитный фреймворк). Из-за своеймонолитности не подходит для сайтов, где нужно генерировать много контента на лету (как и все остальные full-stack фреймворки)

Прежде, чем продолжать, стоит оговориться, что хоть на официальном сайте следующие фреймворки и указаны как самые

популярные, на самом деле, ни один из них не может сравниться по популярности с Django.

- TurboGears 2.0 - этот веб-фреймворк должен был учесть все ошибки предыдущих, включая свою первую версию, Django и Rails. Состоит из нескольких WSGI компонентов (Pylons, SQLAlchemy, Repoze и Genshi). Turbogears имеет архитектуру MVC, создан для быстрой разработки веб-приложений и работы программистов с привычными им инструментами - можно подключить любые WSGI компоненты, однако это может сказаться на производительности. Есть система аутентификации из Repoze и встроенная административная панель. К преимуществам можно отнести его гибкость - он может вести себя не только как full-stack фреймворк, но и как микрофреймворк, что позволяет создать небольшое простое приложение и впоследствии расширить его.
- Masonite - самый молодой (создан всего 3 года назад), но активно развивающийся (на момент написания работы последнее обновление в репозитории было 10 часов назад[12]) фреймворк, как его описывает сам создатель “созданный разработчиками для разработчиков”. Как и Django имеет множество готовых решений для разных задач (отправка писем, уведомления, напоминания). Имеет большие возможности автоматической генерации кода, что ускоряет написание кода[13]. Имеет исчерпывающую документацию, а также платные и бесплатные официальные обучающие материалы. Фреймворк использует MVC архитектуру, пословам разработчиков, самую настоящую среди Python-фреймворков[13]. У Masonite активное дружелюбное сообщество, на вопросы отвечают даже сами разработчики.[14] Тем

не менее русскоязычное сообщество пока не приняло этот фреймворк, информации и статей о нем практически нет.

- web2py - фреймворк, похож на Django. Так же имеет встроенную визуализацию базы данных, административную панель, хорошую степень защиты от разных веб-атак, имеет поддержку REST и форм, а также систему аутентификации. Сами создатели описывают его как “более компактный Django”.[15] Имеет мало обучающих материалов. К плюсам можно отнести неявное подключение всех модулей к каждому проекту и хорошую документацию.

Практически ни один сайт не может работать без базы данных, особенно интернет-магазин. Поскольку Python сам по себе является медленным, непозволительно иметь медленную СУБД. Так как в дальнейшем планируется развивать бизнес, то будет расти и нагрузка на сайт, поэтому СУБД должна хорошо справляться с параллельным подключением большого количества пользователей. Рассмотрим возможные решения для проекта:

- SQLite - является базой данных по умолчанию для языка Python.[16] Является файловой БД и за счет этого обеспечивает очень быстрый доступ к данным. Однако её не используют для многопользовательских приложений, так как увеличивается время доступа к данным, что замедляет приложение; [17]
- PostgreSQL - самая полнофункциональная СУБД, полностью соответствующая стандартам SQL. Также является объектно-ориентированной БД, что хорошо подходит для работы с моделями в приложении. Является мощным инструментом и ориентирована на работы с большими объемами данных. К минусам можно отнести не такую большую популярность как у MySQL и других СУБД, из-за чего могут возникнуть проблемы с поиском

хостинга, а также меньшая скорость из-за большого объема функционала; [17]

- MySQL - довольно популярная СУБД, которую можно использовать для небольших проектов. Лучше всего подходит для веб-разработки. Имеет высокую надежность и скорость работы, которая вытекает из неполного соответствия SQL-стандартам. Также к недостаткамможно отнести снижение скорости при одновременных операциях чтения-записи. [17]

На основании проведенного анализа был выбран фреймворк Django, как самый широко распространенный и самый развитый, также привлекает их панель администратора и подробная документация. Так как он является популярным, вокруг него собралось большое сообщество и практически на все возникающие проблемы уже есть ответы в документации и на сайтах по типу StackOverflow, что тоже облегчит разработку

Среди СУБД был сделан выбор в пользу MySQL, так же из-за популярности и её скорости.

В выборе хостинга важными параметрами является поддержка используемых инструментов, местонахождение на территории нашей страны, а также тарифы размещения. Так как с поддержкой инструментов проблем не возникает из-за их широкой распространенности, выбор будет основан на тарифных планах среди популярных хостингов, находящихся в России. Таким образом, был выбран хостинг Спринг Хост из-за самой низкой стоимости (118 р/мес в сравнении с 160-170 р/мес у многих других), а также хороших отзывов об их технической поддержке.

Глава 2. РЕАЛИЗАЦИЯ КОМПОНЕНТА ПРОЕКТА ВЫБРАННЫМИ ИНСТРУМЕНТАМИ

2.1. Планирование структуры сайта и базы данных

Структура веб-приложения следующая:

- каталог товаров;
- страница каждого продукта;
- корзина;
- личный кабинет пользователя, для отслеживания заказа и в будущем получения скидки постоянного покупателя;
- страница с информацией о доставке и бренде;
- административная часть для добавления новых продуктов и отслеживания заказов.

На рис. 1 приведена итоговая схема базы данных. Изначально была также таблица Tastes, которая должна была отвечать за вкусы, которых по бизнес-логике несколько у каждого продукта. В следующем разделе будет описана проблема из-за которой от нее пришлось отказаться.

Таблица Users уже представлена в Django, поэтому документация советует только расширять её с помощью новых таблиц, связанных с основной. Так появилась связанная таблица MyUsers, которая имеет поле с номером телефона.

Таблица Orders, как понятно из названия, отвечает за заказы пользователей, которых у них может быть много.

Таблица OrderComposition - это одна составляющая заказа, являющаяся одним продуктом, их может быть много у одного заказа.

Таблица Products отвечает за информацию о каждом продукте: название, время изготовления, цена, картинка для отображения на сайте и статус (в наличии, временно нет или больше не изготавливается).

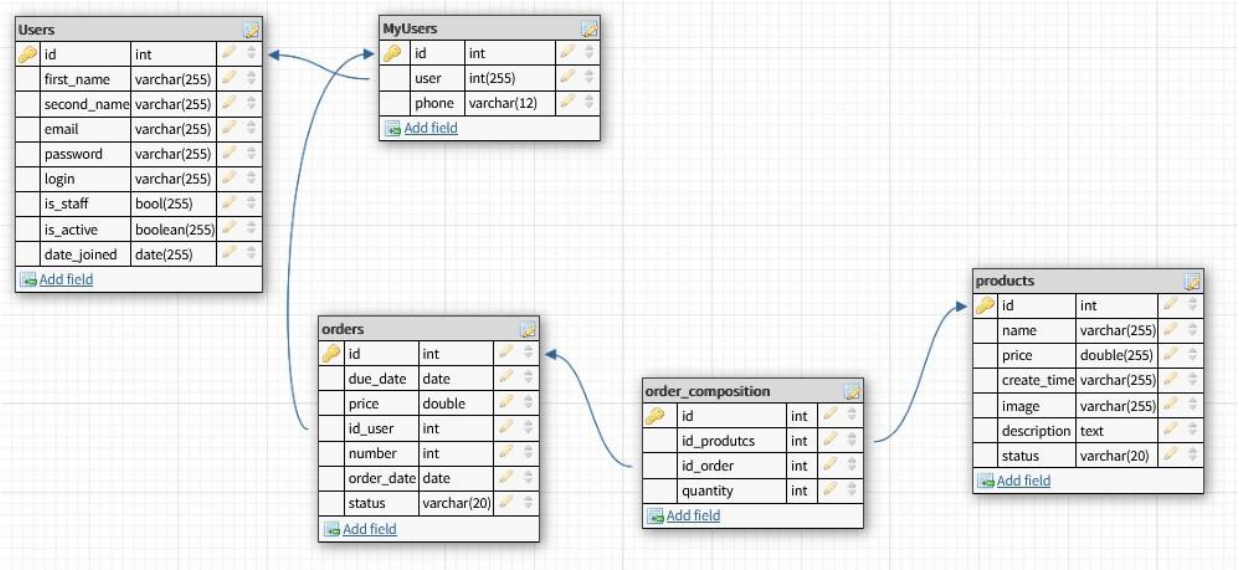

рис. 1 Схема базы данных проекта

2.2 Реализация компонентов сайта

Ниже приведен код создания модели с помощью Django на примере таблицы OrderCompositions.

```
class OrderComposition(models.Model):

                id_order      =      models.ForeignKey('Order',
to_field='id',on_delete= models.PROTECT)

   quantity = models.IntegerField()

      id_product  =  models.ForeignKey('Product',to_field='id',
on_delete=models.PROTECT)

   price = models.FloatField(null=True)


   def__str__(self):

       return str(self.id)
```

После выполнения миграции Django сам создаст таблицу с соответствующими полями в указанной в настройках базе данных. Функция __str__- обязательное требование к моделям, чтобы в дальнейшем была возможность отображать информацию о модели в административной панели.

Рассмотрим процесс верстки страниц. В оформлении сайта были использованы фирменные цвета (бежевый, коричневый, персиковый). Каталог представлен в виде галереи фотографий собственных работ. При нажатии на картинку можно перейти на страницу соответствующего товара.

Django использует подход MTV (Model-Template-View), он отличается от стандартной модели MVC (Model-View-Controller) тем, что в первом подходе представление (view) определяет, какие именно данные будут переданы пользователю, а как их отобразить определяет шаблон

(template), а для MVC, наоборот, важно, как именно данные будут переданы пользователю.

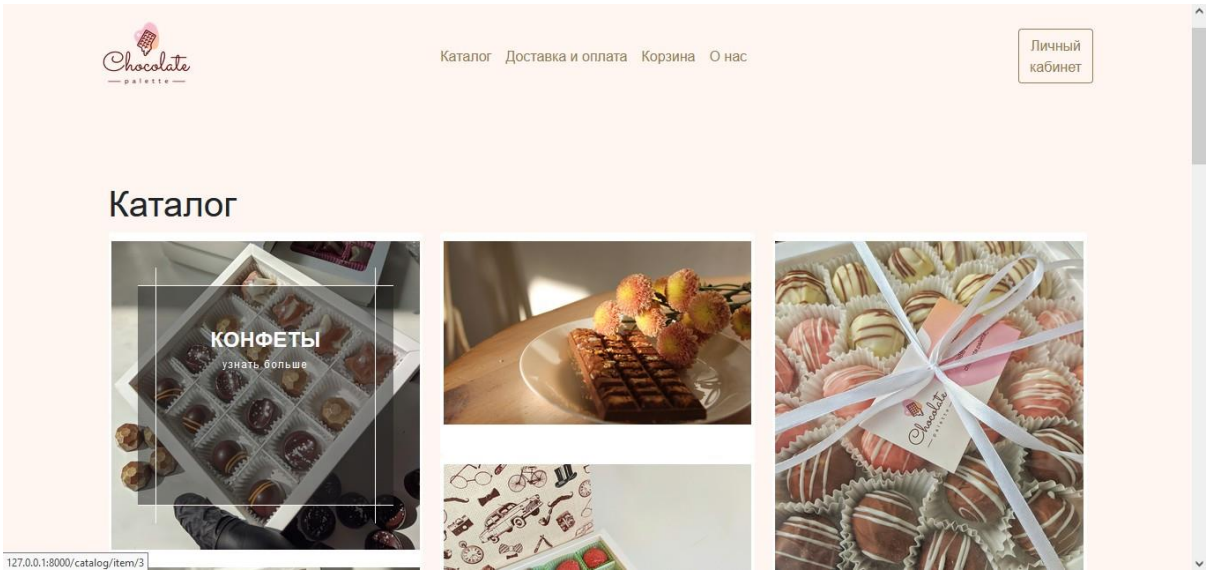

рис. 2 Внешний вид главной страницы сайта

За переходы между страницами отвечают отображения (представления, views) - они же контроллеры в стандартной модели MVC. Отображение выбирается в зависимости от URL, которые указываются отдельно и могут быть заданы также с помощью регулярных выражений. Пример представления для показа страницы продукта представлен в листинге 1.

В примере также можно увидеть использование форм для сбора данных. С ними связана проблема, из-за которой пришлось избавиться от таблиц вкусов. Дело в том, что формы в Django очень завязаны на моделях, есть даже специальные формы для создания моделей, в которых фреймворк сам создаст нужные поля. Однако в итоге не удалось сделать так, чтобы Django считал выбранный вкус входящим в список вкусов, отсортированных по одному конкретному продукту, номер которого можно получить только из отображения.

Также в коде примера оставлено взаимодействие с базой данных, которое производится через модели, здесь представлены создание, фильтрация и получение данных о модели из БД.

Когда вся логика отображения отработала, оно рендерит страницы из шаблонов. В Django используется шаблонизатор Jinja2, позволяющий расширять шаблоны, вставлять в них простейшие операции, такие как циклы или условные операторы и отображение на странице значения переменных. Примеры оформления показаны ниже:

```
{{ product.name }}
```

отображение переменной

```
{% if comp.id_product.id ==
product.id %}
{% endif%}
```

условный оператор

Во время рендеринга страницы вместо этого кода будут подставлены нужные значения, которые передаются в качестве словаря контекста из отображения.

Административная панель Django позволяет управлять таблицами, которые отслеживает приложение. По сути, это удобное отображение БД прямо на сайте. Можно выдать права нескольким отдельным пользователям или целым группам, например, системным администраторам. Внешний вид представлен на рис. 3.

Кроме того, реализована авторизация пользователя. Django предоставляет инструменты, чтобы сделать это буквально за несколько строк кода. В дальнейшем можно настроить, какие из страниц будут отображаться пользователю, с помощью специальных декораторов, отображать историю его заказов и отслеживать его активность.

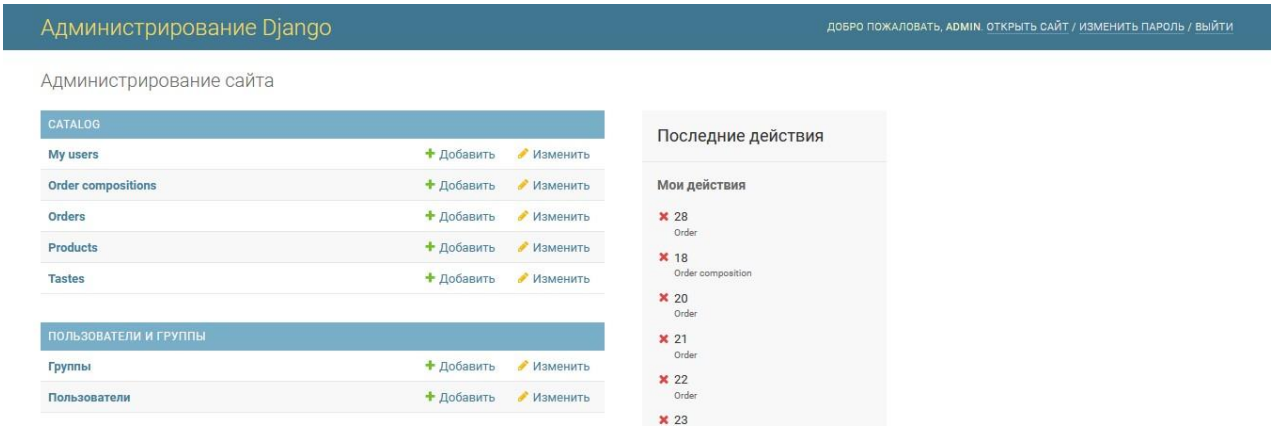

рис. 3 Панель администратора Django

Заключение

В рамках данной курсовой работы было проведено исследование оптимальности выбора языка Python для веб-разработки, сравнительный анализ его популярных full-stack фреймворков, выбрана подходящая для проекта СУБД. По итогам исследований в качестве технологии был выбран фреймворк Django, в качестве СУБД - MySQL. В качестве хостинга выбран платный Спринг Хост.

Также был реализован частичный функционал сайта:

- авторизация пользователя и личный кабинет с историей заказов;
- добавление товаров в корзину;
- оформление заказа;

Литература

1. Резюме Гвидо ван Россума [Электронный ресурс Интернет]. GitHub. URL: https://gvanrossum.github.io/Resume.html
2. Страница Google [Электронный ресурс Интернет]. GitHub. URL: https://github.com/google
3. Youtube [Электронный ресурс Интернет]. Википедия. URL: https://ru.wikipedia.org/wiki/YouTube
4. Репозиторий фреймворка Luigi от Spotify [Электронный ресурс Интернет]. GitHub. URL: https://github.com/spotify/luigi
5. Про выбор Python и Ruby от разработчика Uber [Электронный ресурс Интернет]. Quora. URL: https://qr.ae/pvCuzb
6. Список самых популярных языков программирования [Электронный ресурс Интернет]. Tiobe. URL: https://www.tiobe.com/tiobe-index/
7. Python [Электронный ресурс Интернет]. Википедия. URL: https://ru.wikipedia.org/wiki/Python
8. Визуализация GIL в Python [Электронный ресурс Интернет]. Dabeaz. URL: http://dabeaz.blogspot.com/2010/01/python-gil-visualized.html
9. Статья о фреймворках Python [Электронный ресурс Интернет]. Сайт Python. URL: https://wiki.python.org/moin/WebFrameworks
10. Репозитории с Django [Электронный ресурс Интернет]. GitHub. URL: https://github.com/search?l=Python&q=django&type=Repositories
11. Репозитории с Flask [Электронный ресурс Интернет]. GitHub. URL: https://github.com/search?l=Python&q=flask&type=Repositories
12. Репозиторий фреймворка Masonite [Электронный ресурс Интернет]. GitHub. URL: https://github.com/MasoniteFramework/masonite
13. Чем Masonite отличается от Django [Электронный ресурс Интернет]. MasoniteCasts URL:

https://masonitecasts.com/articles/deploying-masonite-to-pythonanywhere-5RJ9PT

14. Ответ от создателя Masonite на статью [Электронный ресурс Интернет]. Dev.to. URL: https://dev.to/josephmancuso/comment/b22f
15. Сайт фреймворка web2py [Электронный ресурс Интернет]. Сайт web2py. URL: http://www.web2py.com/init/default/what
16. Статья об SQLite3 [Электронный ресурс Интернет]. Сайт Python. URL: https://docs.python.org/3/library/sqlite3.html
17. Иван Бирюков, SQLite, MySQL и PostgreSQL: сравниваем популярные реляционные СУБД [Электронный ресурс Интернет]. Tproger. URL: https://tproger.ru/translations/sqlite-mysql-postgresql-comparison/

Приложение

```
    def item(request, id_item):
        form_data = request.POST if request.method == "POST" else None
        tastes_form = TasteForm(form_data)
        product = Product.objects.get(id=id_item)
        context = {'product': product, 'form_taste': tastes_form}
        if request.method != "POST" or not tastes_form.is_valid():
            return render(request, "catalog/item.html", context)
        user_id = User.objects.get(id=request.user.id)
        num = Order.objects.order_by('number').last()
        if num is None:
            num = 1
        elif num.status == '0':
            num = num.number
        else:
            num = num.number + 1
        order_id = Order.objects.get_or_create(id_user=user_id, status='0', number=num)
```

```
order_comp                                                    =
OrderComposition.objects.filter(id_order=order_id[0],
id_product=id_item)

        if order_comp.count() != 0:

             order_comp=OrderComposition.objects.get(id_orde
          r=order_id[0], id_product=id_item)
          order_comp.quantity+=tastes_form.cleaned_data['quant
          ity']

           order_comp.save()

        else:


OrderComposition.objects.create(id_order=order_id[0],
price=product.price,id_product=product,**tastes_form.cleaned_d
ata)

          return  render(request,  'catalog/succesful_add.html',
{'product': product})
```

Листинг 1. Отображение для показа страницы продукта