МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра теории функций и стохастического анализа

РАЗРАБОТКА ИНТЕРНЕТ-МАГАЗИНА ЮВЕЛИРНЫХ ИЗДЕЛИЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 38.03.05 — Бизнес-информатика

механико-математического факультета
Козорез Виктории Александровны

Научный руководитель
старший преподаватель _____ Н. В. Сергеева

Заведующий кафедрой
д. ф.-м. н., доцент _____ С. П. Сидоров

Саратов 2022

ВВЕДЕНИЕ

Актуальность темы. Современные технологии стали неотъемлемой частью жизни каждого человека. В настоящее время цифровизация проникла в большинство сфер жизни общества, включая, например, совершенно разные виды деятельности: от здравоохранения до торговли. Явление цифровизации особо ярко влияет на развитие торговли. Сейчас невозможно представить продажу товаров без использования интернет-технологий. Почти у каждого производителя имеется свой сайт и интернет-магазин, при помощи которого можно приобрести товар, не выходя из дома, оплатить его и получить доставку прямо до порога своей квартиры

Интернет-торговля позволяет охватить большое количество потребителей, в том числе и за счет расширения географии доставки. Интернет-магазины привлекают покупателей удобством оплаты, экономией времени и денег, немало важен и психологический комфорт – отсутствие очередей, консультантов, возможность в любое время вернуть товар и т.д.

Целью данной работы является разработка интернет-магазина на языке программирования Python с использованием фреймворка Flask, а также построение моделей бизнес-процессов.

Для достижения поставленной цели в работе будут реализованы следующие задачи:

— Определить предметную область и описать функционал сайта.
— Рассмотреть архитектурный стиль REST.
— Рассмотреть инструменты разработки сайта.
— Построить модели бизнес-процессов интернет-магазина.
— Разработать программный код для функционирования интернет-магазина.

Структура и содержание бакалаврской работы. Работа состоит из введения, пяти разделов, заключения, списка использованных источников, содержащего 20 наименований, и одного приложения. Общий объем работы составляет 43 страницы.

Основное содержание работы

Во **введении** обосновывается актуальность работы, формулируется цель работы и решаемые задачи.

В **первом** разделе дается определение сущности интернет-магазина, представлена классификация интернет-магазинов.

Во **втором** разделе рассмотрена архитектура REST, в частности: понятие и базовые принципы REST API и базовые принципы HTTP.

В **третьем** разделе перечислены инструменты разработки. Среди них: язык программирования Python, фреймворки Bootstrap, Flask и расширение SQLAlchemy ORM.

В **четвертом** разделе представлено описание бизнес-процессов интернет-магазина.

Описание предметной области. Целью создания интернет-магазина является реализация ювелирных изделий. После выбора нужного товара из каталога на главной странице зарегистрированный пользователь имеет возможность добавить его в корзину. Перейдя на страницу «Корзина», пользователь при помощи соответствующих кнопок может либо продолжить покупки, либо перейти к оформлению товара. Сформированная персональная корзина хранится в базе данных. Для оформления доставки пользователь должен ввести свои личные данные, такие как: ФИО, номер телефона, адрес электронной почты и адрес доставки. Если форма заполнена корректно, заказ переходит в статус ожидания и отправляется в базу данных. Не авторизованным пользователем покупка недоступна.

Администратор сайта имеет право вносить изменения в каталог товаров, управлять зарегистрированными пользователями, а также отслеживать заказы и сформированные корзины.

В структуру сайта входят следующие страницы:

— Главная страница с каталогом товаров.
— Страница авторизации.
— Страница регистрации.
— Страница с информацией о магазине.
— Корзина.
— Страница оформления заказа.
— Административная панель.

IDEF0-модель интернет-магазина. С помощью программы Ramus Education построим модель IDEF0. На рисунке 1 представлен блок контекст-

ной диаграммы.

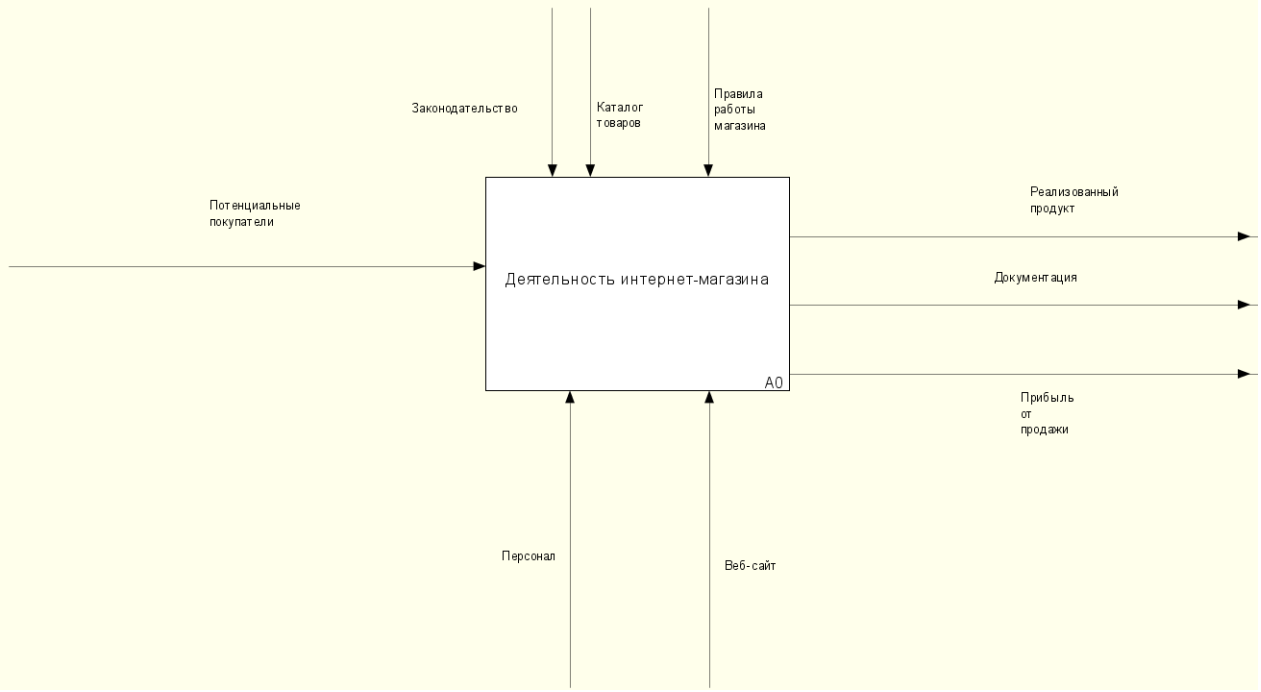

Рисунок 1 – Контекстная диаграмма

На рисунке 2 представлена диаграмма декомпозиции второго уровня.

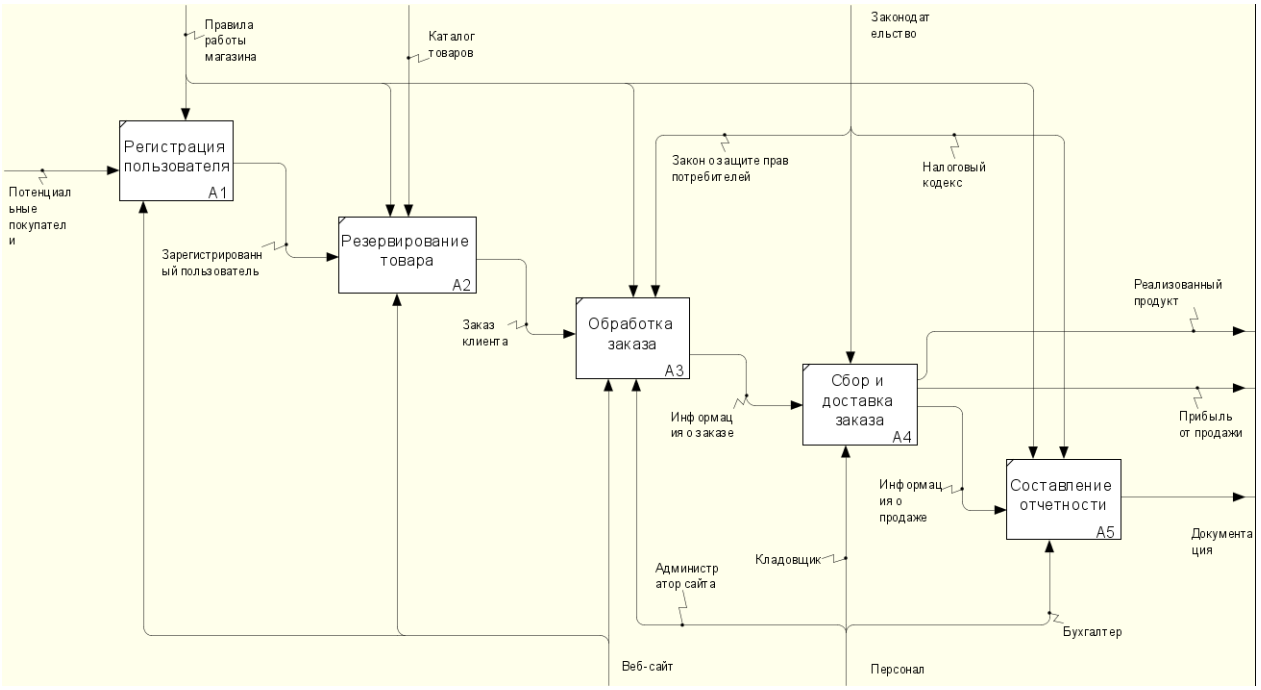

Рисунок 2 – Диаграмма второго уровня

ER-диаграмма базы данных магазина. ER-модель нужна для проектирования базы данных. С помощью данной диаграммы мы определяем

главные сущности и устанавливаем связь между ними.

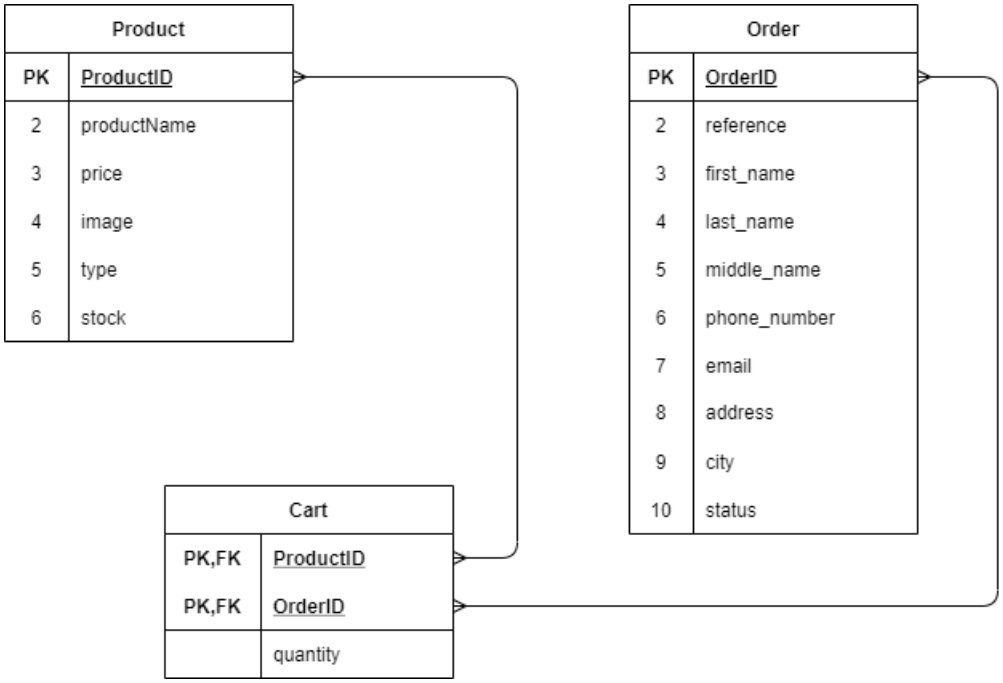

Рисунок 3 – ER-диаграмма

В сущность Product (товар) входят следующие атрибуты:

— productName (название товара);
— price (цена);
— image (изображение);
— type (тип);
— stock (количество в наличии).

В сущность Order (заказ) входят следующие атрибуты:

— reference (идентификатор);
— first_name (имя);
— last_name (фамилия);
— middle_name (отчество);
— phone_number (номер телефона);
— email (электронная почта);
— address (адрес);
— city (город);
— status (статус).

В сущность Cart (корзина) входят следующие атрибуты:

— quantity (количество).

В **пятом** разделе рассмотрен процесс разработки интернет-магазина. Описана структура приложения. Разрабатываемое приложение должно иметь следующую структуру:

```
\jewelryStore
    \website
        __init__.py
        \static
        \templates
        auth.py
        models.py
        views.py
    main.py
```

Описание каждого файла и папки представлено в таблице 1.

Таблица 1 – Структура приложения

Файл	Описание
jewelryStore	Корневая папка
website	Пакет Python с файлами представления, шаблонами и статическими файлами
__init__.py	Этот файл сообщает Python, что папка website – пакет Python
static	Папка со статичными файлами проекта
templates	Папка с шаблонами
views.py	Маршруты и функции представления
auth.py	Авторизация, выход и регистрация
models.py	Модели базы данных
main.py	Точка входа приложения

Для создания проекта библиотеки flask был импортирован класс Flask. В объект app, созданный на основе этого класса, передается имя исполняемого файла.

Для запуска локального сервера используется функция run(). Чтобы не приходилось перезапускать сервер каждый раз при изменении кода, была включена поддержка отладки. Тогда сервер перезагрузит сам себя при изменении кода. Кроме того, отладчик поможет обнаружить, локализовать и устранить возникающие ошибки.

Создав экземпляр app, мы начинаем использовать его для обработки веб-запросов и отправки ответов пользователю. Таким образом мы можем

создавать функции, которые будут отслеживать переходы на разные URL-адреса. Такие функции называются функциями представления. Проще всего определить маршрут в приложении на основе Flask с помощью декоратора app.route, экспортируемого экземпляром приложения. Декораторы функций обеспечивают способ определения специальных режимов работы функций, обертывая их дополнительным слоем логики, реализованной в виде других функций.

База данных. Для работы с базой данных используется дополнительная библиотека flask-sqlalchemy. Из установленной библиотеки импортируеотся класс SQLAlchemy.

В процессе разработки приложения часто возникает необходимость изменять модели базы данных, и когда такое случается, требуется обновить структуру самой базы данных. Так как flask-sqlalchemy способен лишь создавать таблицы, а не обновлять их, необходимо установить расширение flask-migrate и импортировать класс Migrate для организации миграции баз данных. Фреймворк миграции базы данных следит за изменениями в схеме базы данных и может последовательно применять их.

Далее в настройках было указано, какую базу данных программа будет использовать. Приложение было инициализировано с указанными настройками. В качестве базы данных используется SQLite – встраиваемая реляционная система управления базами данных (СУБД), в которой база данных представлена в виде одного файла. Используемая среда базы данных представляет собой библиотеку, с которой компонуется приложение.

Были создан таблицы Product и Order, объединенные связью «многие-ко-многим». То есть, товар может находиться в нескольких заказах и заказ может содержать несколько товаров. Чтобы создать таблицу внутри базы данных, необходимо создать класс, который будет наследовать от класса Model. Затем прописываем внутри данного класса поля. В полях обращаемся к db.Column(), внутри которого прописываем такие параметры, как тип и ограничения. Ограничение unique требует, чтобы каждое значение в столбце было уникальным. Ограничение nullable не позволяет оставить поле пустым.

Для каждой таблицы будет обязательным поле id, содержащее первичный ключ.

Каждую связь «многие-ко-многим» в реляционной базе данных нужно заменить на связь «один-ко-многим» с помощью введения дополнительных таблиц. Для этого была создана таблица Cart.

Функция create_database(app) создает базу данных. В функцию create_app() импортируются все модели и вызывается функция create_database(app).

Таким образом, была создана база данных. Если в нее понадобится внести какие-то изменения, необходимо использовать команды flask db migrate и flask db ugrade для осуществления миграции.

Регистрация. Для создания функции регистрации был написан код с использованием HTTP-запросов. В программе использовались два основных запроса: GET и POST.

В базу данных была помещена таблица User для хранения аккаунтов пользователей. Таблица User имеет следующие атрибуты: email, имя пользователя и пароль.

Далее была создана форма регистрации пользователя при помощи средств HTML. Метод POST передает данные формы на сервер, где они используются для добавления нового пользователя в БД.

Аналогичным образом работает функция **авторизации**. Форма авторизации пользователя имеет следующие поля: email и пароль.

Административная панель. С помощью административной панели можно управлять базой данных, не прибегая к использованию SQL-запросов. Для создания административной панели было использовано расширение flask-admin.

Программа имеет следующие модели: User, Product, Cart, Order. Исходя из этого, был написан код инициализации административной панели.

Корзина. Для реализации корзины были использованы сессии. Сессии предназначены для хранения сведений о пользователях при переходе между несколькими страницами. Пользователи не могут изменить содержимое сессии, не зная секретный ключ.

Была создана функция handle_cart(), которая хранит свойства корзины, а также считает количество товара, общее количество товаров и общую сумму покупки. При первом запросе пользователя сессия устанавливается

как пустой словарь. Также были реализованы функции remove_from_cart() для удаления товара из корзины и in_stock(), чтобы отслеживать количество товаров в наличие. Функцию представления index() возвращающает список товаров.

В функции быстрого добавления товара в корзину впервые используется session.modified для сохранения изменений структуры данных.

Товары в корзине будут отображены только в том случае, если количество товара больше нуля. Если товаров в корзине нет, пользователь увидит надпись «Корзина пуста».

Функция представления cart() отображает страницу «Корзина». Благодаря декоратору login_required корзина доступна только авторизованным пользователям.

Оформление. Для создания формы оформления применено расширение Flask-WTF. Форма оформления содержит следующие поля: фамилия, имя, отчество, номер телефона, email, адрес и город. Все поля, кроме отчества, должны быть заполнены корректной информацией.

Был создан класс Item_checkout(), наследующий от FlaskForm. Из расширения Flask-WTF импортированы типы полей и валидаторы. Валидация – это процесс проверки данных на соответствие различным критериям. При обнаружении несоответствия сервер выдает соответствующее сообщение.

Функция представления checkout() отображает страницу оформления товаров и заполняет атрибуты таблицы Order данными формы.

Представления, использующие FlaskForm, автоматически получают CSRF-токен для предотвращения CSRF-атак. Токены CSRF – это уникальные случайные числа, используемые один раз, генерируемые для каждого запроса к защищенной странице.

Основные результаты

1. Определена сущность интернет-магазина, представлена классификация интернет-магазинов: в зависимости от способа получения дохода и в зависимости от представляемых товаров и комплекса выполняемых функций.

2. Рассмотрена архитектура REST. Определены понятие и базовые принципы REST API, а также базовые принципы HTTP.

3. Определены инструменты, используемые в разработке.

4. Описаны бизнес-процессы интернет-магазина. Определена предметная область. Построены двухуровневая IDEF0-модель и ER-диаграмма базы данных, где были определены главные сущности и связь между ними.

5. Разработан интернет-магазин. Создана база данных для хранения информации о товарах, заказах и пользователях. Добавлены функции регистрации и авторизации пользователей. Создана административная панель. Разработана корзина и функция оформления заказа. Программный код приводится в **приложении А**.