

Сунь Юэ, Магистрант,  
Амурский государственный университет

## ОСНОВЫ РАЗРАБОТКИ И ПРОДАЖИ ВЕБ-САЙТОВ: ОТ РЕАЛИЗАЦИИ САЙТА ДО ФУНКЦИОНАЛЬНЫХ МОДУЛЕЙ САЙТА

**Аннотация.** Ключевой этап разработки всего проекта направлен на преобразование проектной документации в действительно используемый программный продукт, а также на проверку его функциональности, стабильности и плавности пользовательского опыта через системное тестирование. Процесс реализации включает не только написание кода, но и выбор технологий, построение архитектуры, проектирование базы данных, интеграцию фронтенда и бэкенда, а также множество других сложных инженерных задач.

**Ключевые слова:** Правильность, стабильность, плавность.

Реализация данной торговой платформы использует современный стек технологий веб-разработки, чтобы обеспечить высокую производительность системы, её высокую доступность, масштабируемость и удобство сопровождения. Общая архитектура основана на шаблоне проектирования MVC (модель-вью-контроллер) с разделением фронтенда и бэкенда [2], что позволяет проводить разработку параллельно и облегчает последующее сопровождение и расширение функционала.

Выбор технологического стека. Бэкенд. Используется Python + Django. Django известен своей функцией «из коробки» [1], включает мощный ORM (объектно-реляционное отображение), систему аутентификации, панель администратора, что позволяет быстро создавать надежные серверные сервисы. Фронтенд. Используются HTML5, CSS3 и JavaScript в качестве основы [4], а также фреймворк Bootstrap для реализации адаптивной верстки, обеспечивающей корректное отображение сайта на ПК, планшетах и смартфонах. Динамическое взаимодействие разрабатывается с использованием jQuery и чистого JavaScript. База данных. Выбрана MySQL [3] в качестве системы управления реляционными базами данных. Она стабильна по производительности, имеет активное сообщество, хорошо интегрируется с Django и эффективно обрабатывает структурированные данные о товарах, заказах и пользователях. Сервер. Окружение разворачивается с Nginx в качестве обратного прокси и сервера статических ресурсов совместно с Gunicorn в качестве WSGI-сервера для работы приложения Django [5].

Ключевые сторонние сервисы. Платежные интерфейсы, Интеграция с Alipay и WeChat Pay в песочнице для обеспечения безопасности и стабильности платежных процессов. Хранение изображений, Используется Qiniu Cloud или другие сервисы объектного хранения для хранения изображений товаров, загруженных пользователями, что снижает нагрузку на сервер и ускоряет загрузку через CDN (сеть доставки контента). Кэш: Используется Redis в качестве базы данных для кэширования, для хранения сессий пользователей [6], популярных товаров, информации о корзине и т.д., что значительно повышает скорость отклика системы. Проектирование базы данных: Проектирование базы данных следует третьей нормальной форме, чтобы уменьшить избыточность данных и обеспечить их целостность и согласованность [7]. Основные таблицы: Таблица пользователей (Users). хранит информацию о регистрации пользователей, такую как имя пользователя, зашифрованный пароль, электронная почта, номер телефона, адрес и т.д.

Таблица товаров (Products). хранит информацию о товарах, включая название, описание, цену, количество на складе, ID категории, основное изображение, изображения



деталей и т. д. Таблица категорий товаров (Categories), хранит информацию о категориях, поддерживает многоуровневую классификацию. Таблица заказов (Orders), хранит основную информацию о заказах, такую как номер заказа, ID пользователя, общая сумма, статус оплаты, адрес доставки и т.д. Таблица товаров заказов (Order\_Items), хранит конкретную информацию о товарах в заказах, связана с таблицами Orders и Products, фиксирует снимок товара при покупке (название, цена, количество). Таблица корзины (Cart), хранит товары в корзине пользователя. Таблица адресов доставки (Addresses), хранит несколько адресов доставки пользователя.

Клиентская программа является прямым интерфейсом взаимодействия обычного пользователя с платформой, и её удобство использования и полнота функционала напрямую определяют успех платформы. Мы реализовали следующие основные функциональные модули.

Модуль регистрации и входа пользователей. Реализация. Глубокая настройка встроенной системы аутентификации Django `django.contrib.auth`. При регистрации проверяется уникальность имени пользователя и электронной почты, пароль хранится с высокой степенью защиты с использованием алгоритма хеширования PBKDF2, что гарантирует, что даже в случае утечки базы данных пароли пользователей не будут легко украдены. Предоставляется функция «Запомнить меня», реализуемая через продление срока действия сессии. Реализован процесс сброса пароля через электронную почту, пользователь, нажав «Забыли пароль», получает письмо с уникальной ссылкой для сброса, срок действия которой составляет 1 час. Соображения безопасности [8], Ограничение числа попыток входа для предотвращения брутфорс-атак. Использование CSRF-токенов при отправке всех форм для защиты от межсайтовой подделки запросов.

Модуль просмотра и поиска товаров. Реализация деталей, Страница списка товаров, Реализована функция пагинации с использованием класса `Paginator` в Django, на каждой странице отображается 20 товаров. Поддерживается сортировка по цене, объему продаж, дате размещения и другим параметрам. Страница использует технологию ленивой загрузки (`Lazy Load`), изначально загружаются только изображения в видимой области, что ускоряет открытие страницы. Поиск товаров, Интегрирован полнотекстовый поисковый движок (например, Django-Haystack с Whoosh или Elasticsearch), поддерживается поиск по ключевым словам в названии товара, описании, категории, а также исправление ошибок при поиске пининем. Предоставляются умные подсказки поиска (`Autocomplete`), что повышает удобство для пользователей. Фильтрация товаров. С помощью технологии AJAX реализована фильтрация без перезагрузки страницы, пользователи могут быстро отфильтровать товары по диапазону цен, категориям товаров, брендам и другим параметрам. Страница деталей товара. Отображается основное изображение высокого разрешения (с эффектом увеличительного стекла), фотографии с разных углов, подробное описание, характеристики, отзывы пользователей, статус наличия на складе и т.д. Данные страницы популярных товаров кэшируются в Redis, что снижает нагрузку на базу данных.

Модуль корзины и управления. Детали реализации, Данные корзины хранятся в локальном хранилище браузера (`LocalStorage`) до входа пользователя в систему. После входа данные автоматически синхронизируются с базой данных на сервере. Это обеспечивает единый опыт для пользователя на разных устройствах или в разных сессиях браузера. Реализованные функции включают. Добавление товаров, Пользователь может добавлять товары в корзину на странице списка товаров или на странице деталей, при этом в реальном времени отображается общее количество товаров в корзине. Изменение количества. На странице корзины пользователь может легко увеличивать или уменьшать количество товаров, система автоматически рассчитывает и обновляет промежуточные суммы и общую стоимость.



Удаление товаров, Поддерживаются одиночное и множественное удаление. Проверка действительности товаров, Перед переходом пользователя к оформлению заказа система автоматически проверяет актуальный статус запасов и цен товаров в корзине и предоставляет четкие уведомления о недействительных товарах (например, снятых с продажи или недостатков на складе).

Модуль заказов и оплаты: Детали реализации, Это ядро процесса транзакции [9]. Мы разработали строгую конечную автоматическую систему для управления жизненным циклом заказа (ожидает оплаты -> оплачено -> отправлено -> завершено/отменено). Создание заказа, Пользователь переходит с корзины на страницу оформления заказа, подтверждает адрес доставки и информацию о товарах, затем нажимает « Отправить заказ ». Система выполняет предварительное резервирование запасов ( чтобы избежать перепродажи ) и формирует заказ со статусом « ожидает оплаты ». Интеграция оплаты, Вызов API Alipay/WeChat Pay для генерации QR-кода оплаты или перенаправления на платежный шлюз. Платформа настроила асинхронный интерфейс уведомлений (Notify) для получения обратного вызова о результате оплаты. Это критически важно для обеспечения согласованности данных. даже если из-за сетевых проблем пользователь не успел вернуться после оплаты, система сможет обновить статус заказа через обратный вызов. Центр заказов. Пользователь может просматривать все свои заказы в разделе « Мои заказы » личного кабинета и фильтровать их по различным статусам (ожидает оплаты, ожидает отправки, ожидает получения, ожидает оценки) для управления, например, отмена неоплаченных заказов, подтверждение получения, подача заявки на возврат/возврат средств и т. д.

Модуль личного кабинета. Детали реализации, Это личное пространство пользователя, объединяющее различные функции управления личной информацией. Управление личной информацией, Пользователь может изменять никнейм, аватар, номер телефона, пароль для входа и другие данные. Управление адресами доставки, Поддерживается добавление, редактирование и удаление нескольких адресов доставки, а также установка адреса по умолчанию. Мои коллекции, Пользователь может добавлять интересующие товары в закладки, чтобы удобно находить их позже. Функция оценки, Пользователь может оставлять отзывы и ставить оценки полученным товарам, поддерживается загрузка изображений; после проверки администратором отзывы отображаются на странице товара.

#### *Список литературы:*

1. Django Software Foundation. (2023). Django Documentation.
2. Fielding, R. T. (2000). Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine.
3. Oracle Corporation. (2023). MySQL 8.0 Reference Manual.
4. Mozilla Developer Network. (2023). Web technology for developers.
5. Nginx, Inc. (2023). NGINX Documentation.
6. Redis Ltd. (2023). Redis Documentation.
7. García-Molina, H., Ullman, J. D., & Widom, J. (2008). Database Systems: The Complete Book (2nd ed.). Pearson Education.
8. Rescorla, E. (2018). The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446. IETF.
9. Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley Professional.

