ТЕХНИЧЕСКИЙ ПРОЕКТ

«Централизованный сервис для учёта книг библиотеки»

2023 г.

1. **Описание проекта**

Сервис предназначен для ведения учёта выдачи книг, отслеживания спроса на различные книги, хранения данных о читателях и книгах.

Целью выполнения проекта является разработка web-приложения, включающего в себя API на стороне сервера и desktop-приложение для работы с API.

На рисунках 1, 2 представлена диаграмма IDEF0, отображающая основные процессы ИС, и её декомпозиция.

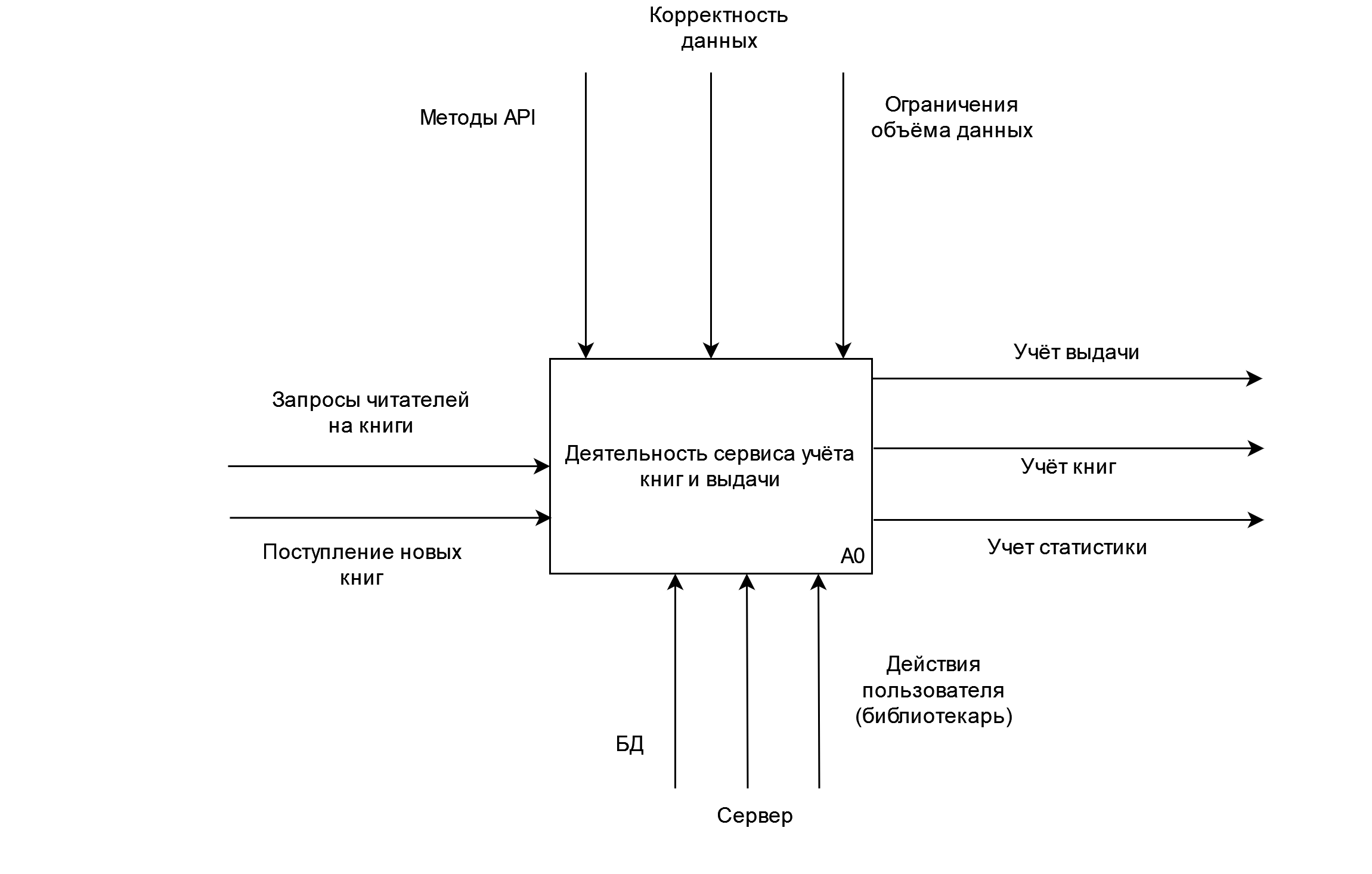


Рисунок 1 – Диаграмма IDEF0

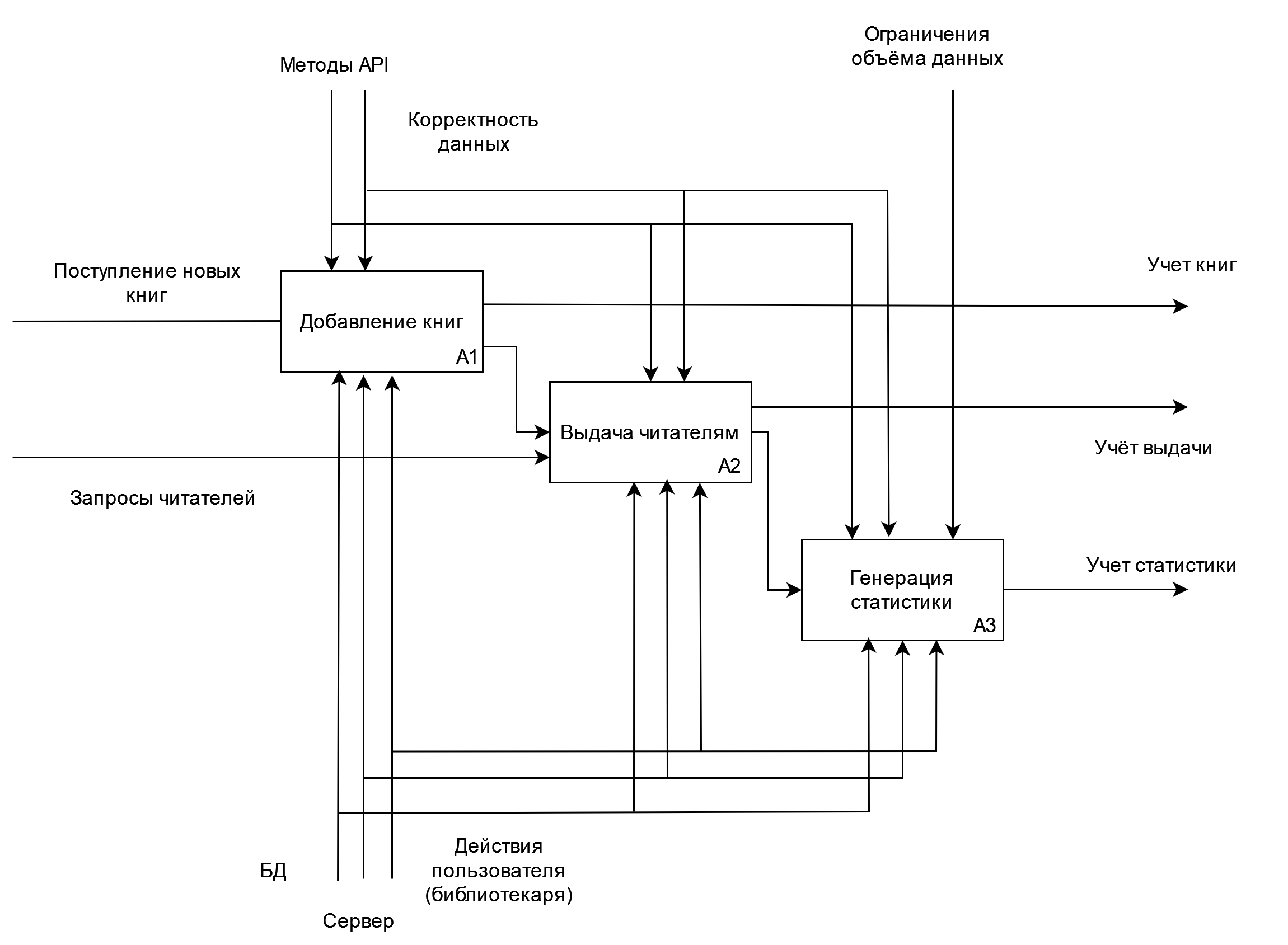


Рисунок 2 – Диаграмма декомпозиции IDEF0

1. **Уточнение структуры данных**

На рисунках 3, 4 представлены модели данных. Физическая модель создаётся в базе на основе ORM для Spring.



Рисунок 3 – Логическая модель данных

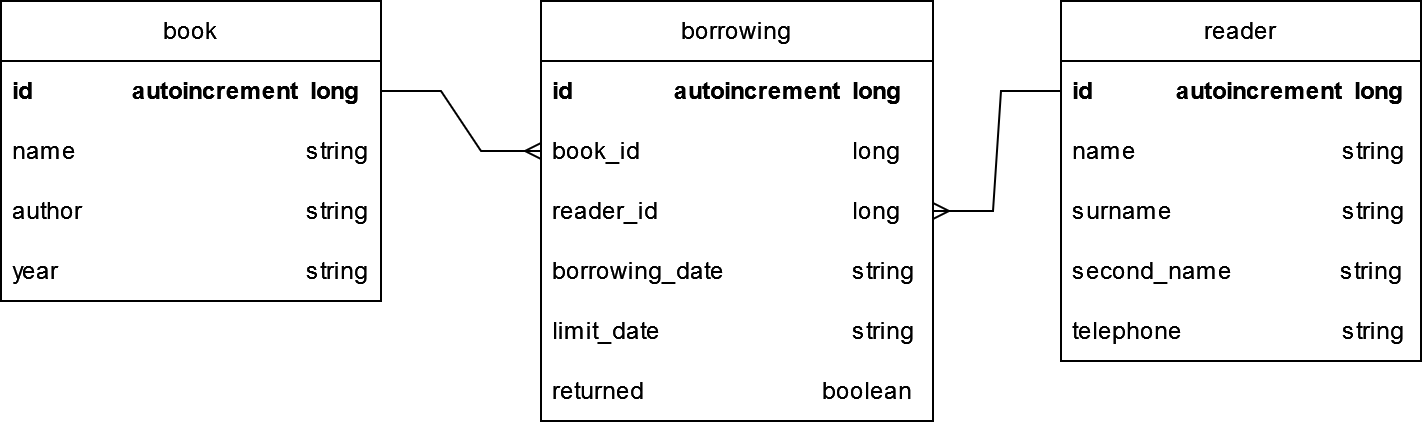


Рисунок 4 – Физическая модель данных

Хранение даты в строковом виде обеспечит большую универсальность и возможности для интеграции.

На рисунке 5 представлена DFD-диаграмма потоков данных.

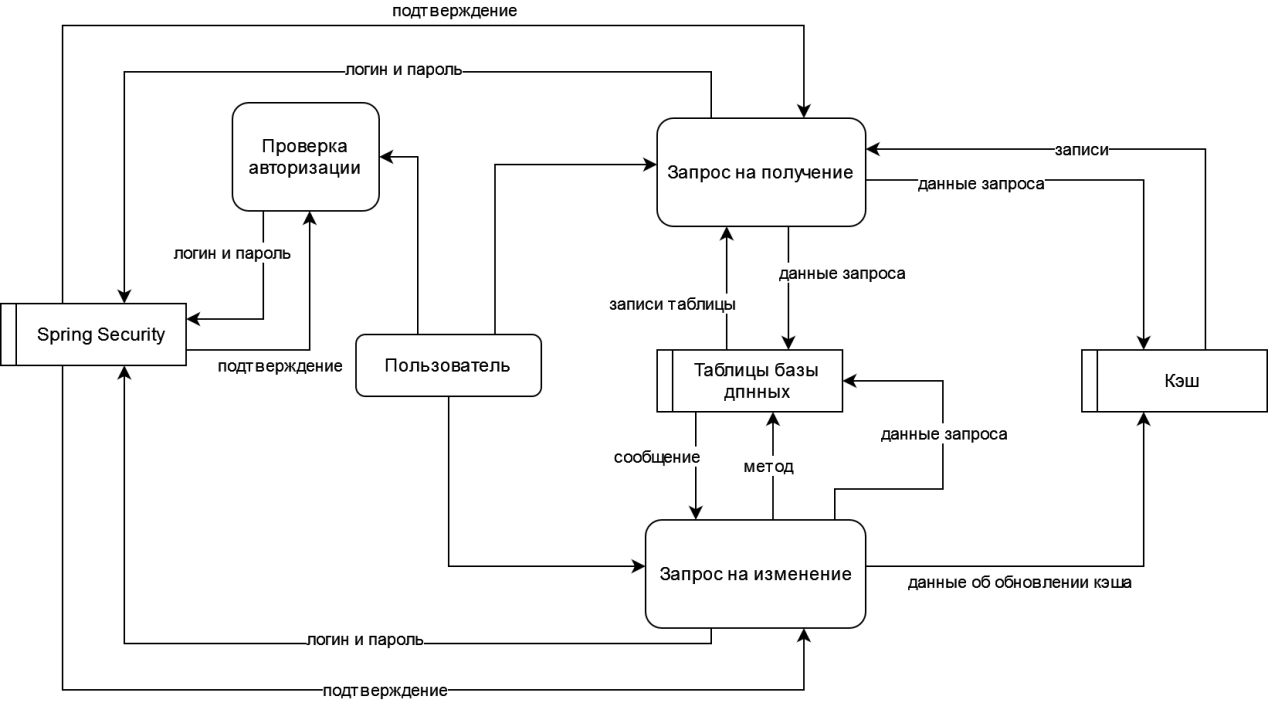


Рисунок 5 – Диаграмма потоков данных

Хранение данных в серверной части web-приложения осуществляется с помощью базы данных. Предусмотрено временное хранилище данных в оперативной памяти для кэша. Кэш должен поддерживать актуальное состояние, а также периодически сбрасываться. Период сброса кэша составляет 10 минут (для каждой записи время отсчитывается отдельно). Кэшированию подвергаются результаты выполнения методов только для таблиц книг, читателей и выдачи. Кэширование происходит по ключу, который соответствует значению параметра для функций (параметры запроса). Для страниц ключом будет являться номер страницы, а для одного значения - id.

В клиентском приложении постоянное хранилище данных не предусмотрено, все данные хранятся в оперативной памяти.

1. **Описание API разрабатываемого backend-приложения**

Для API должна быть создана документация в виде спецификации OpenAPI, представляющая собой Swagger-интрфейс для доступа к API. Таким образом остаётся возможность создания любого клиента для данного backend-сервиса.

Все запросы должны включать заголовок базовой авторизации.

API предоставляет следующие методы:

**Читатели:**

1. Получение страницы списка читателей

GET-запрос

Относительный URL: /reader

Параметр “page” - числовой номер страницы, > 0

Структура JSON ответа:

[

{

"id": integer,

"name": "string",

"surname": "string",

"secondName": "string",

"telephone": "string"

}

]

1. Обновление данных для читателя

PUT-запрос

Относительный URL: /reader

Структура JSON-тела запроса:

{

"id": integer,

"name": "string",

"surname": "string",

"secondName": "string",

"telephone": "string"

}

Структура JSON ответа (итоговые данные после обновления):

{

"id": integer,

"name": "string",

"surname": "string",

"secondName": "string",

"telephone": "string"

}

1. Добавление читателя

POST-запрос

Относительный URL: /reader

Структура JSON-тела запроса:

{

"name": "string",

"surname": "string",

"secondName": "string",

"telephone": "string"

}

Структура JSON ответа:

{

"id": integer,

"name": "string",

"surname": "string",

"secondName": "string",

"telephone": "string"

}

1. Получение данных читателя по id

GET-запрос

Относительный URL: /reader/{id}

Структура JSON ответа:

{

"id": integer,

"name": "string",

"surname": "string",

"secondName": "string",

"telephone": "string"

}

1. Удаление читателя по id

DELETE-запрос

Относительный URL: /reader/{id}

Ответ представляет собой строку-уведомление об удалении со значением id

**Книги:**

1. Получение страницы списка книг

GET-запрос

Относительный URL: /book

Параметр “page” - числовой номер страницы, > 0

Структура JSON ответа:

[

{

"id": integer,

"name": "string",

"author": "string",

"year": "string"

}

]

1. Обновление данных для книги

PUT-запрос

Относительный URL: /book

Структура JSON-тела запроса:

{

"id": integer,

"name": "string",

"author": "string",

"year": "string"

}

Структура JSON ответа (итоговые данные после обновления):

{

"id": integer,

"name": "string",

"author": "string",

"year": "string"

}

1. Добавление книги

POST-запрос

Относительный URL: /book

Структура JSON-тела запроса:

{

"name": "string",

"author": "string",

"year": "string"

}

Структура JSON ответа:

{

"id": integer,

"name": "string",

"author": "string",

"year": "string"

}

1. Получение данных книги по id

GET-запрос

Относительный URL: /book/{id}

Структура JSON ответа:

{

"id": integer,

"name": "string",

"author": "string",

"year": "string"

}

1. Удаление книги по id

DELETE-запрос

Относительный URL: /book/{id}

Ответ представляет собой строку-уведомление об удалении со значением id

**Выдача:**

1. Получение страницы списка выдачи

GET-запрос

Относительный URL: /borrowing

Параметр “page” - числовой номер страницы, > 0

Структура JSON ответа:

[

{

"id": integer,

"returned": true,

"borrowingDate": "string",

"limitDate": "string",

"bookId": integer,

"readerId": integer

}

]

1. Обновление данных для выдачи

PUT-запрос

Относительный URL: /borrowing

Структура JSON-тела запроса:

{

"id": integer,

"returned": true,

"borrowingDate": "string",

"limitDate": "string",

"bookId": integer,

"readerId": integer

}

Структура JSON ответа (итоговые данные после обновления):

{

"id": integer,

"returned": true,

"borrowingDate": "string",

"limitDate": "string",

"bookId": integer,

"readerId": integer

}

1. Добавление выдачи

POST-запрос

Относительный URL: /borrowing

Структура JSON-тела запроса:

{

"returned": true,

"borrowingDate": "string",

"limitDate": "string",

"bookId": integer,

"readerId": integer

}

Структура JSON ответа:

{

"id": integer,

"returned": true,

"borrowingDate": "string",

"limitDate": "string",

"bookId": integer,

"readerId": integer

}

1. Получение данных выдачи по id

GET-запрос

Относительный URL: /borrowing/{id}

Структура JSON ответа:

{

"id": integer,

"returned": true,

"borrowingDate": "string",

"limitDate": "string",

"bookId": integer,

"readerId": integer

}

1. Удаление выдачи по id

DELETE-запрос

Относительный URL: /borrowing/{id}

Ответ представляет собой строку-уведомление об удалении со значением id

**Статистика:**

1. Получение списка самых популярных книг

GET-запрос

Относительный URL: /statistic

Параметр “limit” - (число от 1 до 100) сколько записей выводить.

Структура JSON ответа:

[

"string"

]

Каждая строка (string) представляет собой перечисление следующий набор данных, разделённых запятой: номер книги, название, автор, год, количество выдач.

1. Получение списка самых популярных книг за период

GET-запрос

Относительный URL: /statistic/period

Параметр “limit” - (число от 1 до 100) сколько записей выводить. “date” - дата в формате “yyyy-MM-dd”.

Структура JSON ответа:

[

"string"

]

Каждая строка (string) представляет собой перечисление следующий набор данных, разделённых запятой: номер книги, название, автор, год, количество выдач.

**Выданные и невозвращенные книги:**

1. Получить сводку по невозвращенным книгам, срок которых вышел

GET-запрос

Относительный URL: /book-borrowing/over

Параметр “page” - числовой номер страницы, > 0

Структура JSON ответа:

[

{

"book\_id": integer,

"book\_name": "string",

"author": "string",

"borrowing\_id": integer,

"returned": true,

"borrowing\_date": "string",

"limit\_date": "string",

"reader\_id": integer,

"reader\_name": "string"

}

]

1. Получить сводку по невозвращенным книгам, срок возврата которых не подошёл

GET-запрос

Относительный URL: /book-borrowing/in

Параметр “page” - числовой номер страницы, > 0

Структура JSON ответа:

[

{

"book\_id": integer,

"book\_name": "string",

"author": "string",

"borrowing\_id": integer,

"returned": true,

"borrowing\_date": "string",

"limit\_date": "string",

"reader\_id": integer,

"reader\_name": "string"

}

]

1. Получить сводку по всей выдаче

GET-запрос

Относительный URL: /book-borrowing/all

Параметр “page” - числовой номер страницы, > 0

Структура JSON ответа:

[

{

"book\_id": 0,

"book\_name": "string",

"author": "string",

"borrowing\_id": 0,

"returned": true,

"borrowing\_date": "string",

"limit\_date": "string",

"reader\_id": 0,

"reader\_name": "string"

}

]

1. **Формы представления данных**

В клиентском приложении данные представлены в виде графического интерфейса, прототипы форм представлены на рисунках 6 - 9.

На рисунке 6 представлен прототип экранной формы основной рабочей области приложения. Форма «Выбранная запись таблицы» представляет собой набор полей ввода для атрибутов (атрибут id недоступен для изменения), и заполняется данными, когда была выбрана (нажата) строка в таблице. Кнопка «Обновить» обращается к серверу для получения актуальной информации для данной страницы таблицы, «Обновить все» получает информацию для всех таблиц основной рабочей области с учётом страниц. Снизу таблиц расположены кнопки для навигации по страницам, навигация возможна для страниц с номером больше нуля. На каждой странице отображается максимум 10 записей. В окне основной рабочей области (рисунок 6) есть кнопки перехода на страницы: самые популярные книги («Популярные книги»), сводка по выданным книгам («Сводка по выдаче»), выданные книги, срок возврата которых истёк («Срок вышел»), и выданные на текущий момент книги, срок возврата которых ещё не подошёл («Срок не подошёл»).

При запуске приложения появляется окно авторизации для ввода логина и пароля (рисунок 7).

Форма поиска позволяет ввести идентификатор и получить данные по этому идентификатору, отображая в таблице только найденную запись. Выпадающий список позволяет выбрать таблицу, для которой будет выполнен поиск. «Сброс» очищает поле поиска и восстанавливает текущую страницу для всех таблиц.

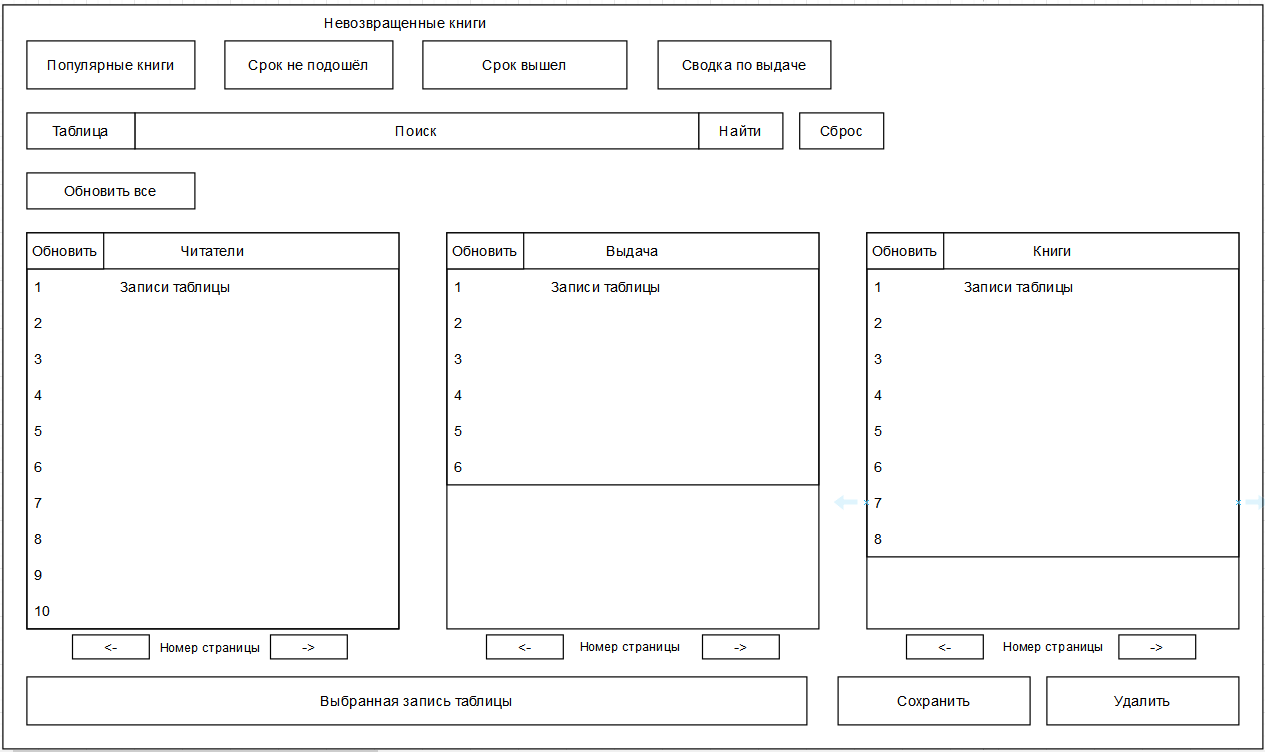


Рисунок 6 – Экранная форма основной рабочей области

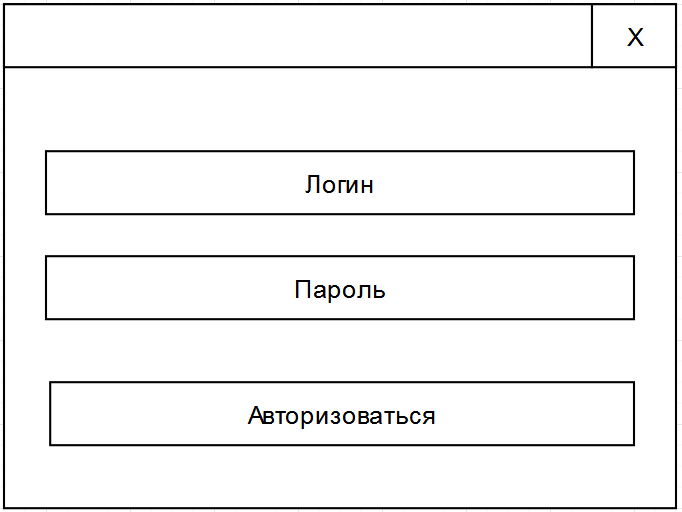


Рисунок 7 – Окно авторизации

На рисунке 8 представлена экранная форма для статистики: список самых популярных книг, включая количество выдачи для каждой книги. Количество отображаемых книг соответствует указанному значению в форме ввода для чисел. С помощью формы ввода даты можно указать период (дату начала для вычисления статистики). Для того чтобы период учитывался, необходимо поставить отметку в поле (checkbox), расположенном перед формой даты.

Предупреждения об ошибках выводятся в виде диалоговых окон, состоящих из сообщений о причине ошибки.

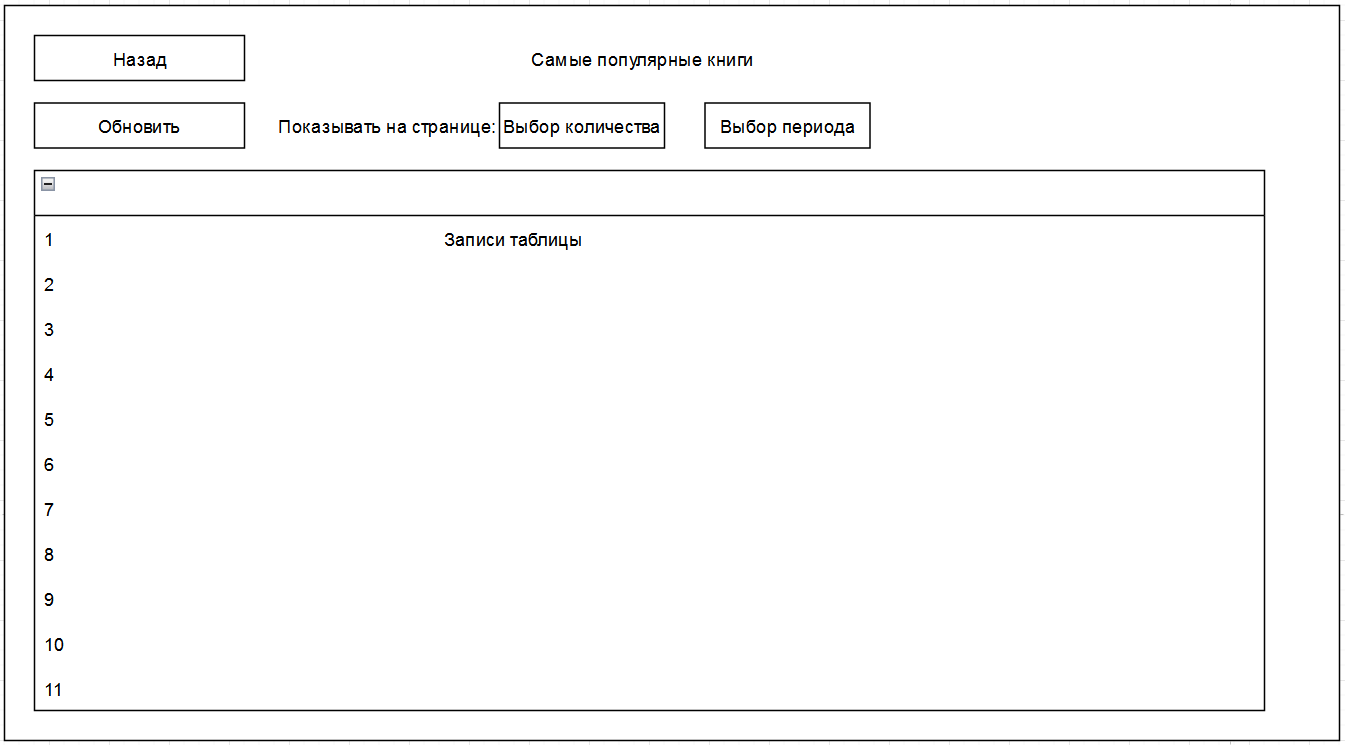


Рисунок 8 – Экранная форма для статистики

На рисунке 9 представлена экранная форма, элементы которой являются общими для страниц сводки по всем выданным книгам, книгам, срок возврата которых истёк и срок возврата которых ещё не подошёл.

В таблице отображается максимум 10 записей на страницу.

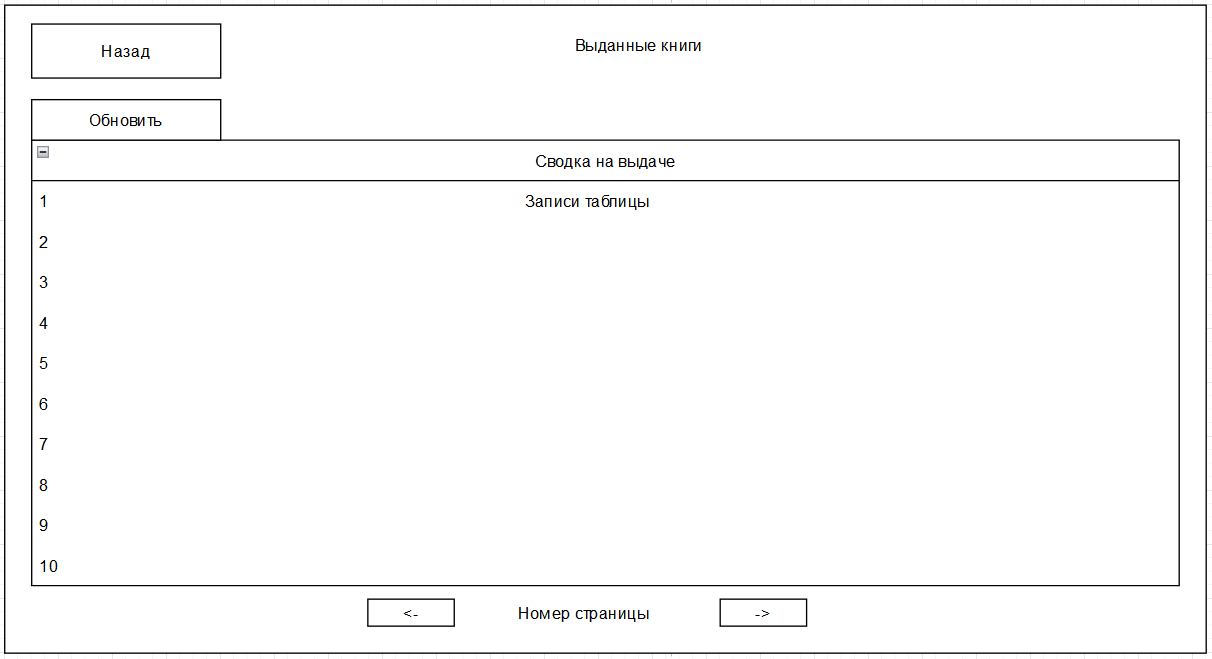


Рисунок 9 – Экранная форма для списка невозвращенных книг

1. **Требования к языкам, библиотекам и технологиям**

Серверная часть:

Для разработки используется Java 17 версии. Spring фреймворк, в частности: Spring Boot, Spring Cloud Netflix (микросервисы для создания балансироващика нагрузки и маршрутизатора), Spring Security (для реализации авторизации - базовой), СУБД PostgreSQL, Распределённый кэш Hazelcast.

Клиент: Python 3, PyQT 6 версии для создания интерфейсов и библиотека requests для работы с API.

1. **Архитектура системы**

Для реализации маршрутизатора запросов (и в том числе балансировщика нагрузки) используется микросервисная архитектура, которая содержит: «Service» - основной модуль с REST API (может быть запущен в нескольких экземплярах), «API Gateway» - маршрутизатор запросов и «Discovery service», который является в данном случаи служебным и выполняет функцию регистрации остальных сервисов. На рисунке 10 представлена схема микросервисов.

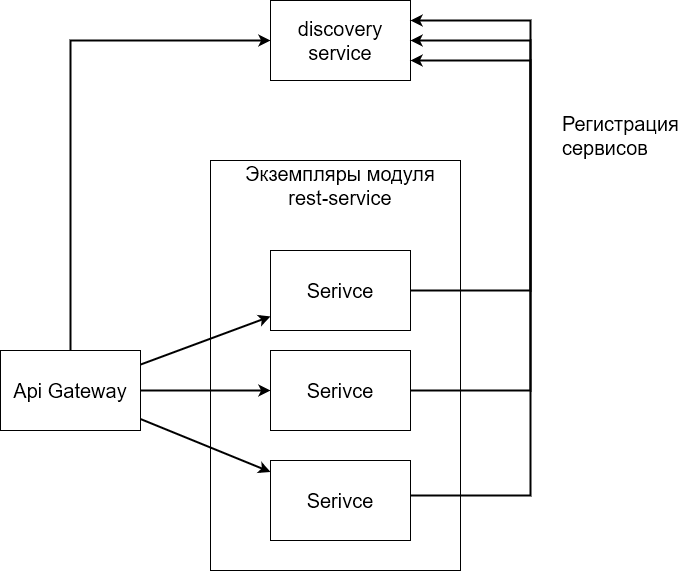


Рисунок 10 – Схема микросервисов

Распределённый кэш Hazelcast может работать с несколькими экземлярами приложения, схема - рисунок 11.

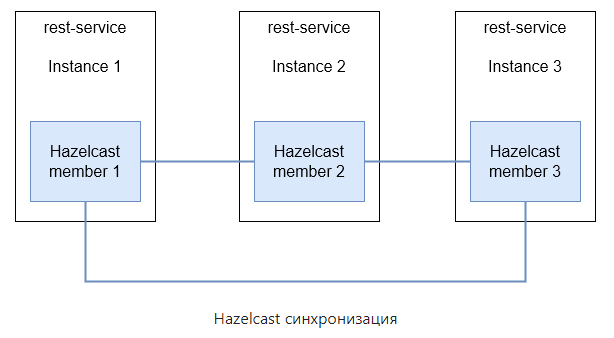


Рисунок 11 – Распределённый кэш

На рисунке 12 представлена схема обработки запроса клиента и отправки ему ответа.

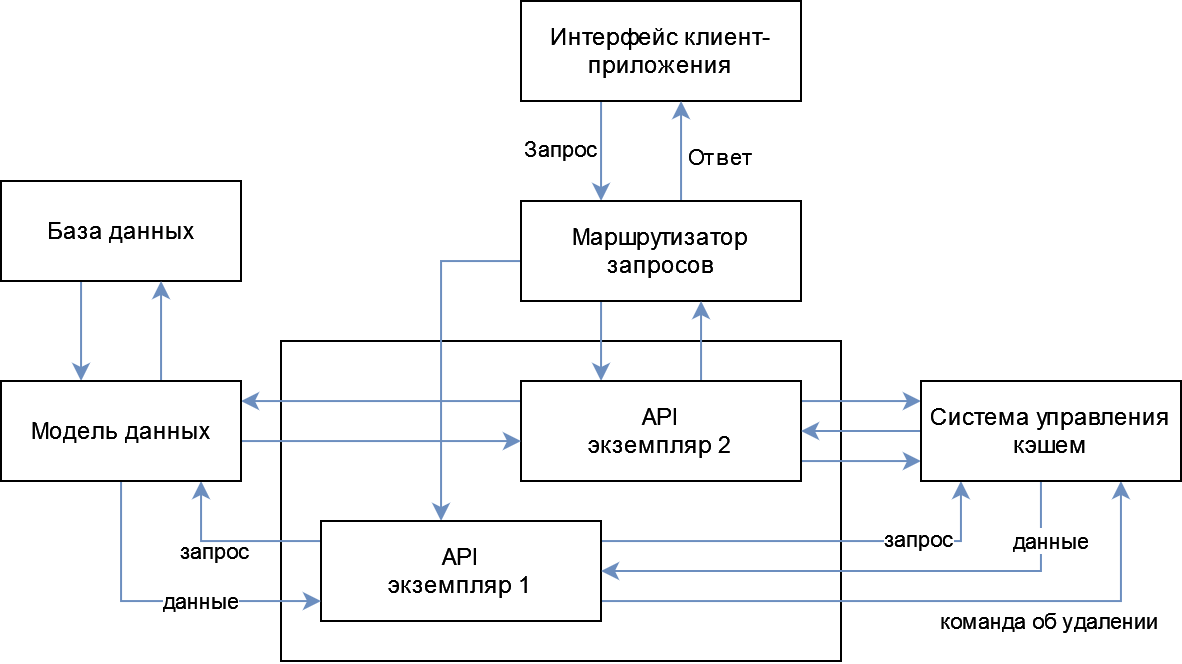


Рисунок 12 – Схема обработки запроса