

Государственное бюджетное общеобразовательное учреждение города
Москвы «Школа № 2098
имени Героя Советского Союза Л.М. Доватора»

Автоматизированное управление столовой

Участники:

Ученик 11 “Т” класса ГБОУ школа №2098

Редикальцева Мария Антоновна

Ученик 11 “Т” класса ГБОУ школа №2098

Зборовский Михаил Романович

Руководитель:

Учитель информатики ГБОУ школа

№2098

Малевин Дмитрий Сергеевич

Москва, 2026

Содержание

Введение	3
Цель и задачи работы.....	3
Гипотеза	4
Анализ технических требований	5
Обоснование выбора языка программирования и используемых программных средств	6
Описание завершенного продукта	7
Безопасность данных и Docker-развертывание	10
Описание особенностей и аргументация выбранного типа СУБД.....	11
Программный код:	12
Результаты	12
Список использованных источников	13

Введение

В современном мире качественное питание становится важным аспектом повседневной жизни, особенно в учебных заведениях. Актуальность данной работы обусловлена цифровизацией всех сфер образования, при которой организация питания часто остаётся на уровне устаревшего бумажного учёта. Это создаёт очевидный разрыв между современными технологическими возможностями и реальными процессами в школьных столовых.

Выбор темы проекта обоснован необходимостью создания комплексной системы управления питанием, которая соответствовала бы современным принципам здорового питания и индивидуального подхода к учащимся. Такая система отвечает как насущным потребностям образовательных учреждений, так и соответствует общим тенденциям цифровой трансформации социальной сферы.

Ключевая проблема, которую решает проект, заключается в отсутствии эффективной связи между всеми участниками процесса питания — учениками, поварами и администрацией. Это проявляется в трудностях планирования меню, невозможности системного учёта индивидуальных особенностей учащихся, таких как аллергии и пищевые предпочтения, а также в неэффективном использовании ресурсов из-за ручного учёта и отсутствия аналитики. Разрабатываемое веб-приложение создаёт единую цифровую платформу для решения этих задач, обеспечивая удобное взаимодействие, персонализацию питания и оптимизацию работы столовой.

Цель и задачи работы

Цель проекта заключается в создании автоматизированной веб-системы управления школьной столовой, призванной заменить устаревшие бумажные методы учёта и организации питания. Данная система должна обеспечить эффективное взаимодействие между учениками, поварами и администраторами за счёт цифровизации ключевых процессов: от выбора и оплаты блюд учащимися с учётом индивидуальных пищевых особенностей до контроля остатков

продуктов, формирования заявок на закупку и генерации аналитической отчётности.

Основные задачи работы включают разработку интуитивно понятного интерфейса для каждой категории пользователей, реализацию механизмов регистрации и авторизации, внедрение функционала просмотра и оплаты меню, организацию системы учёта выданных блюд и остатков на кухне, а также создание инструментов для анализа данных и формирования отчётов. Особое внимание уделяется обеспечению безопасности хранения и обработки персональных данных, реализации устойчивой архитектуры приложения, тестированию всех сценариев использования и подготовке полной документации с инструкцией по развёртыванию и видеодемонстрацией работоспособности системы.

Гипотеза

Мы предполагаем, что внедрение веб-приложения для управления школьной столовой значительно повысит эффективность организации питания и удовлетворенность всех участников процесса. Конкретно, мы ожидаем, что:

- **Оптимизация процессов планирования питания:** Цифровая система позволит поварам заранее планировать меню с учетом сезонности, баланса рациона и популярности блюд, что сократит время на ежедневное составление планов и уменьшит пищевые отходы
- **Повышение персонализации питания:** Возможность для учеников указывать пищевые ограничения, аллергены и предпочтения позволит формировать более индивидуальный подход к питанию, что особенно важно для детей с особыми диетическими потребностями.
- **Улучшение информированности и вовлеченности:** Прозрачное и доступное цифровое меню с описанием блюд и их составом повысит осведомленность учеников и родителей о питании, что будет способствовать формированию более осознанного пищевого поведения.
- **Сокращение временных затрат и ошибок:** Автоматизация процесса сбора заказов и предпочтений сократит время на обработку информации и

минимизирует ошибки при приготовлении, связанные с неучтенными аллергенами или предпочтениями.

• **Повышение удовлетворенности всех участников:** Удобный интерфейс как для учеников (выбор блюд, отслеживание меню), так и для поваров (планирование, учет продуктов) улучшит взаимодействие между кухней и потребителями, что повысит общую удовлетворенность качеством питания и обслуживания.

Таким образом, мы предполагаем, что разработанное веб-приложение не только упростит управление работой столовой, но и станет важным инструментом для создания более здоровой, персонализированной и эффективной системы школьного питания.

Анализ технических требований

Обязательная функциональность приложения для повара:

1. Авторизация в системе.
2. Учет выданных завтраков и обедов.
3. Контроль остатков продуктов и готовых блюд.
4. Внесение заявок на закупку продуктов.

Обязательная функциональность приложения для ученика:

1. Регистрация и авторизация.
2. Просмотр меню завтраков и обедов.
3. Оплата питания (разовый платеж или абонемент).
4. Отметка о получении питания.
5. Указание пищевых аллергий и предпочтений.
6. Оставление отзывов о блюдах.

Функционал для администратора:

1. Авторизация в системе.
2. Просмотр статистики оплат и посещаемости.
3. Согласование отчетов по питанию и затратам.

Дополнительная функциональность приложения со стороны пользователя:

1. Модуль уведомлений для учеников и сотрудников.

Обоснование выбора языка программирования и используемых программных средств

Мы выбрали Python, так как это один самых популярных, а также простых языков программирования. Python является языком программирования для большинства разработчиков в сегодняшнем технологическом ландшафте и существует с начала 90-х годов. Его многочисленные преимущества и оптимизированные функции делают его главным конкурентом для проектов разработки программного обеспечения.

Flask — это легковесный веб-фреймворк для языка Python, который предоставляет минимальный набор инструментов для создания веб-приложений.

JavaScript — это многофункциональный язык программирования, работающий с HTML и CSS. HTML задает разметку сайта, CSS отвечает за внешний вид, а JavaScript все это оживляет. С помощью кода на JavaScript программист определяет, как страница отреагирует на действия пользователя.

Ядро приложения

- Python 3.8+ - основной язык программирования
- Flask - веб-фреймворк для обработки запросов и маршрутизации
- Werkzeug - выбрана для хеширования паролей как стандартный, надежный и хорошо интегрированный с Flask инструмент. Она предоставляет готовые, безопасные реализации современных алгоритмов, что исключает ошибки при самостоятельной реализации криптографических функций и гарантирует защиту данных.

- Jinja2 — шаблонизатор для генерации HTML

База данных

- SQLite3 — встраиваемая реляционная база данных

Генерация отчетов

- reportlab — создание PDF-документов с поддержкой кириллицы
- csv — встроенный модуль для генерации CSV-файлов
- json — встроенный модуль для экспорта данных в JSON

Обработка данных

- `datetime` — работа с датами и временем
- `re` — регулярные выражения для валидации
- `os` — работа с файловой системой
- `io` — операции ввода-вывода в памяти
- `functools` — декораторы и функциональное программирование

Развертывание и конфигурация

- `Docker` (опционально) - применяется для контейнеризации приложения, что обеспечивает его стабильную и воспроизводимую работу в любой среде. Использование контейнеров позволяет упаковать сервис со всеми зависимостями в изолированное окружение, гарантируя идентичное поведение на этапах разработки, тестирования и развертывания. Это значительно упрощает процесс деплоя и масштабирования системы.

Фронтенд технологии

- `HTML5` — разметка страниц
- `CSS3` — стилизация интерфейса
- `JavaScript (ES6+)` — клиентская логика

Инструменты разработки

- `Git` — контроль версий
- `VS Code/PyCharm` — среда разработки

Этот технологический стек обеспечивает надежную, масштабируемую и простую в поддержке систему, соответствующую всем требованиям проекта автоматизации школьной столовой. Каждый компонент выбран на основе баланса между функциональностью, производительностью и простотой использования в условиях образовательного учреждения.

Описание завершённого продукта

Разработано веб-приложение «Умная школьная столовая» — комплексная система для автоматизации организации питания, полностью заменяющая устаревший бумажный документооборот. Приложение доступно в любом браузере, предлагая персонализированные интерфейсы для трёх ключевых групп пользователей: ученик, повар, администратор.

Ученики могут легко регистрироваться в системе, пополнять личный баланс и выбирать питание (завтрак, обед или абонемент) из актуального меню с учётом пищевых ограничений. После оплаты выбранный тип питания резервируется за учеником. В столовой повар в своём интерфейсе в реальном времени отмечает получение питания конкретным учеником, что мгновенно фиксируется в системе.

Повара получают удобный инструмент для учёта: они видят список учеников, пришедших на питание, и одним кликом подтверждают выдачу. Система автоматически корректирует остатки продуктов, а при их недостатке самостоятельно формирует заявку на закупку, упрощая логистику.

Администраторы управляют системой через панель с аналитикой: контролируют финансовые потоки, посещаемость, утверждают заявки на продукты и формируют детальные отчёты по любому периоду. Это даёт полную прозрачность процессов и данных для эффективного управления столовой.

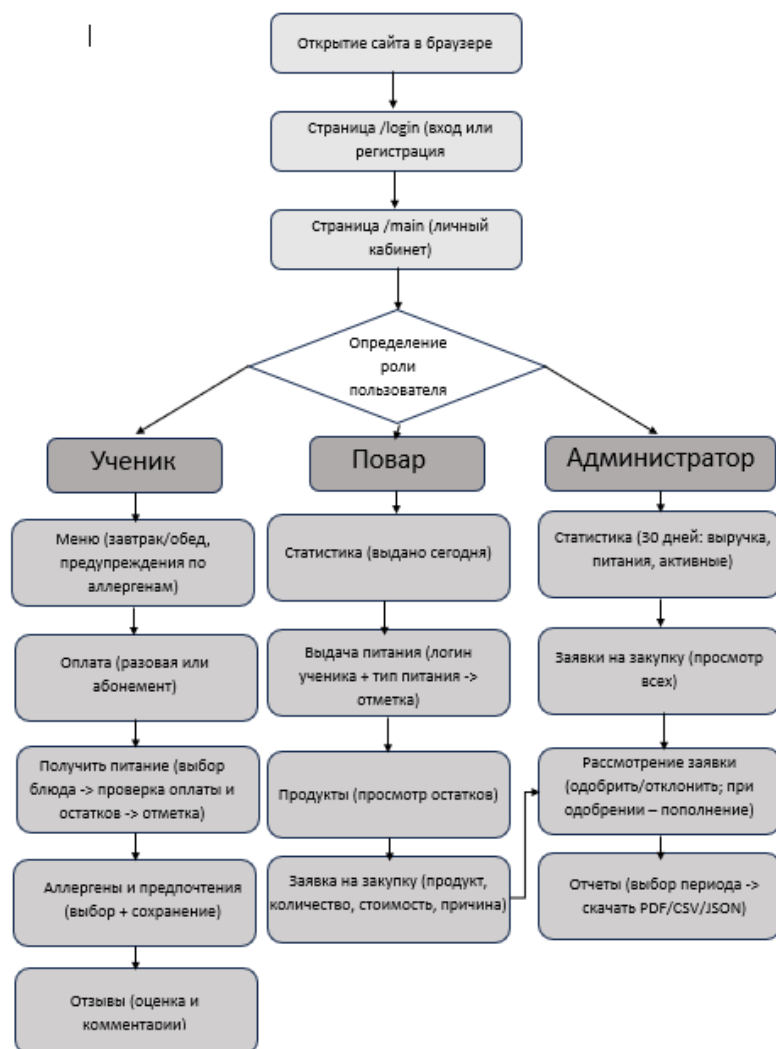


Рис.1 – Структурная схема

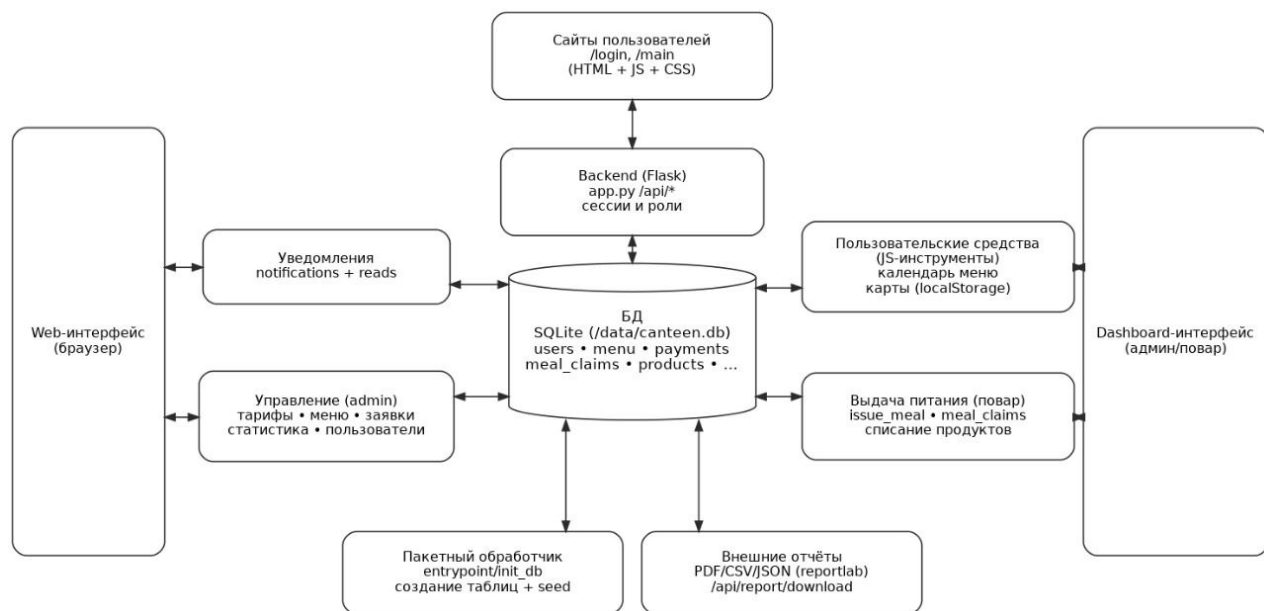


Рис. 2 – Функциональная схема.

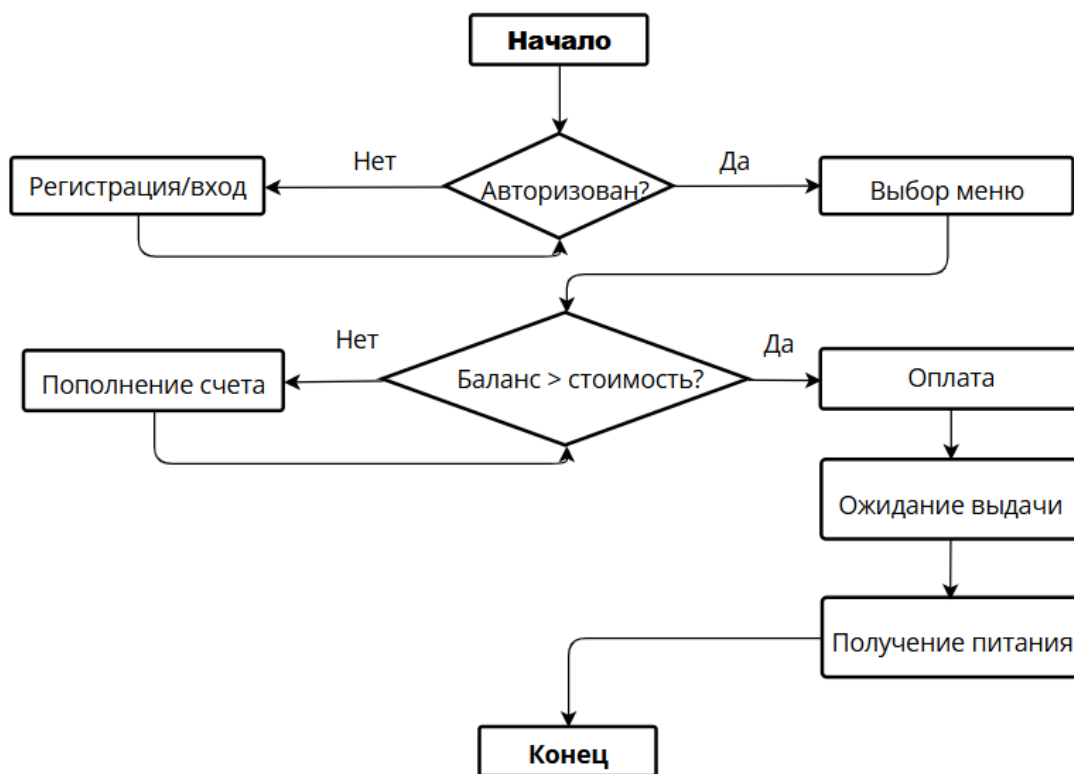


Рис. 3 - Блок-схема работы основного алгоритма (ученик)

Безопасность данных и Docker-развертывание

В проекте реализована многоуровневая защита платёжных данных, основанная на принципе токенизации, которая применяется в современных финансовых системах.

Ключевая идея заключается в том, что чувствительные данные банковской карты — её номер, срок действия, CVV-код и имя держателя — никогда не покидают безопасное окружение пользователя и ни в какой форме не сохраняются в нашей системе. Вместо этого применяется безопасная подмена: после ввода данных карты генерируется уникальный случайный идентификатор — токен, а для удобства пользователя запоминаются только последние четыре цифры номера. Именно этот токен и последние цифры, не несущие риска, передаются на сервер и хранятся в базе данных.

Серверная часть настроена так, чтобы принимать и обрабатывать исключительно эти токены. Любые попытки передать напрямую полный номер карты, CVV или другие реквизиты немедленно блокируются. Архитектура базы данных также не содержит таблиц или полей, предназначенных для хранения реальных платёжных реквизитов.

Таким образом, данный подход полностью исключает риск компрометации платёжной информации пользователей даже в гипотетических сценариях утечки базы данных или взлома сервера, обеспечивая уровень безопасности, соответствующий стандартам современных платёжных сервисов.

Docker в данном проекте позволяет обеспечить единое и воспроизводимое окружение для запуска Flask-приложения и базы данных SQLite3. Контейнеризация упрощает развёртывание проекта на любом компьютере или сервере без необходимости ручной установки зависимостей. Docker изолирует приложение от системы, что повышает стабильность и снижает риск конфликтов версий библиотек. Это особенно удобно для тестирования, демонстрации и переноса проекта между средами. В результате использование Docker ускоряет настройку проекта и упрощает его сопровождение.

Описание особенностей и аргументация выбранного типа СУБД.

В качестве системы управления базами данных в проекте выбрана SQLite3. Данная СУБД не требует установки и настройки серверной части и хранит всю базу данных в одном файле, что упрощает развёртывание и сопровождение приложения. SQLite3 поддерживает основные возможности реляционных баз

данных и язык SQL, что достаточно для реализации функционала проекта (работа с пользователями, меню, оплатами и отчётами). Проект не рассчитан на высокую нагрузку и большое количество одновременных подключений, поэтому ограничения SQLite3 не являются критичными. Таким образом, использование SQLite3 позволяет обеспечить простоту, надёжность и удобство разработки без избыточной сложности.

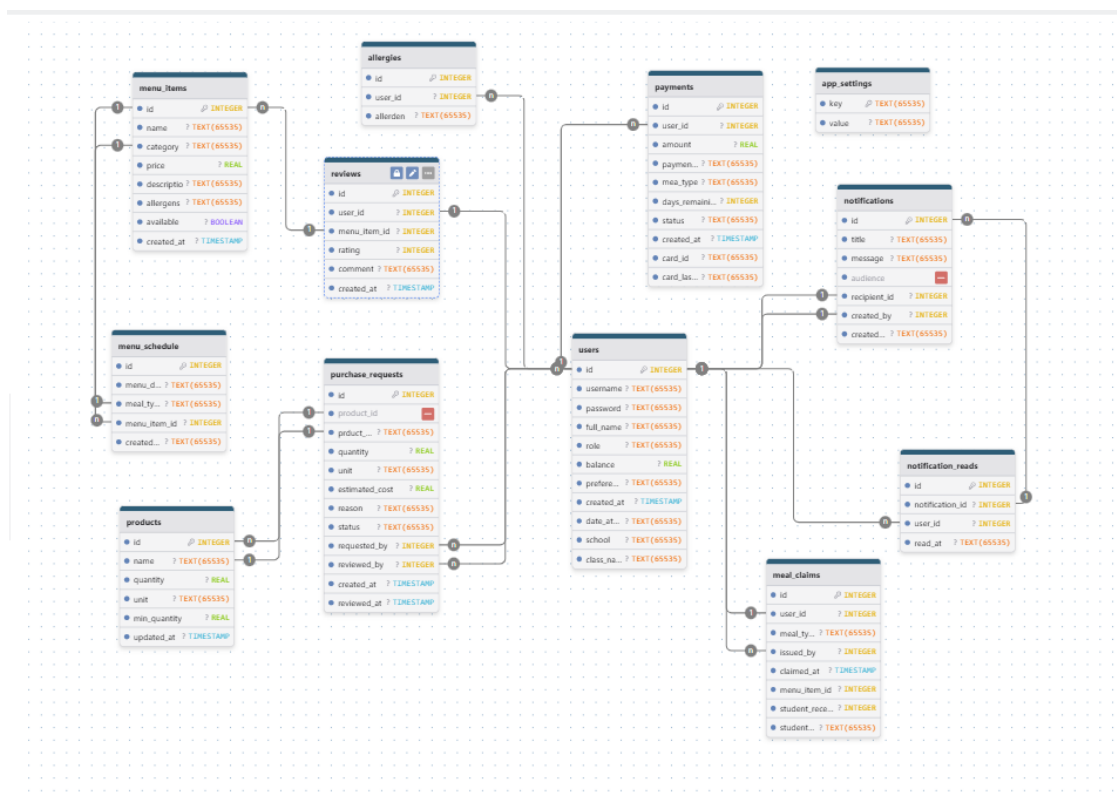


Рис. 4 – Схема организации данных

Программный код:

<https://github.com/arsIk320/predprof2026>

Результаты

В результате работы успешно разработано и реализовано веб-приложение «Умная школьная столовая» — комплексная система, полностью автоматизирующая процессы организации питания. Продукт обеспечивает удобное взаимодействие всех участников: ученики получили возможность выбирать и оплачивать питание с учётом индивидуальных особенностей, повара — эффективно вести учёт выдачи и остатков продуктов, а администрация — контролировать процессы через детализированную аналитику и отчётность.

Система готова к внедрению благодаря модульной архитектуре, безопасному хранению данных и поддержке контейнеризации через Docker. Практическая значимость проекта подтверждается полным соответствием техническому заданию и решением ключевых проблем бумажного документооборота.

Перспективы дальнейшего развития включают интеграцию с внешними сервисами (платёжными системами, бухгалтерскими программами), добавление мобильного приложения для родителей и расширение аналитического модуля с использованием технологий прогнозной аналитики для оптимизации закупок и планирования меню.

Список использованных источников

1. Flask Documentation [Электронный ресурс].
доступа: <https://flask.palletsprojects.com/en/3.0.x/> (дата обращения: 02.11.2025).
2. SQLite [Электронный ресурс]: документация.
доступа: <https://www.sqlite.org/docs.html> (дата обращения: 15.11.2025).
3. JavaScript [Электронный ресурс] // MDN Web Docs.
доступа: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (дата обращения: 29.11.2025)..
4. HTML: HyperText Markup Language [Электронный ресурс] // MDN Web Docs. – Режим доступа: <https://developer.mozilla.org/en-US/docs/Web/HTML> (дата обращения: 12.12.2025).
5. CSS: Cascading Style Sheets [Электронный ресурс] // MDN Web Docs. – Режим доступа: <https://developer.mozilla.org/en-US/docs/Web/CSS> (дата обращения: 12.12.2025).
6. The Python Language Reference, version 3.14[Электронный ресурс]. – Режим доступа: <https://docs.python.org/3/> (дата обращения: 20.12.2025).
7. Docker Documentation [Электронный ресурс]. – Режим доступа: <https://docs.docker.com/>.