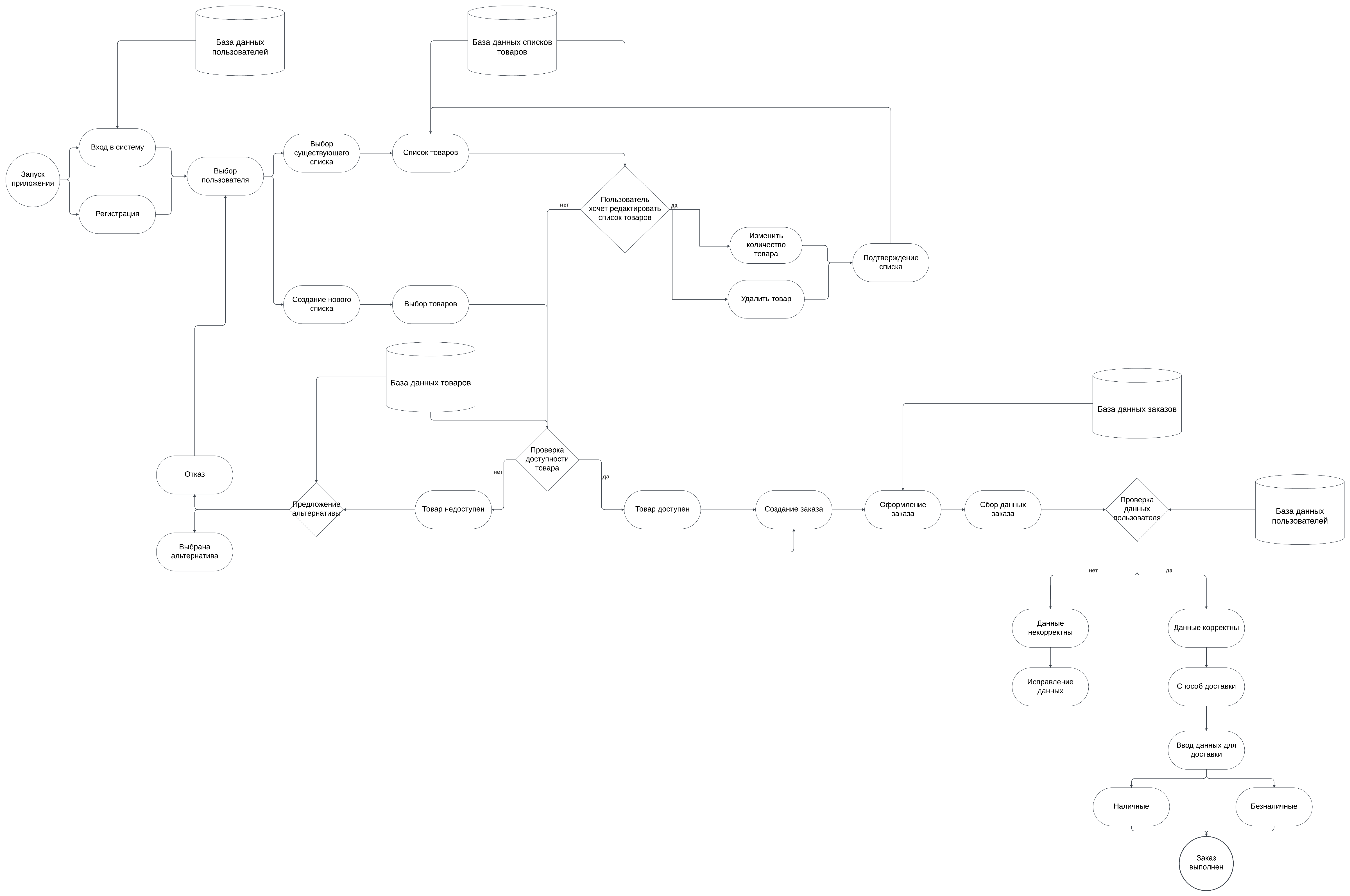
**Задание 1:**  
Бизнес-логика мобильного приложения

**Задание 2:**

Основные функции данного приложения -

1. **Создание списка покупок** – возможность быстро создать новый список или выбрать уже существующий.
2. **Добавление и редактирование товаров** – поиск, изменение количества, удаление, добавление заметок.
3. **Отмечать купленные товары** – удобно, чтобы не запутаться при покупках.
4. **Сохранение списков** – чтобы можно было повторно использовать их в будущем.
5. **Совместное использование списка** – можно отправить список другим людям.
6. **Примерный расчёт стоимости** – если указаны цены, приложение покажет итоговую сумму.
7. **Напоминания** – уведомления о незавершённых списках или регулярных покупках.
8. **Авторизация и личный кабинет** – сохранение данных пользователя и синхронизация списков.
9. **Оформление заказа (если реализовано)** – возможность заказать товары (если такая функция будет добавлена).

**Задание 3:**

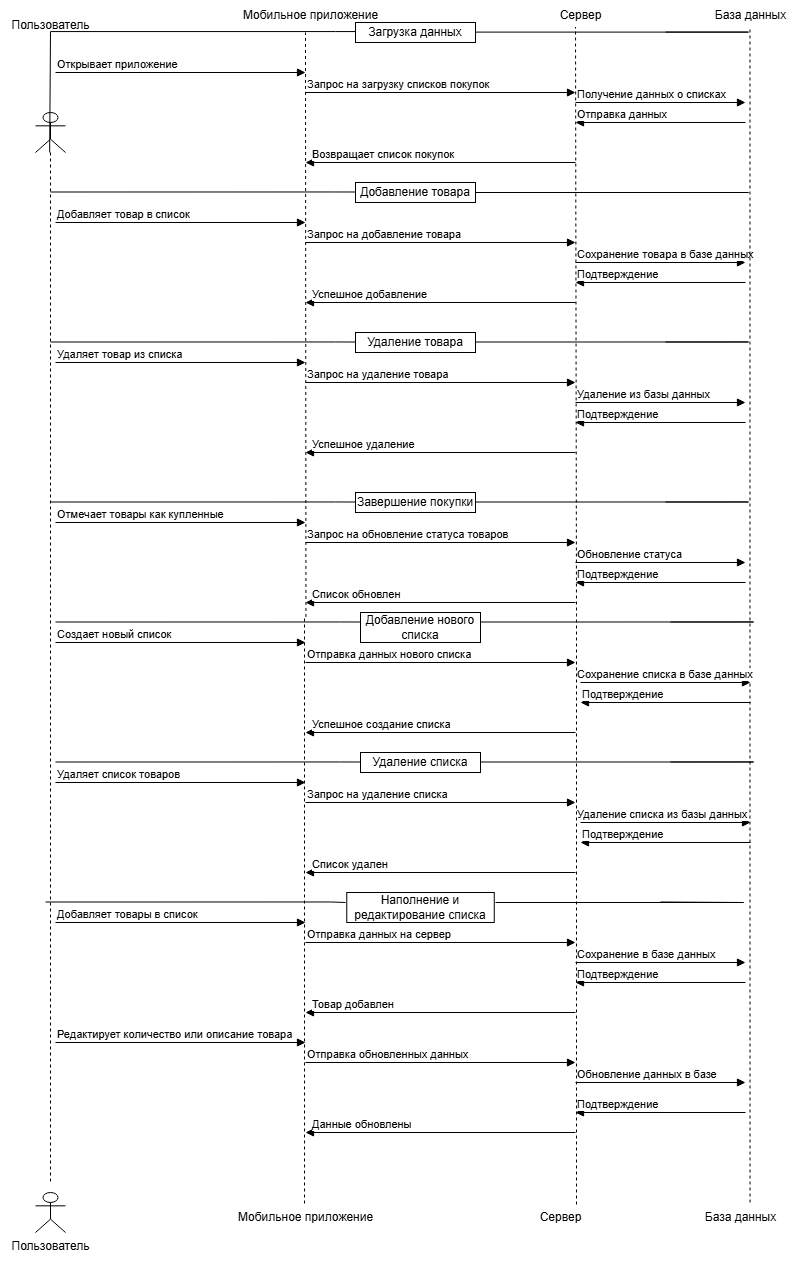
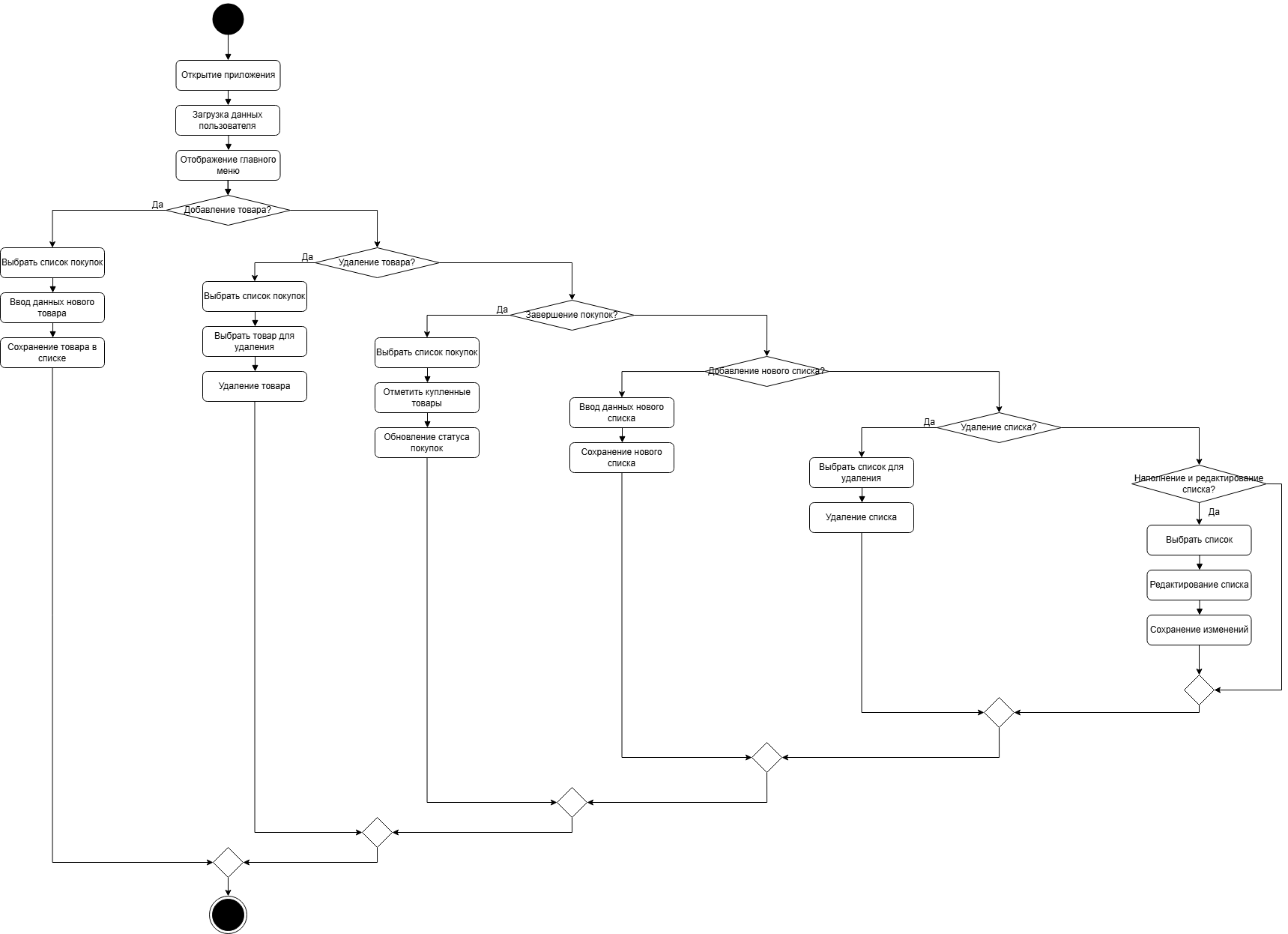
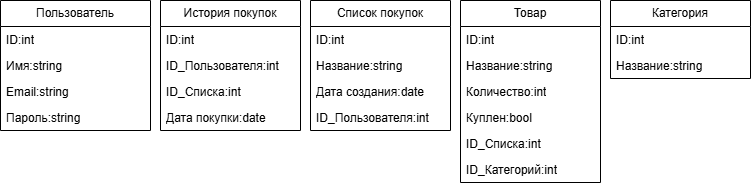


Диаграмма последовательностей

Диаграмма активностей

ER-Диаграмма

Примеры API методов

**POST /register** - Регистрация пользователя  
**POST /login** - Авторизация пользователя  
**GET /shopping-lists** - Получение списков пользовател

**POST /shopping-lists** - Создание нового списка

**DELETE /shopping-lists/3** - Удаление списка

**GET /shopping-lists/1/products** - Просмотр товаров в списке

**POST /shopping-lists/1/products** - Добавление товара в список

**PUT /shopping-lists/1/products/3** - Редактирование товара

**DELETE /shopping-lists/1/products/3** - Удаление товара из списка

**PUT /shopping-lists/1/products/3/purchase** - Отметить товар как купленный

**GET /purchase-history** - История покупок

**Задание 4:**

[**https://www.figma.com/design/tk4tbaX8nXDw0Tx2bAB5eQ/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF-%D0%BF%D0%BE%D0%BA%D1%83%D0%BF%D0%BA%D0%B0-%D1%82%D0%BE%D0%B2%D0%B0%D1%80%D0%BE%D0%B2-%D0%B2-%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B5?node-id=0-1&t=0ZgwQdnJNtfgufed-1**](https://www.figma.com/design/tk4tbaX8nXDw0Tx2bAB5eQ/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF-%D0%BF%D0%BE%D0%BA%D1%83%D0%BF%D0%BA%D0%B0-%D1%82%D0%BE%D0%B2%D0%B0%D1%80%D0%BE%D0%B2-%D0%B2-%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B5?node-id=0-1&t=0ZgwQdnJNtfgufed-1)



**Задание 5:**

**API:**

### **1. Получение списка покупок**

**Метод:**  
GET /shopping-lists/{list\_id}/items

**Описание:**  
Возвращает актуальный список товаров для определённого списка покупок.

* **list\_id (UUID)** – идентификатор списка покупок.

Ответ: JSON, содержащий массив объектов товаров.

### **2. Отметка товара как купленного**

**Метод:**  
PATCH /shopping-lists/{list\_id}/items/{item\_id}

**Описание:**  
Обновляет статус товара на «куплено» и фиксирует дату покупки.

Тело запроса:

{

"status": "purchased",

"purchase\_date": "2024-12-18T18:00:00Z"

}

**Ответ:**  
JSON с обновлённой информацией о товаре

3. Откупка товара

**Метод:**  
PATCH /shopping-lists/{list\_id}/items/{item\_id}/unpurchase

**Описание:**  
Обновляет статус товара на «не куплено» (отменяет отметку о покупке).

**Тело запроса:**

{

"status": "pending",

"purchase\_date": null

}

**Ответ:**  
JSON с обновлённой информацией о товаре

### **4. Получение истории покупок**

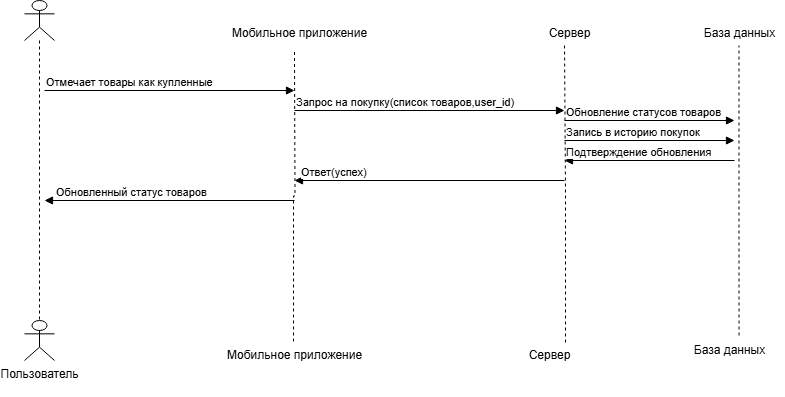
**Метод:**  
GET /purchase-history

**Описание:**  
Возвращает историю покупок для определённого пользователя.

**Параметры запроса:**

* user\_id (UUID) – идентификатор пользователя.

**Ответ:**  
JSON с массивом записей истории покупок. Каждая запись может содержать информацию о списке покупок, дате покупки, итоговой сумме и деталях по каждому товару.

Диаграмма последовательности

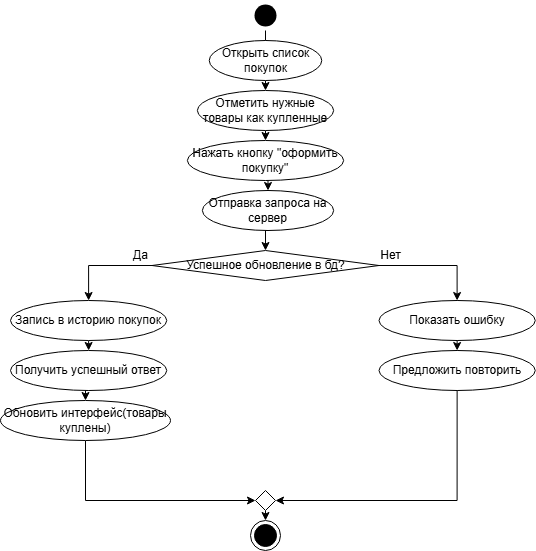


Диаграмма активностей

**Задание 6:**

**Основные сложности разработки:**

**1. Синхронизация данных между клиентом и сервером:**

* Обеспечение актуальности данных при одновременной работе с разных устройств.
* Обработка конфликтов при обновлении одного и того же списка или товара.

**2. Интуитивно понятный пользовательский интерфейс:**

* Разработка простого, понятного и удобного дизайна для быстрой работы с списками.
* Поддержка адаптивного интерфейса для различных устройств (смартфоны, планшеты).

**Вопросы заказчику:**

* Требуется ли реализовывать возможность совместного редактирования списка несколькими пользователями? Если да, то какие права доступа и уровни разрешений требуются?
* Какие требования по безопасности данных существуют? Нужно ли использовать дополнительные меры (например, двухфакторную аутентификацию)?
* На каких платформах должно работать приложение (iOS, Android, кроссплатформенные решения)?
* Какое количество пользователей ожидается в начальном этапе и в перспективе? Есть ли требования по максимальному числу одновременных запросов?

**Задание 7:**

**1. Общая стоимость книг для каждого автора и отсортировать результат в порядке убывания;**

SELECT a.AuthorName,

SUM(b.Price) AS TotalPrice

FROM Authors a

JOIN Books b ON a.Id = b.AuthorId

GROUP BY a.AuthorName

ORDER BY TotalPrice DESC;

**2. Стоимость книг автора превышает 1500**

SELECT a.AuthorName,

SUM(b.Price) AS TotalPrice

FROM Authors a

JOIN Books b ON a.Id = b.AuthorId

GROUP BY a.AuthorName

HAVING SUM(b.Price) > 1500;

**3. Вывести авторов с количеством книг**

SELECT a.AuthorName,

COUNT(b.Id) AS BookCount

FROM Authors a

LEFT JOIN Books b ON a.Id = b.AuthorId

GROUP BY a.AuthorName;

**4. Получить автора без книг**

SELECT a.AuthorName

FROM Authors a

LEFT JOIN Books b ON a.Id = b.AuthorId

WHERE b.Id IS NULL;