Вкладка 1

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ГОРОДА МОСКВЫ "ЛИЦЕЙ "ВТОРАЯ ШКОЛА" ИМЕНИ В.Ф. ОВЧИННИКОВА"

Математико-программистский профиль

ДОКУМЕНТАЦИЯ

проекта по теме: *Командный Кейс №2 “Управление столовой”*

Ученики 10 класса

Галимов Вячеслав

Луговцова Арина

Арсланова Яна

«19» февраля 2026 г.

Москва

2026

Обоснование выбора языка программирования и используемых программных средств

Язык программирования: Python

Python был выбран в качестве основного языка программирования по следующим причинам:

- **Простота и читаемость кода.** Python отличается лаконичным синтаксисом, что облегчает командную разработку, поддержку и масштабирование проекта.
- **Большое сообщество и поддержка.** Для Python существует множество библиотек и фреймворков, что ускоряет разработку и позволяет быстро находить решения возникающих проблем.
- **Кроссплатформенность.** Приложения Python легко запускать на различных операционных системах без существенных изменений в коде.
- **Поддержка асинхронного программирования.** Это важно для современных веб-приложений, где требуется высокая производительность и обработка большого количества одновременных запросов.

Фреймворк: FastAPI

FastAPI выбран для создания серверной части по следующим причинам:

- **Высокая производительность.** FastAPI построен на асинхронной библиотеке Starlette и обеспечивает высокую скорость работы.
- **Современный подход к разработке.** FastAPI поддерживает аннотации типов, что повышает надежность кода и упрощает отладку.
- **Простота интеграции с базами данных и внешними сервисами.**

Фронтенд: HTML, CSS, JavaScript

Для клиентской части выбраны стандартные веб-технологии:

- **HTML** — для разметки страниц.
- **CSS** — для стилизации и адаптивности интерфейса.
- **Javascript** — для динамического взаимодействия с пользователем и отправки запросов к серверу.

Этот стек обеспечивает:

- **Кроссплатформенность и доступность.** Приложение работает в любом современном браузере.
- **Гибкость в дизайне и функциональности.** Легко реализовать интерактивные элементы и адаптировать интерфейс под разные устройства.

Такой выбор технологий позволяет быстро разрабатывать, тестировать и масштабировать проект.

Структурная схема программного продукта

1. Клиентская часть (front12345):

- **HTML-шаблоны:**
  - Основные страницы: main, hello, order_details, error, 404, test_error
  - Админ-панель: все файлы в папке admin/
  - Страницы регистрации и входа: все файлы в папке register_login/
- **Стили (CSS):** cafeteria_styles, dashboard
- **Скрипты (JS):** dashboard_core, dashboard_history, dashboard_order, error_render, script

- Документация по ошибкам: ERRORS_README.md

2. Серверная часть (NewAtt):

- Основные модули: main.py, auth.py, menu_parser.py, models.py, schemas.py, render_front.py, logger.py, docx_utils.py, init_db.py, verify_depencies.py
- Служебные файлы и папки: README.md, openapi.json, requirements.txt, logger_logs/, uploads/, pycache/

3. Тесты:

- test_api.py — тестирование API

Функциональная схема программного продукта

Блок-схема работы основного алгоритма

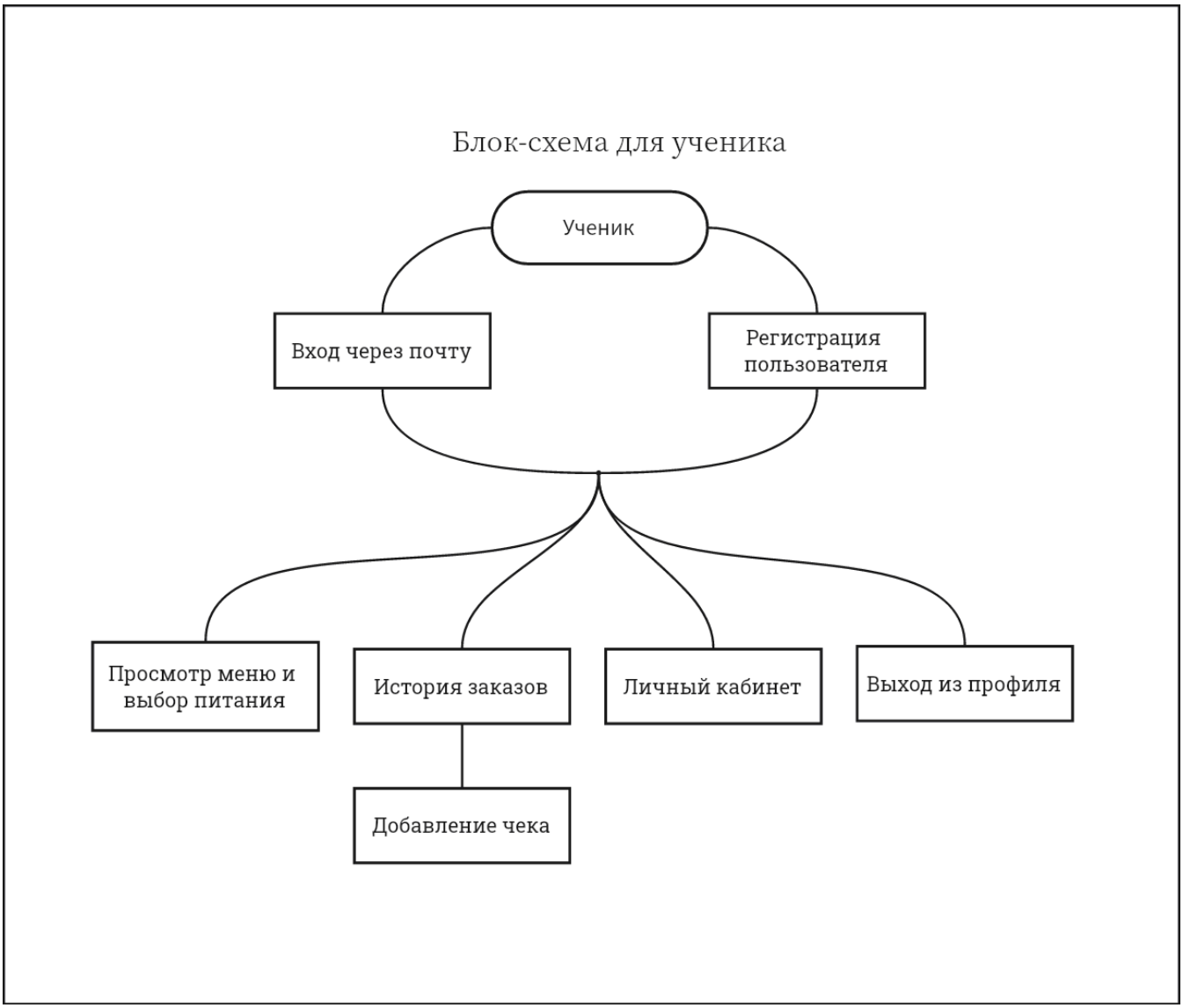

Рисунок 1 — Блок-схема для ученика.

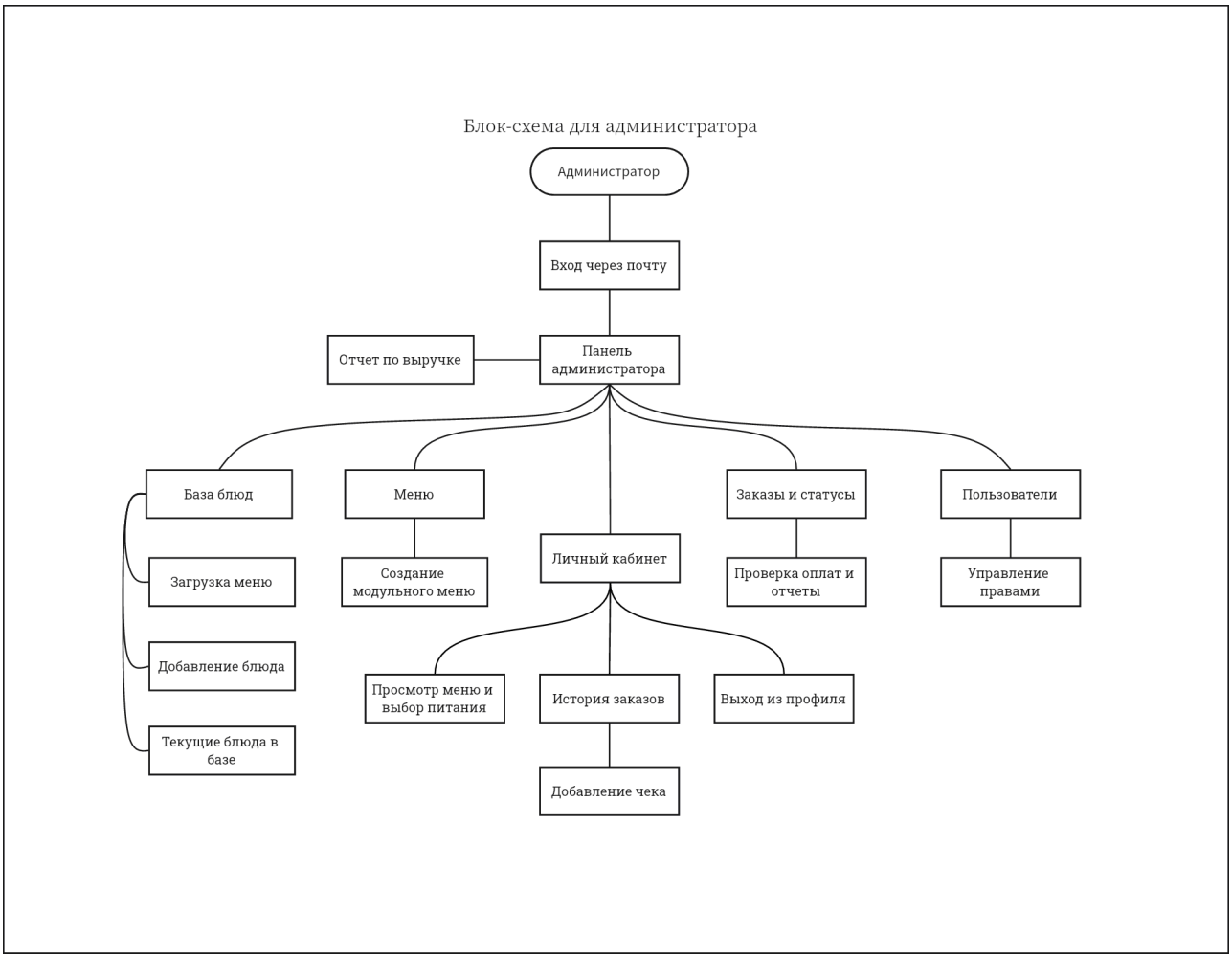

Рисунок 2 — Блок-схема для администратора.

Описание особенностей и аргументация выбранного типа СУБД

Выбранная СУБД: SQLite (реляционная база данных)

Особенности и преимущества:

- **Встроенность и простота использования.** SQLite не требует отдельного сервера или сложной настройки — база данных хранится в одном файле, что упрощает развертывание и переносимость проекта.

- **Поддержка стандартного SQL.** Это облегчает написание запросов и, при необходимости, миграцию на более мощные СУБД.
- **Высокая производительность для небольших и средних проектов.** SQLite отлично справляется с задачами, где количество одновременных пользователей и объем пользователей не превышают средние значения.
- **Минимальные требования к инфраструктуре.** Не требуется отдельный сервер или сложная настройка окружения, что особенно удобно для учебных, прототипных и небольших командных проектов.
- **Безопасность и надежность.** SQLite поддерживает транзакции, обеспечивает целостность данных и устойчивость к сбоям.

Аргументация выбора:

- **Быстрый старт и простота интеграции с Python.** Благодаря библиотеке sqlite3, входящей в стандартную поставку Python, интеграция с FastAPI и другими инструментами происходит без дополнительных зависимостей.
- **Удобство для командной работы и тестирования.** Один файл базы данных можно легко копировать, архивировать и использовать для резервного копирования или переноса между машинами.
- **Возможность масштабирования.** При необходимости, архитектура приложения позволяет перейти на более производительную СУБД с минимальными изменениями в коде, так как используются стандартные SQL-запросы и ORM.

Вывод: SQLite — оптимальный выбор для данного проекта на этапе разработки и тестирования, а также для небольших и средних командных решений. Он обеспечивает баланс между простотой, надежностью и

функциональностью, позволяя при необходимости легко перейти на более сложные решения без существенных изменений в архитектуре приложения.