

Е. А. Макаренко^{1✉}, А. А. Басаргин¹

Разработка автоматизированной системы управления заказами и запасами для интернет-магазина малого бизнеса с интеграцией сторонних API

¹Сибирский государственный университет геосистем и технологий,
г. Новосибирск, Российская Федерация
e-mail: egor.makarenkoxs@mail.ru

Аннотация. В условиях цифровизации и ускоренного роста сегмента электронной торговли малый бизнес сталкивается с рядом проблем при организации процессов учета товарных запасов и обработки клиентских заказов. Отсутствие доступных и адаптированных под конкретные задачи решений приводит к увеличению операционных затрат, снижению скорости обслуживания и появлению ошибок в логистике. Настоящее исследование направлено на разработку программного прототипа автоматизированной системы, которая позволит интернет-магазинам малого бизнеса эффективно управлять заказами и запасами, а также интегрироваться с внешними платформами с использованием открытых API. В работе проведен анализ существующих информационных систем, описаны технологические средства взаимодействия с API поставщиков, обоснован выбор архитектуры и стека технологий. Реализован программный прототип с использованием фреймворка FastAPI и СУБД SQLite, позволяющий автоматизировать процессы поступления, хранения и обработки информации о товарах. Полученные результаты свидетельствуют о возможности масштабирования решения и его практической применимости в сфере малого онлайн-ритейла.

Ключевые слова: автоматизация, интернет-магазин, управление запасами, заказ, API, FastAPI

E. A. Makarenko^{1✉}, A. A. Basargin¹

Development of an automated order and inventory management system for a small business online store with integration of third-party APIs

¹Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: egor.makarenkoxs@mail.ru

Abstract. In the context of digitalization and rapid expansion of the e-commerce sector, small businesses face numerous challenges in organizing inventory and managing customer orders. The lack of accessible and tailored software solutions leads to higher operational costs, slower order processing, and logistical errors. This study focuses on the development of a software prototype of an automated system that helps small online stores manage orders and inventory effectively, while also integrating with external platforms using open APIs. The paper presents an overview of existing order and inventory management systems, explores technologies for API interaction, and justifies the selected architecture and technology stack. A prototype was implemented using the FastAPI framework and SQLite database to automate the flow and processing of product-related data. The obtained results demonstrate the solution's scalability and practical value for small-scale e-commerce businesses.

Keywords: automation, online store, inventory management, order, API, FastAPI

Введение

Рынок электронной торговли демонстрирует устойчивую тенденцию к росту, охватывая все более широкий спектр бизнесов, включая малые предприятия. Особенно заметной становится цифровая трансформация в сегменте малого бизнеса, где предприниматели активно переходят к онлайн-модели продаж. Вместе с открывающимися возможностями цифровой среды усиливается и нагрузка на организационные процессы. Одной из наиболее острых проблем, с которой сталкиваются владельцы интернет-магазинов, является необходимость оперативного и точного управления товарными запасами, а также бесперебойной обработки заказов.

На практике управление запасами и заказами часто осуществляется вручную либо с применением простейших средств, таких как электронные таблицы. Такой подход, несмотря на его начальную доступность, нередко приводит к серьезным последствиям: ошибкам при комплектации, задержкам в доставке, недостоверной информации о наличии товара и, как следствие, снижению доверия со стороны клиентов. Дополнительную сложность представляет необходимость регулярно отслеживать обновления ассортимента и цен у поставщиков, синхронизируя эти данные с собственным магазином. Отсутствие автоматизации этого процесса может приводить к несоответствию информации и финансовым потерям.

В условиях высокой конкуренции и ограниченных ресурсов задача внедрения простой, но функциональной системы управления становится особенно актуальной. При этом критически важным аспектом является возможность интеграции с внешними платформами, такими как маркетплейсы и поставщики, посредством использования открытых API. Такое взаимодействие позволяет поддерживать актуальность информации о товарах, ценах и наличии, а также автоматически формировать заказы на основании свежих данных.

Настоящее исследование направлено на проектирование и реализацию программного прототипа системы, способной автоматизировать процессы учета запасов и обработки заказов в условиях малого интернет-бизнеса. В качестве фреймворка выбран FastAPI – легковесное и высокопроизводительное средство для создания REST-сервисов с авто-документацией, что существенно упрощает сопровождение и развитие проекта. В качестве хранилища данных используется встраиваемая СУБД SQLite, обеспечивающая кроссплатформенность и удобство развертывания.

Научная и практическая значимость работы заключается в демонстрации возможности решения актуальной прикладной задачи с помощью свободно расширяемых и доступных инструментов. Полученные в ходе реализации прототипа результаты могут быть использованы как отправная точка для создания более сложных автоматизированных систем управления в сфере электронной коммерции.

Методы и материалы

Проектирование и реализация прототипа системы управления заказами и запасами выполнялись с использованием современных инструментов, ориентированных на простоту внедрения, высокую скорость разработки и гибкость масштабирования. При выборе технологий основное внимание уделялось не только функциональности, но и доступности для начинающих разработчиков и индивидуальных предпринимателей, не обладающих глубокими техническими компетенциями.

В качестве среды разработки использовалась интегрированная IDE PyCharm, обеспечивающая удобную работу с проектами на языке программирования Python. Последний был выбран за счет своей читаемости, широкой экосистемы и поддержки множества библиотек, востребованных при создании веб-приложений и организации взаимодействия с API.

Основу серверной части составил фреймворк FastAPI, который позволяет создавать масштабируемые RESTful-приложения с лаконичным синтаксисом и встроенной генерацией документации в формате Swagger/OpenAPI. Это существенно облегчает интеграцию и тестирование системы как на этапе разработки, так и при последующем внедрении.

Для хранения информации использована встраиваемая реляционная СУБД SQLite, не требующая отдельной установки сервера. Благодаря простоте и легкости она является оптимальным решением для прототипов и локальных систем. При этом проектная структура допускает замену базы данных на PostgreSQL или MySQL без значительной переработки кода.

Приложение построено по модульному принципу. Модели (models.py) описывают структуру таблиц, схемы (schemas.py) – правила валидации данных средствами Pydantic, маршруты (routes.py) – обработку HTTP-запросов, а основной файл (main.py) инициализирует приложение и подключает необходимые компоненты.

Для получения актуальных данных о товарах с внешнего ресурса использовались асинхронные HTTP-запросы, возвращающие JSON-ответы. В качестве примера была реализована интеграция с сайтом крупного поставщика компьютерной техники, каталог которого обрабатывается при помощи библиотек `httpx`, `beautifulsoup4` и `re`. Этот модуль позволяет автоматически извлекать данные о товарах, их наличии и ценах, после чего информация сохраняется или обновляется в локальной базе данных.

Особое внимание в процессе разработки уделялось реализации механизмов фильтрации и предотвращения дублирования данных. Все компоненты проекта структурированы таким образом, чтобы обеспечить возможность дальнейшего расширения и адаптации системы под конкретные бизнес-задачи.

Результаты

В результате проведенной работы был разработан программный прототип автоматизированной системы, предназначенной для управления заказами и запаса-

сами интернет-магазина с возможностью интеграции внешних данных через API поставщиков. Система реализована в виде веб-приложения с архитектурой, основанной на принципах REST и модульной организации кода:

1. Архитектура и функциональные блоки

Разработанная система имеет модульную архитектуру, обеспечивающую четкое разделение логики и упрощающее масштабирование. В ее основе лежит принцип разделения ответственности, при котором каждый компонент выполняет строго определенные функции. На рис. 1 представлена подробная схема.

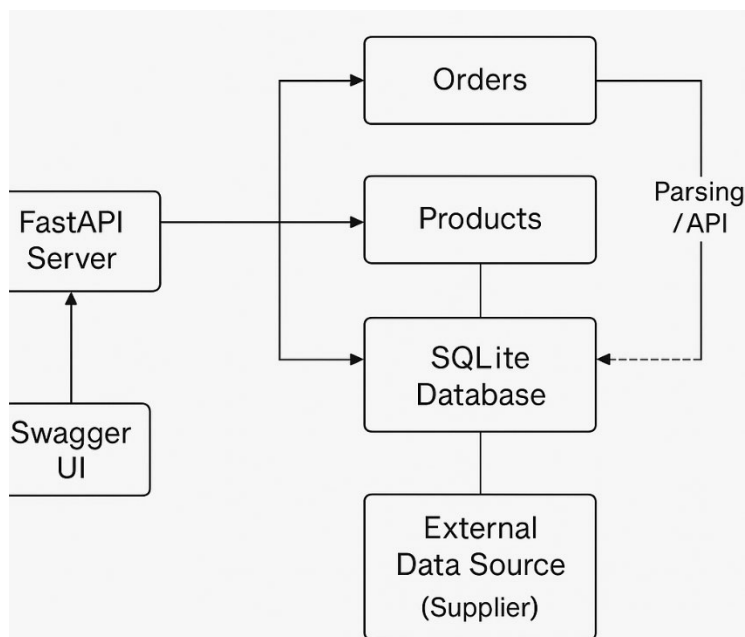


Рис. 1. Функциональные блоки

На уровне серверной логики выделяются три основных блока. Первый – модуль управления продуктами, отвечающий за операции добавления, редактирования и удаления товарных позиций. В рамках этого модуля хранится информация о наименовании, артикуле, остатке, текущей цене, а также дате последнего обновления каждой позиции.

Второй – модуль заказов, который реализует механизм обработки клиентских заявок. Он позволяет фиксировать поступающие заказы, отслеживать их текущий статус (например, «создан», «в обработке», «выполнен», «отменен»), а также автоматически уменьшать количество товара на складе после оформления.

Третий – модуль интеграции с внешними источниками, реализующий взаимодействие с API поставщиков. На данном этапе прототип поддерживает получение информации с сайта поставщика «Ситилинк» путем извлечения данных из HTML-каталога и их преобразования в формат, совместимый с внутренними структурами системы. Такой подход позволяет актуализировать базу товаров в автоматическом режиме.

Все вышеуказанные модули функционируют как независимые маршруты в структуре FastAPI-приложения. Код организован по файлам, что облегчает тестирование, сопровождение и возможную замену отдельных компонентов без нарушения работы всей системы. Структура базы данных нормализована, что исключает избыточность и повышает согласованность данных.

2. Реализация и взаимодействие компонентов

Система работает в асинхронном режиме, что положительно сказывается на скорости обработки запросов, особенно при интеграции с внешними API. Для визуализации и тестирования маршрутов используется встроенная Swagger-документация, автоматически генерируемая FastAPI.

Ниже приведены примеры реализованных маршрутов (рис. 2).

Method	Endpoint	Description
GET	/	Root
GET	/products/all	Read All Products
GET	/products	Read Products
POST	/products	Create Product
GET	/products/{product_id}	Read Product
DELETE	/products/{product_id}	Delete Product
PUT	/products/{product_id}	Update Product
POST	/sync-products	Sync Products

Рис. 2. Реализованные маршруты

3. Эффективность автоматизации

Для оценки эффективности прототипа была проведена имитационная проверка обработки заказов и обновления ассортимента. Сравнивались затраты времени на ручную обработку заказов и обновление информации о товарах с автоматическим вариантом.

Таблица 1

Сравнение трудозатрат при ручной и автоматизированной обработке

Этап работы	Вручную (минут)	Автоматизировано (минут)	Снижение времени (%)
Внесение одного заказа	10	2	80
Обновление 50 позиций	45	5	88,9

Таким образом, внедрение даже базовой версии системы позволяет значительно снизить трудозатраты и повысить точность учета. Кроме того, система может быть запущена на любом локальном или облачном сервере, не требуя мощного оборудования или длительной настройки.

Обсуждение

Разработка прототипа автоматизированной системы управления заказами и запасами показала, что даже при ограниченных ресурсах и отсутствии масштабной ИТ-инфраструктуры малый бизнес может эффективно внедрять цифровые инструменты. Практические результаты подтверждают обоснованность использования современных веб-фреймворков, таких как FastAPI, в сочетании с легкими базами данных на этапе начальной автоматизации.

Одним из ключевых достижений реализованной системы стало внедрение механизма интеграции с внешними источниками данных. На примере парсинга каталога поставщика демонстрируется возможность автоматического обновления информации о товарах – даже при отсутствии официального API. Использование асинхронных запросов и парсинга HTML позволило организовать регулярное обновление каталога без участия пользователя.

Сравнительный анализ трудозатрат до и после внедрения системы показал, что автоматизация существенно снижает долю рутинных операций и уменьшает вероятность ошибок, связанных с человеческим фактором. Особенно это важно для небольших онлайн-магазинов, где ограниченные кадровые ресурсы сочетаются с высоким объемом операционной нагрузки. Повышение прозрачности процессов и доступ к актуальной информации позволяют владельцу бизнеса быстрее принимать решения и оперативно реагировать на изменения.

Сравнение с существующими коммерческими информационными системами – такими как «1С: Управление торговлей», «МойСклад» или Odoo – показывает, что предложенный прототип выигрывает в простоте внедрения, отсутствии лицензионных ограничений и возможности полной адаптации под конкретные потребности. Однако он уступает более зрелым решениям по таким параметрам, как поддержка многопользовательского режима, финансовый учет, интеграция с онлайн-кассами и отчетностью.

Важно отметить, что текущая реализация ориентирована преимущественно на серверную часть. Пользовательский интерфейс, визуализация аналитики, а также система авторизации остаются перспективными направлениями дальнейшей разработки. Тем не менее, архитектура приложения уже сейчас позволяет наращивать функциональность по мере роста задач и масштабов бизнеса.

Заключение

Результаты проведенного исследования показали, что автоматизация процессов управления заказами и товарными запасами в малом бизнесе может быть реализована без необходимости в дорогостоящем программном обеспечении или специализированной инфраструктуре. Разработанный программный прототип подтвердил практическую возможность частичного или полного отказа от ручных операций, за счет чего снижается риск ошибок и повышается эффективность повседневной работы.

Применение стека технологий на основе FastAPI и SQLite позволило создать решение, сочетающее простоту развертывания с гибкостью дальнейшего

развития. Асинхронная модель обработки данных, реализованная в системе, способствует улучшению отклика и масштабируемости, а модульная структура проекта позволяет адаптировать функциональность под конкретные задачи бизнеса. Учитывая полученные результаты, можно говорить о высоком потенциале таких решений для повседневного использования в сегменте электронной коммерции.

В дальнейшем проект может быть расширен в сторону более тесного взаимодействия с пользователем. Разработка графического интерфейса управления через веб-браузер упростит повседневную эксплуатацию системы. Не менее важным направлением является внедрение авторизации и разграничения прав доступа, что обеспечит контроль операций и повысит уровень безопасности.

Параллельно с этим, интеграция сторонних сервисов – платежных платформ и служб доставки – позволит автоматизировать цикл обслуживания от момента оформления заказа до его передачи клиенту. Дополнительно предполагается включение аналитического блока, способного формировать отчеты, отслеживать тенденции и помогать в принятии решений на основе собранной статистики. Все это делает систему перспективной с точки зрения практического применения и дальнейшего развития в рамках малого бизнеса.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Григорьев В. А. FastAPI на практике. — СПб. : Питер, 2022. — 224 с.
2. Панкратова Н. Д. Логистика: Учебное пособие. — М.: Юрайт, 2021. — 384 с.
3. Соловьев С. И. Электронная коммерция: Учебник. — М.: КНОРУС, 2020. — 320 с.
4. Массе М. REST API Design Rulebook. — O'Reilly Media, 2011.
5. FastAPI: Documentation. URL: <https://fastapi.tiangolo.com> (дата обращения: 15.04.2025).
6. Чугунов А. В. Электронный бизнес и электронная коммерция. — СПб. : БХВ-Петербург, 2019. — 352 с.
7. Дьяков А. А. Программирование интернет-приложений на Python. — М.: ДМК Пресс, 2020. — 240 с.
8. ГОСТ Р 7.0.5–2008. Библиографическая ссылка. Общие требования и правила составления.
9. Воронин А. С. Разработка ИС малого бизнеса. — М.: Академический проект, 2018. — 288 с.

© Е. А. Макаренко, А. А. Басаргин, 2025