

Я. В. Галицын^{1✉}, П. Ю. Бугаков¹

Разработка серверного программного обеспечения информационной системы для поддержки процесса коммуникации с клиентами компании «IT Advance»

¹Сибирский государственный университет геосистем и технологий,
г. Новосибирск, Российская Федерация
e-mail: galicin20045@gmail.com

Аннотация. В статье рассматривается задача повышения эффективности коммуникации с клиентами компании за счет создания надежной серверной части информационной системы. На практике информация о взаимодействиях с клиентами часто фиксируется вручную или с использованием разрозненных средств, что затрудняет анализ, управление и масштабирование процессов. Целью работы является разработка отказоустойчивого, масштабируемого и безопасного серверного программного обеспечения, обеспечивающего централизованное хранение данных, ведение истории контактов, авторизацию пользователей и предоставление доступа к информации через API. В реализации использованы FastAPI, PostgreSQL, Docker и современные механизмы аутентификации. Проведенное тестирование подтвердило устойчивость системы и ее применимость в условиях реальной работы компании.

Ключевые слова: серверное программное обеспечение, информационная система, клиенты, взаимодействие, API, безопасность данных, база данных

Y. V. Galicyn^{1✉}, P. Yu. Bugakov¹

Development of server software for an information system to support the process of communication with clients of «IT Advance»

¹Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: galicin20045@gmail.com

Abstract. The article addresses the challenge of improving communication efficiency with company clients by developing a reliable backend component of an information system. In practice, data on client interactions is often recorded manually or through disparate tools, making it difficult to analyze, manage, and scale communication processes. The goal of this work is to develop fault-tolerant, scalable, and secure backend software that provides centralized data storage, interaction history tracking, user authentication, and access to information via an API. The implementation uses FastAPI, PostgreSQL, Docker, and modern authentication mechanisms. Testing has confirmed the system's stability and its applicability in real-world business operations.

Keywords: server software, information system, clients, interaction, API, data security, database

Введение

В условиях цифровой трансформации бизнеса возрастает потребность в автоматизации внутренних процессов, включая обработку обращений клиентов. Эффективная коммуникация с клиентами – ключевой фактор стабильной работы компании, влияющий на уровень сервиса, лояльность и повторные обращения.

Каждый день сотрудники компании взаимодействуют с десятками клиентов: проводят консультации, фиксируют договоренности, уточняют статусы заявок. Однако подобная информация зачастую сохраняется вручную – в Excel-файлах, текстовых документах или мессенджерах. Это приводит к разрозненности данных, дублированию действий и затрудняет последующий анализ. По мере роста клиентской базы такие подходы теряют эффективность.

Анализ рынка показал, что большинство готовых решений ориентированы либо на массовые продажи, либо требуют сложной настройки и не учитывают специфику внутренних бизнес-процессов компании. Универсальные CRM-системы часто избыточны и неудобны в условиях ограниченного бюджета и ресурсов.

На этом фоне особенно актуальной становится разработка серверной части информационной системы, обеспечивающей централизованное хранение и обработку информации, безопасную аутентификацию пользователей, интеграцию с пользовательским интерфейсом и предоставление API для гибкого доступа к данным. Такая архитектура позволяет упорядочить взаимодействие с клиентами, повысить надежность процессов и обеспечить масштабируемость системы по мере роста компании.

Методы и материалы

Проектирование серверной части информационной системы началось с анализа бизнес-процессов и сценариев взаимодействия между сотрудниками компании и клиентами. На основе этих сценариев была построена диаграмма прецедентов, отражающая основные функции, реализуемые сервером: аутентификация пользователей, предоставление списка клиентов, получение и обновление истории взаимодействий, формирование аналитических данных. Данная диаграмма позволила определить границы ответственности серверной части и точки входа API, необходимые для взаимодействия с клиентским интерфейсом (рис. 1) [1].

На следующем этапе была разработана диаграмма последовательности, демонстрирующая пошаговое взаимодействие между компонентами при выполнении одного из ключевых сценариев – добавлении комментария к взаимодействию с клиентом. Диаграмма отражает процесс от ввода данных пользователем до их обработки на сервере и сохранения в базе данных. Особое внимание уделено этапам валидации, проверке авторизации с использованием JWT и выполнению SQL-запросов. Это позволило формализовать поведение серверной части при обращении к API, обеспечить корректную обработку запросов и уточнить структуру взаимодействия с клиентской частью (рис. 2).

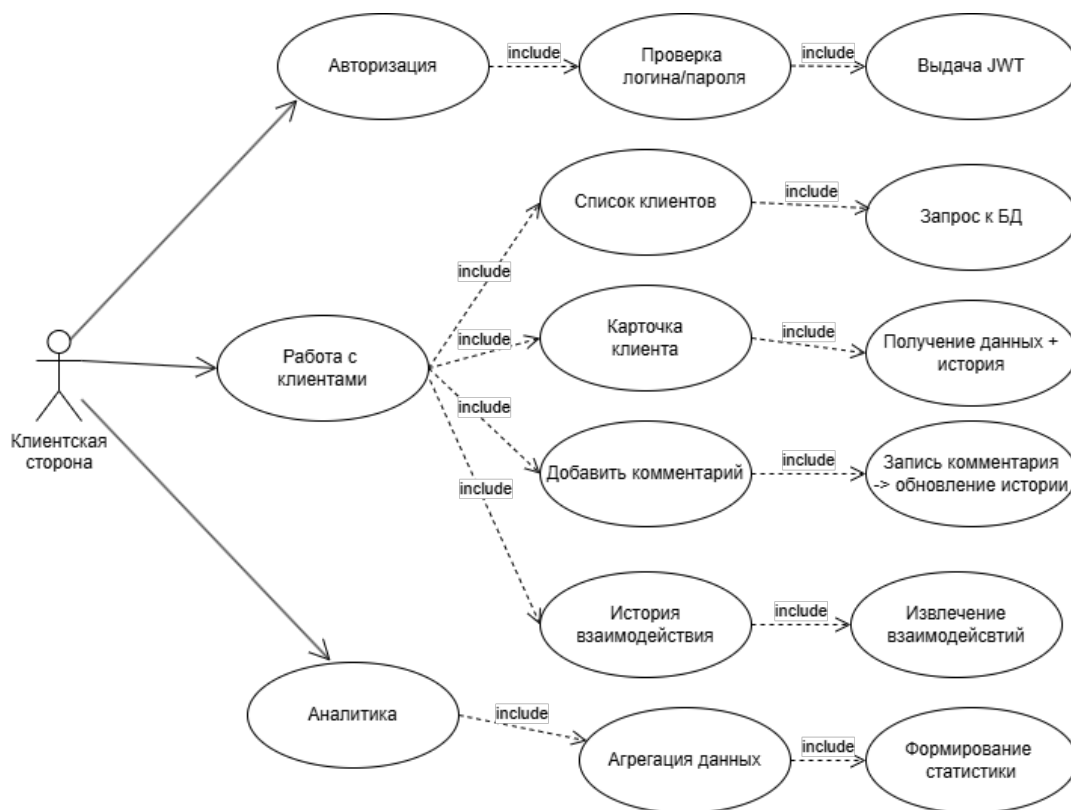


Рис. 1. Диаграмма прецедентов

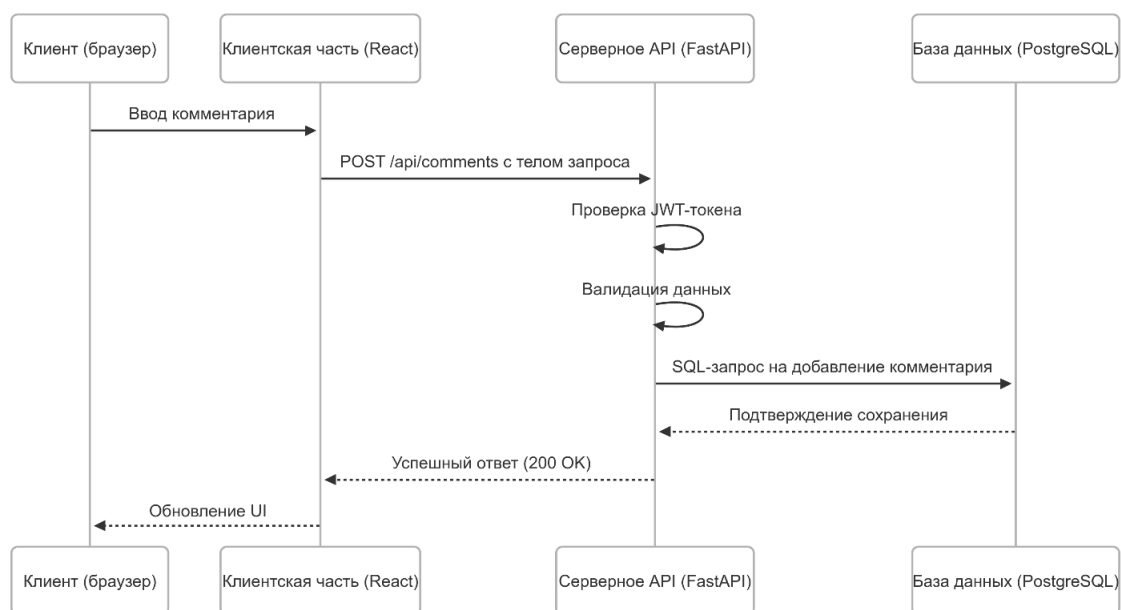


Рис. 2. Диаграмма последовательности

После логического моделирования предметной области была спроектирована структура базы данных, отражающая ключевые сущности системы и связи между ними. Этот этап обеспечил фундамент для надежного хранения, целостности и

структурированной обработки информации о взаимодействиях с клиентами. Проектирование осуществлялось в инструменте DrawSQL, что позволило визуализировать модель и заранее продумать архитектуру хранения данных (рис. 3).

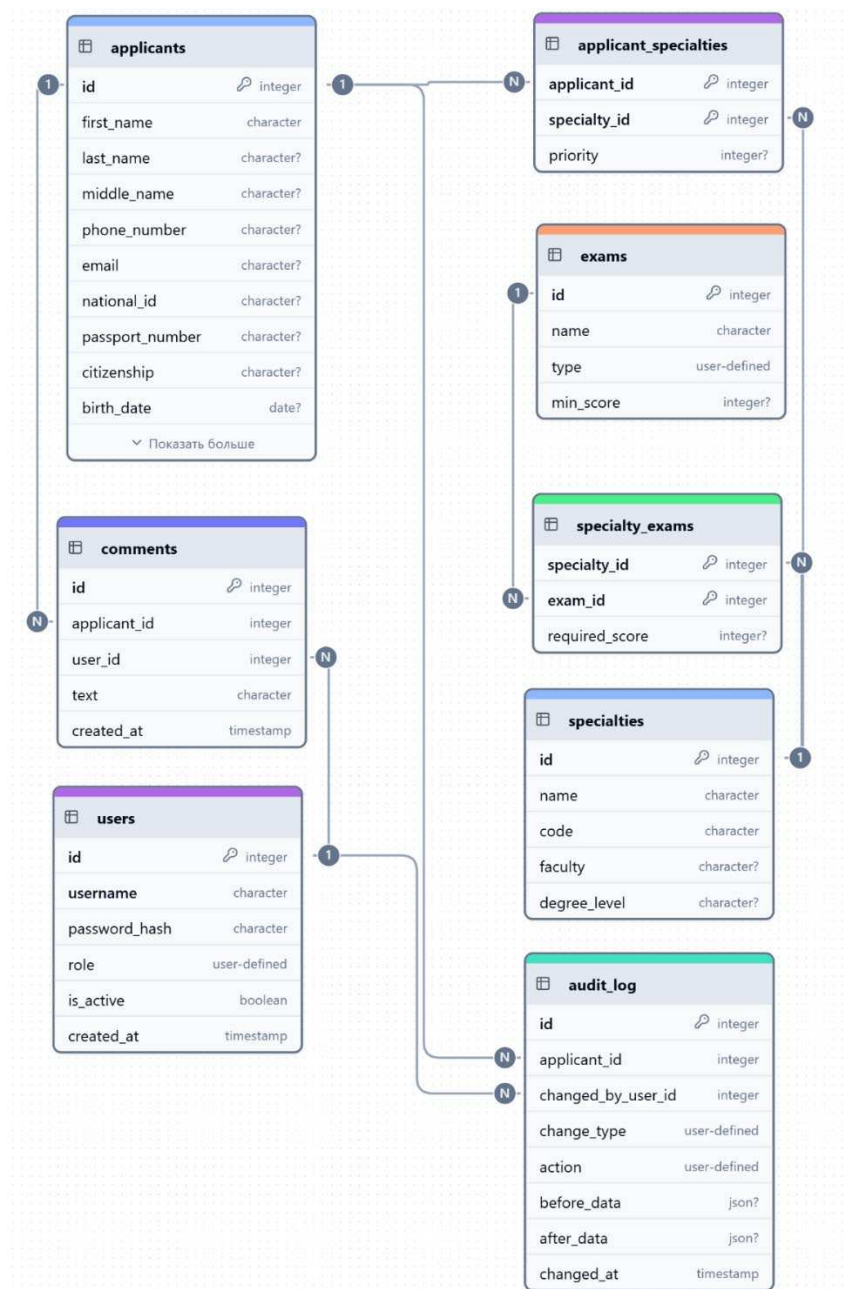


Рис. 3. Структура базы данных

Разработка серверной части информационной системы велась с использованием фреймворка FastAPI, который предоставляет современный и высокопроизводительный подход к созданию веб-приложений. Основное внимание при реализации backend-части было уделено модульности, читаемости кода, а также обеспечению масштабируемости и расширяемости архитектуры (рис. 4) [2].

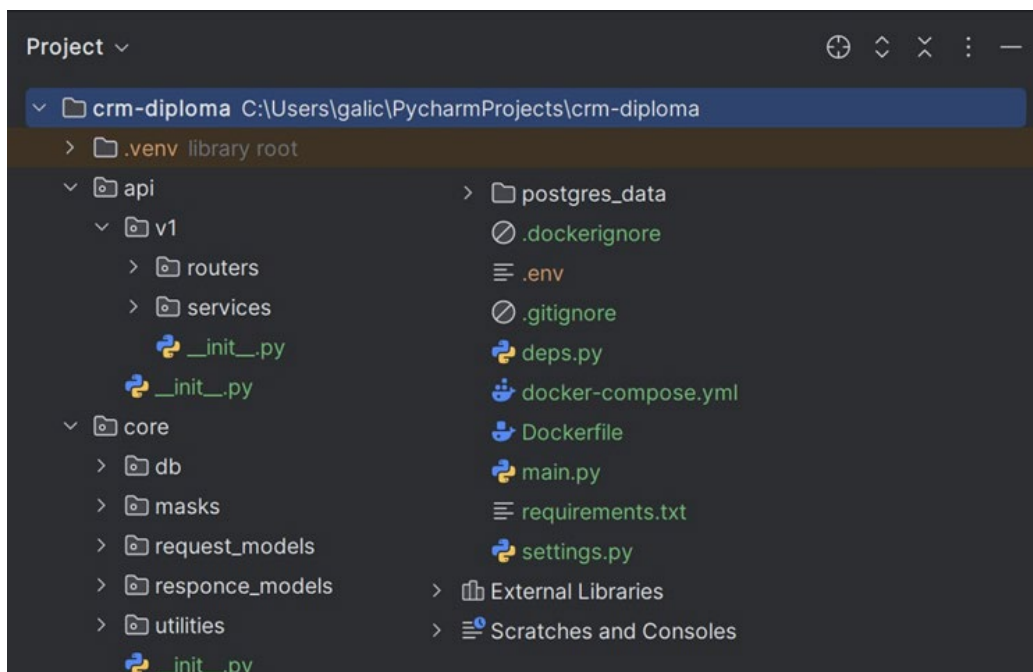


Рис. 4. Структура проекта

Для реализации серверной части информационной системы использовались следующие технологии и инструменты:

- FastAPI – асинхронный веб-фреймворк для создания REST API с поддержкой декларативного описания маршрутов, типизации и автоматической генерации документации по спецификации OpenAPI;
- PostgreSQL – реляционная СУБД, обеспечивающая надежное хранение и обработку данных, поддержку транзакций и сложных связей между таблицами;
- SQLAlchemy – ORM-библиотека для описания моделей и бизнес-логики в виде Python-классов с возможностью миграции и валидации схем базы данных;
- JWT (JSON Web Token) – механизм аутентификации, реализующий безопасную идентификацию пользователей и управление сессиями без сохранения состояния на сервере;
- Docker – инструмент контейнеризации, применяемый для упаковки серверного приложения вместе со всеми зависимостями и конфигурациями, что обеспечивает воспроизводимость и простоту развертывания.

Используемый стек технологий обеспечивает масштабируемость, отказоустойчивость и безопасность, а также упрощает сопровождение и перенос решения в продуктивную среду [3–6].

Результаты

В результате работы было разработано и протестировано серверное программное обеспечение, обеспечивающее надежную и безопасную обработку запросов в рамках информационной системы для поддержки взаимодействия с

клиентами компании. Серверная часть реализует REST API для основных функций: авторизации, управления данными клиентов, фиксации комментариев, работы со справочниками и получения аналитической информации.

Разработка велась на основе фреймворка FastAPI с использованием асинхронного программирования и типизированных схем. Для хранения данных применялась СУБД PostgreSQL, а взаимодействие с ней осуществлялось через ORM SQLAlchemy. Обеспечение безопасности доступа реализовано через JWT-аутентификацию.

Для оценки корректности работы серверного приложения было проведено ручное тестирование ключевых API-эндпоинтов. Тесты охватывали как стандартные сценарии (авторизация, CRUD-операции), так и проверку валидации, ошибок доступа и устойчивости к нагрузке [7]. Результаты тестирования серверной части приведены в табл. 1.

Таблица 1

Результаты тестирования серверной части

№	Тестируемый компонент	Описание теста	Ожидаемый результат	Статус
1	Авторизация	Вход с валидными данными	JWT-токен в ответе	Пройден
2	Авторизация	Вход с неверным паролем	Ошибка 401	Пройден
3	Получение списка клиентов	GET-запрос к /clients/	Список клиентов	Пройден
4	Добавление клиента	POST-запрос с корректными данными	Код 201, объект в ответе	Пройден
5	Добавление клиента	POST-запрос без обязательных полей	Ошибка 422	Пройден
6	Удаление клиента	DELETE-запрос с валидным ID	Код 204	Пройден
7	Добавление комментария	POST-запрос к /comments/	Код 201, объект в ответе	Пройден
8	Доступ к защищенному эндпоинту	GET-запрос без JWT	Ошибка 401	Пройден
9	Работа при высокой нагрузке	Множественные параллельные запросы	Устойчивый отклик сервера	Пройден

Заключение

Разработка серверной части информационной системы взаимодействия с клиентами компании позволила создать надежную, масштабируемую и безопасную архитектуру, ориентированную на автоматизацию ключевых процессов коммуникации. Реализованный REST API обеспечивает централизованное хранение данных, ведение истории взаимодействий, разграничение прав доступа и интеграцию с клиентской частью. Внедрение серверной логики устранило дублирование

информации, повысило согласованность работы сотрудников и обеспечило аналитическую основу для принятия управленческих решений.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Баланов, А. Бэкенд-разработка веб-приложений. Архитектура, проектирование и управление проектами. Учебное пособие. — М.: Лабиринт, 2022. — 320 с.
2. FastAPI Documentation. — Режим доступа: <https://fastapi.tiangolo.com/>, свободный. — Дата обращения: 01.05.2025.
3. PostgreSQL Documentation. — Режим доступа: <https://www.postgresql.org/docs/>, свободный. — Дата обращения: 01.05.2025.
4. SQLAlchemy Documentation. — Режим доступа: <https://docs.sqlalchemy.org/>, свободный. — Дата обращения: 01.05.2025.
5. Alembic Documentation. — Режим доступа: <https://alembic.sqlalchemy.org/>, свободный. — Дата обращения: 01.05.2025.
6. Docker Documentation. — Режим доступа: <https://docs.docker.com/>, свободный. — Дата обращения: 01.05.2025.
7. Сейбольд, К. PostgreSQL: эффективная работа с базами данных. — СПб.: БХВ-Петербург, 2021. — 320 с.

© Я. В. Галицын, П. Ю. Бугаков, 2025