САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бекенд разработка

Отчет

Домашняя работа 1

Выполнил:

Фадеев Д.А.

К3439

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

Текст задачи на лабораторную/практическую работу: спроектировать и описать набор диаграмм для микросервисной системы:

— общая архитектура решения (сервисы, взаимодействия и клиент-серверный контур);

— диаграмма компонентов;

— диаграммы баз данных по каждому сервису;

— диаграммы основных пользовательских сценариев, позволяющих пройти полный путь работы с приложением.

Ход Работы

Общее архитектурное решение

Общее архитектурное решение показано на рисунке 1:

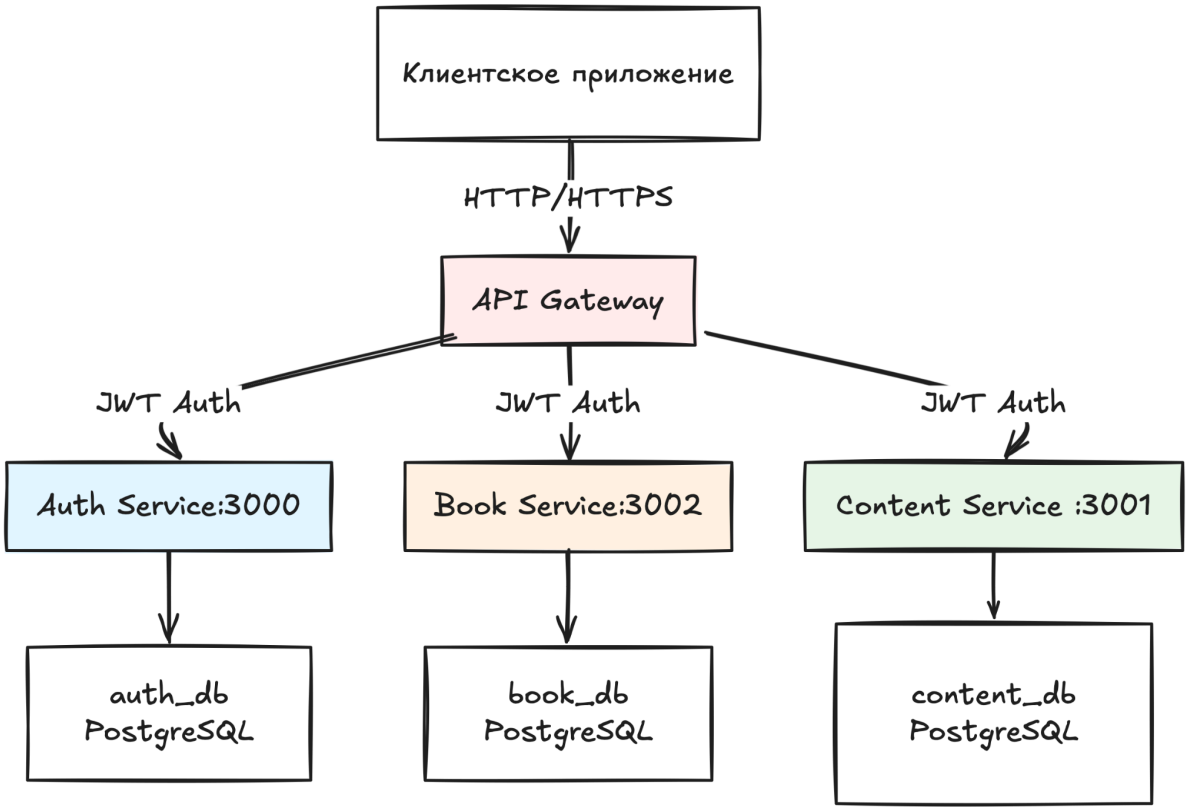

Рисунок 1 – Диаграмма общего архитектурного решения

Диаграмма показывает трехуровневую архитектуру с клиентским слоем, API Gateway для единой точки входа, тремя микросервисами с изолированными базами данных и RabbitMQ для асинхронного взаимодействия. Все компоненты оркестрируются через Docker Compose, что обеспечивает изоляцию и независимое масштабирование сервисов.

Диаграмма компонентов

Диаграмма компонентов показано на рисунке 1:

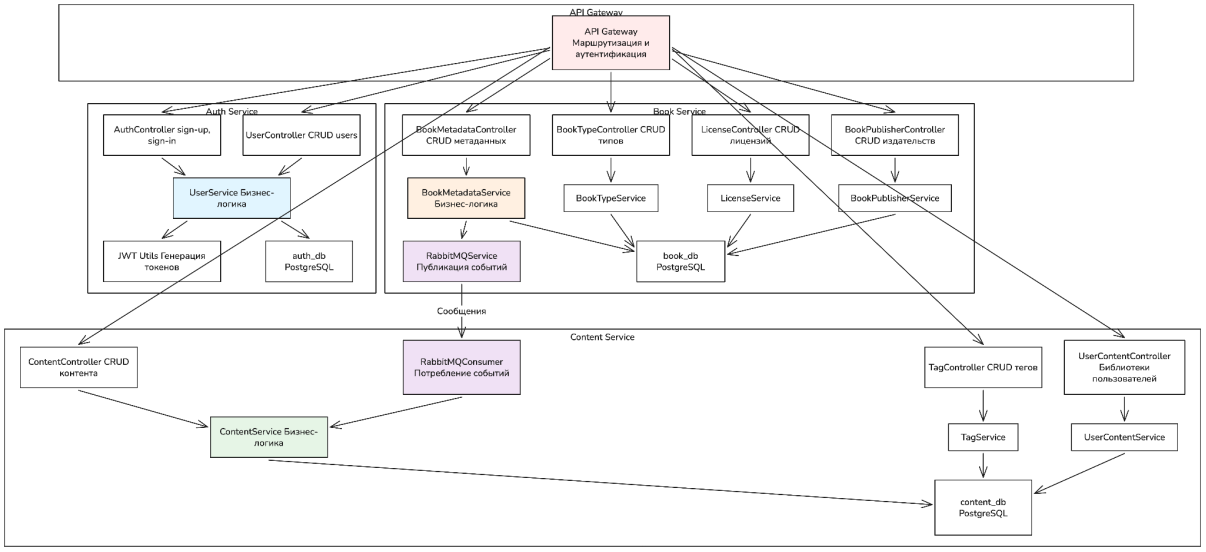

Рисунок 2 – Диаграмма компонент

Диаграмма демонстрирует внутреннюю структуру каждого микросервиса с разделением на контроллеры (обработка HTTP-запросов), сервисы (бизнес-логика) и репозитории (работа с БД). Общие компоненты (JWT middleware, Swagger, обработчики ошибок) используются всеми сервисами для обеспечения единообразия аутентификации, документации и обработки исключений.

Базы данных

Базы данных представлены на рисунках 3-5.

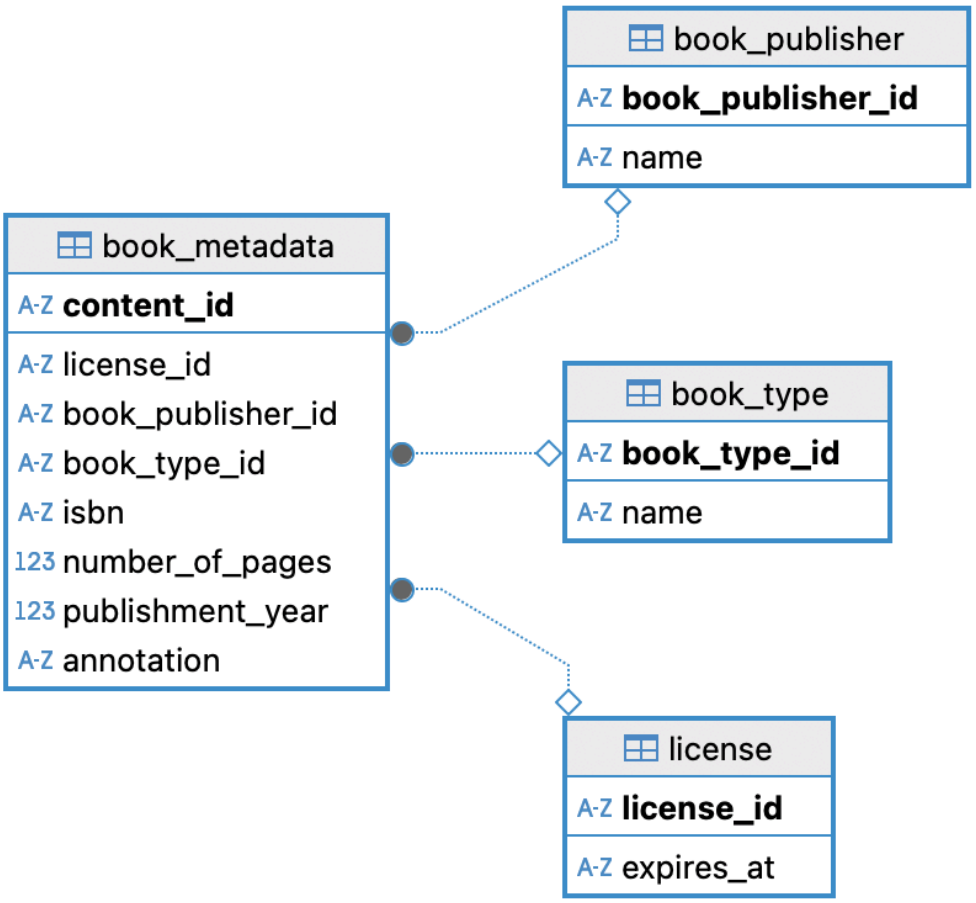

Рисунок 3 – Книги

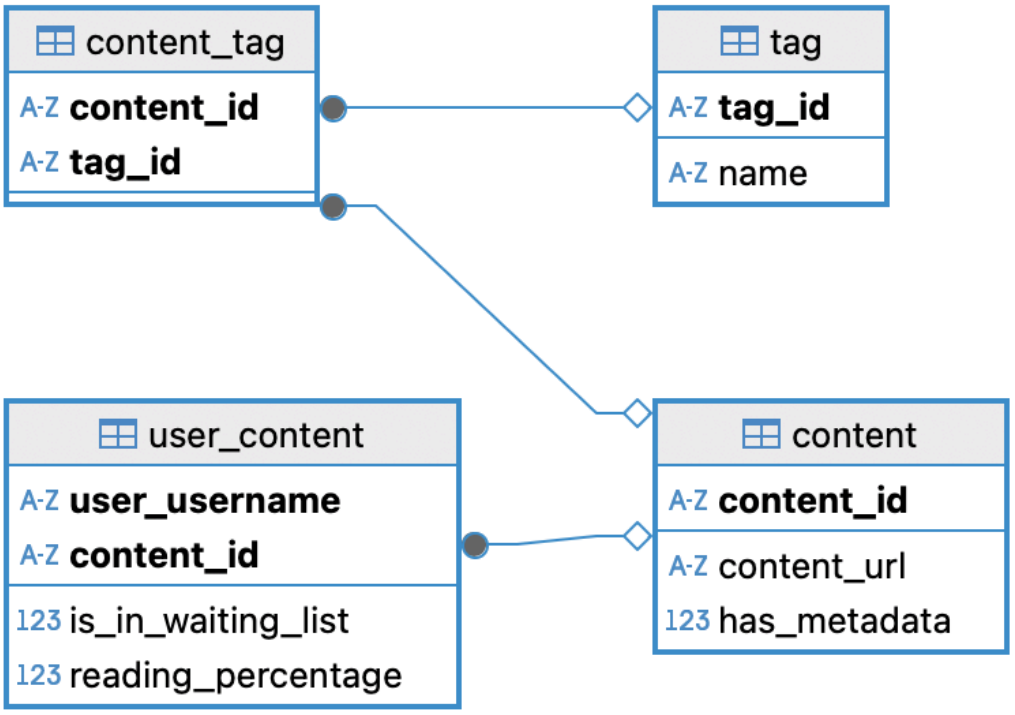

Рисунок 4 – Контент

Рисунок 5 – Auth Service

Пользовательские сценарии

Сценарий регистрации и авторизации пользователя показан на рисунке 6:

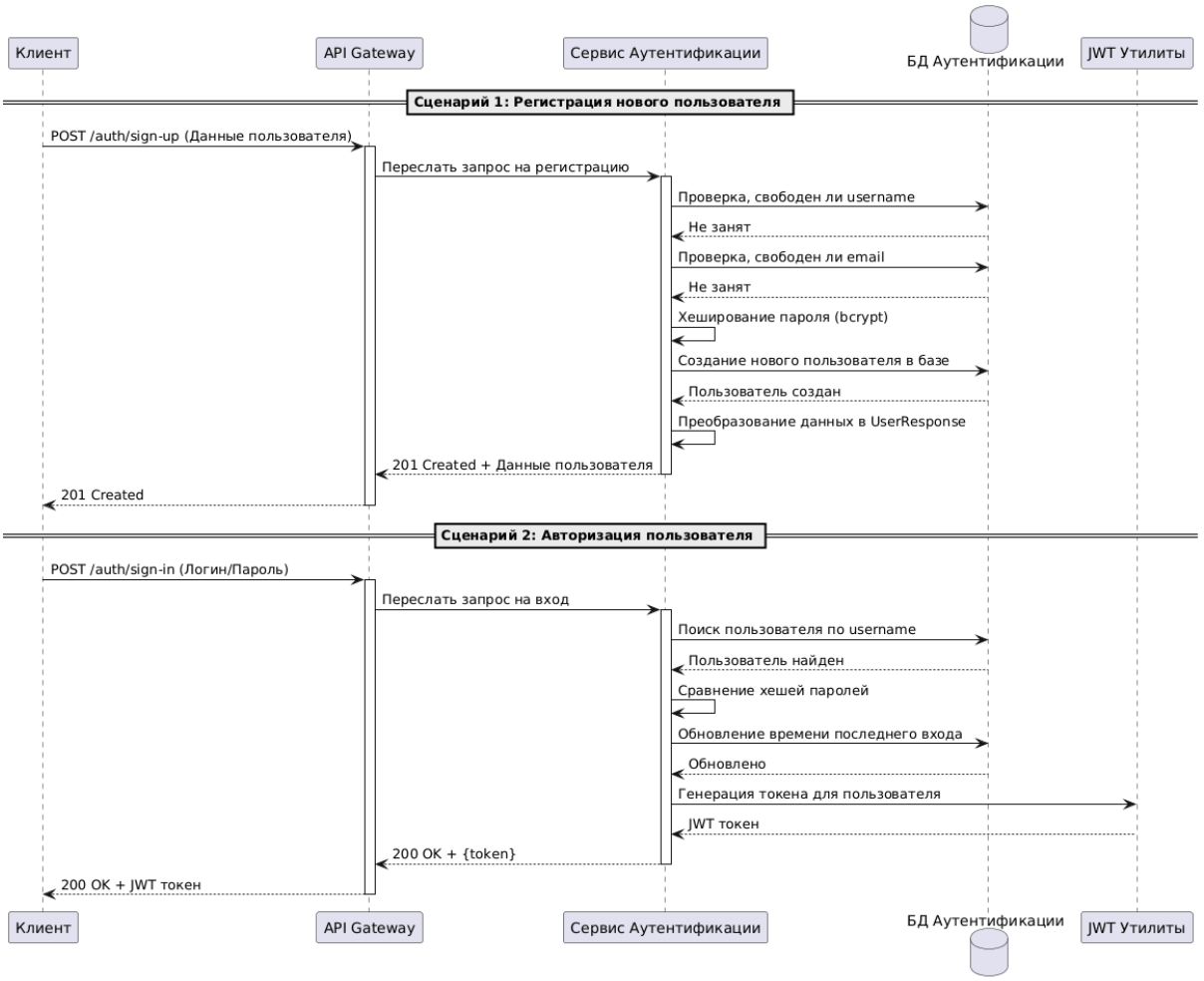

Рисунок 6 – Сценарий регистрации и авторизации пользователя

Диаграмма демонстрирует процесс создания метаданных книги администратором с валидацией связанных сущностей (BookType, License, BookPublisher) и последующей асинхронной синхронизацией через RabbitMQ. После сохранения метаданных в Book Service, событие публикуется в очередь, и Content Service асинхронно обновляет флаг has_metadata у соответствующего контента, обеспечивая слабую связанность между сервисами.

Сценарий добавление книги в библиотеку показан на рисунке 7:

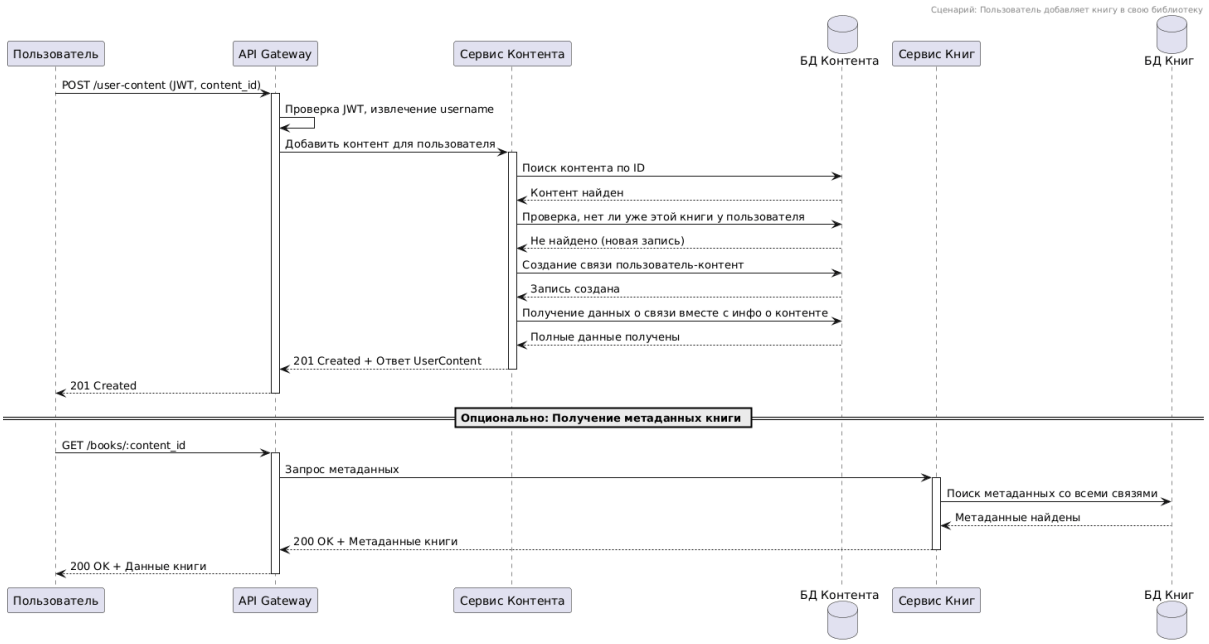

Рисунок 7 – Пользователь добавляет книгу в свою библиотеку

Диаграмма показывает процесс добавления контента в пользовательскую библиотеку с проверкой существования контента и созданием связи UserContent, а также опциональное получение метаданных книги из Book Service. Система позволяет пользователям управлять своей библиотекой контента и отслеживать прогресс чтения, при этом метаданные книг хранятся в отдельном сервисе для обеспечения разделения ответственности.

Сценарий создания метаданных книги показан на рисунке 8:

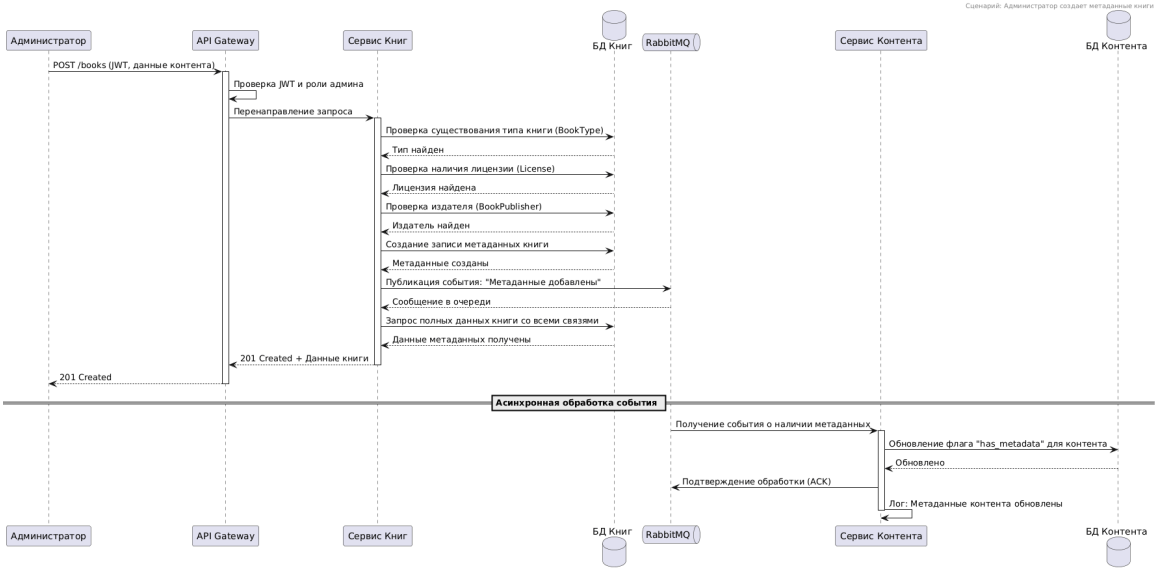

Рисунок 8 – Администратор создаёт метаданные книги

Диаграмма демонстрирует процесс создания метаданных книги администратором с валидацией связанных сущностей (BookType, License, BookPublisher) и последующей асинхронной синхронизацией через RabbitMQ. После сохранения метаданных в Book Service, событие публикуется в очередь, и Content Service асинхронно обновляет флаг has_metadata у соответствующего контента, обеспечивая слабую связанность между сервисами.

Вывод

В ходе работы спроектированы и описаны ключевые диаграммы микросервисного решения: архитектурная, компонентная, схемы БД по сервисам и диаграммы пользовательских сценариев. Полученная модель показывает разделение ответственности между сервисами, точки межсервисного взаимодействия и полный пользовательский путь от регистрации до отклика на вакансию и коммуникации.