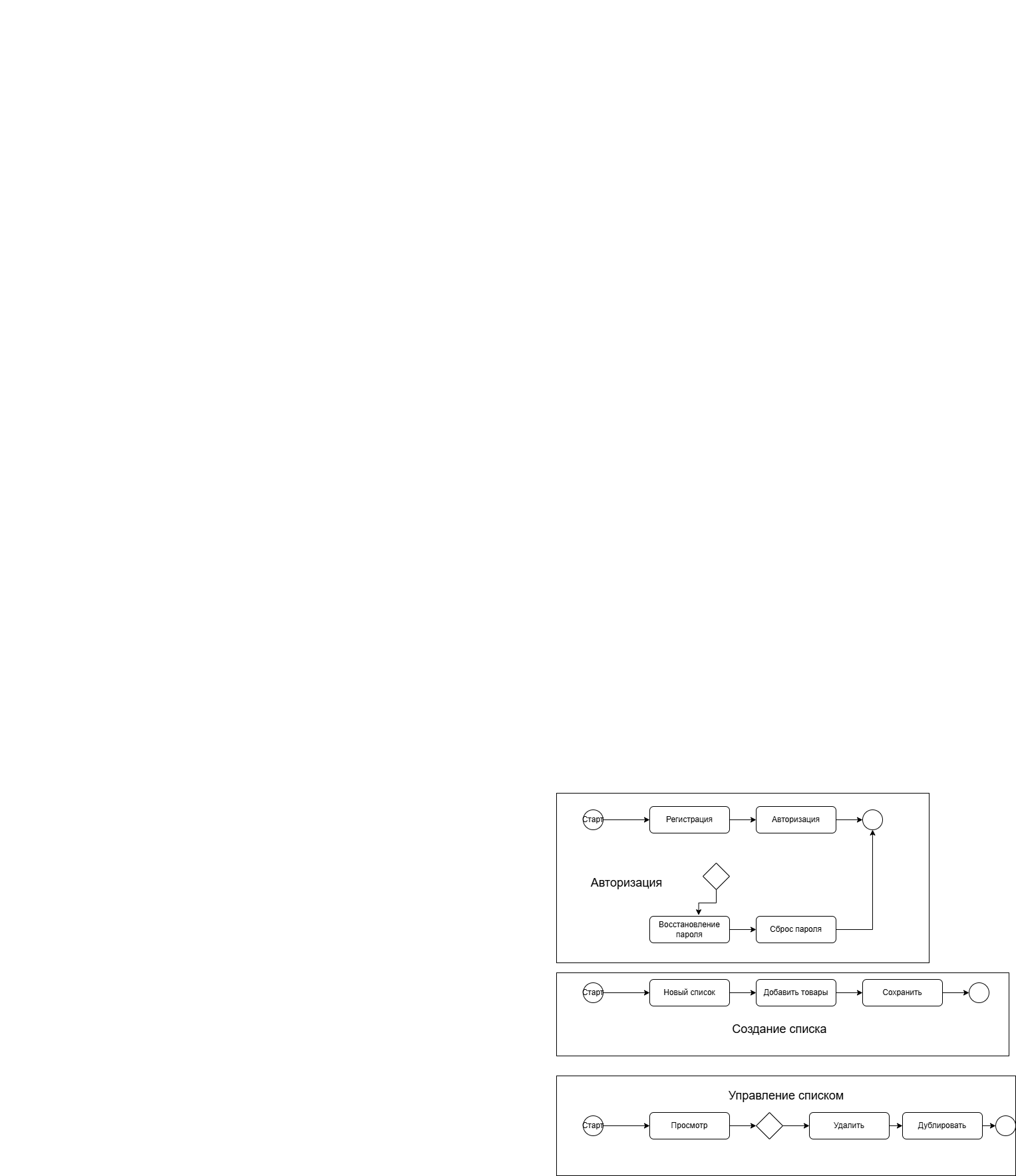
Федорин Максм Исдо-42

Задание 1



Задание 2

Основные функции приложения «Список покупок в магазинах»

1. Создание и редактирование списков покупок

Добавление, удаление и изменение товаров (название, количество, категория).

Возможность отмечать товары как купленные/некупленные.

1. Синхронизация данных между устройствами

Автоматическое сохранение списков в облаке.

Доступ к спискам с любого устройства после авторизации.

1. Категоризация товаров

Группировка товаров по категориям (например, «Молочные продукты», «Овощи», «Бытовая химия»).

Возможность создавать пользовательские категории.

1. Напоминания и уведомления

Установка времени и даты для напоминаний о походе в магазин.

Push-уведомления с актуальным списком покупок.

1. Совместное использование списков

Возможность делиться списками с другими пользователями (семья, друзья).

Редактирование списка несколькими участниками в режиме реального времени.

1. Резервное копирование и восстановление данных

Автоматическое резервное копирование в облако.

1. Голосовой ввод товаров

Добавление товаров через голосовые команды

1. Интуитивный интерфейс

Простой дизайн с возможностью настройки тем (светлая/тёмная тема).

Быстрый поиск по спискам и товарам.

Задание 3

**API методы**

**Добавление списка:**

POST /api/lists

{

"name": "Продукты на неделю",

"user\_id": 123

}

**Ответ:**

{ "id": 456, "name": "Продукты на неделю", "created\_at": "2023-10-20" }



**Удаление списка:**

DELETE /api/lists/{listId} 

**Добавление товара:**

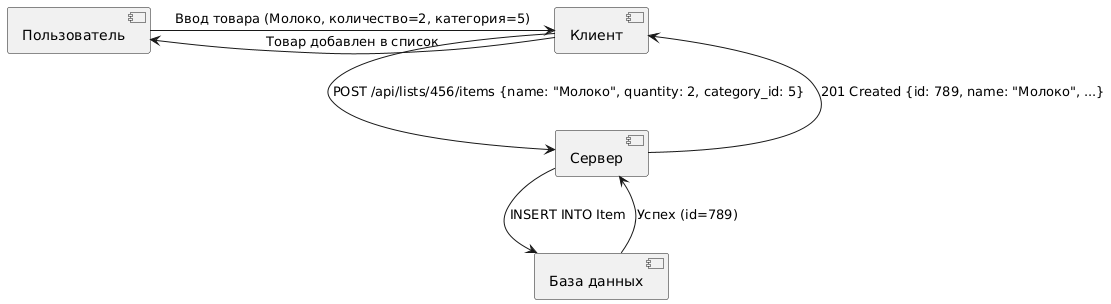
POST /api/lists/{listId}/items

{

"name": "Молоко",

"quantity": 2,

"category\_id": 5

} 

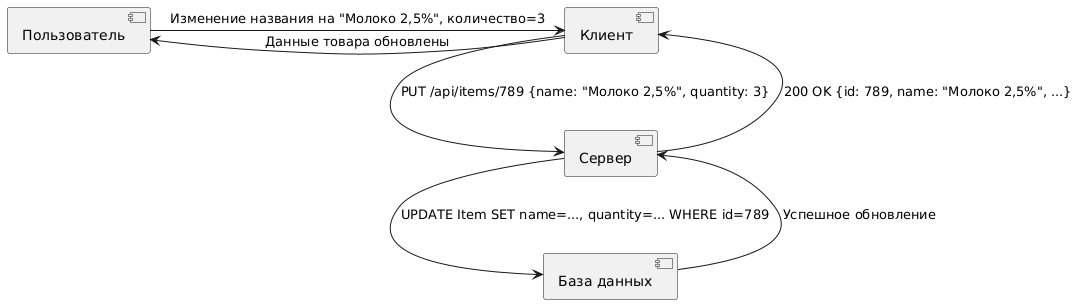
**Редактирование товара:**

PUT /api/items/{itemId}

{

"name": "Молоко 2,5%",

"quantity": 3

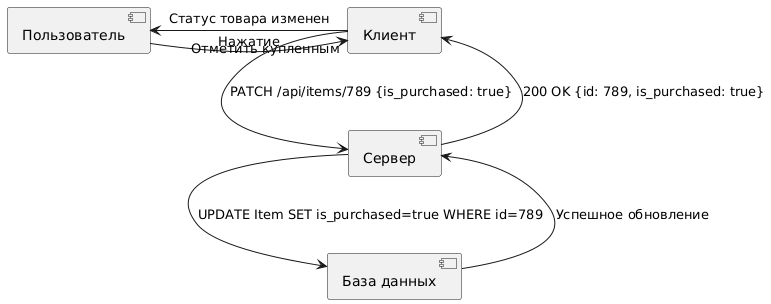
} 

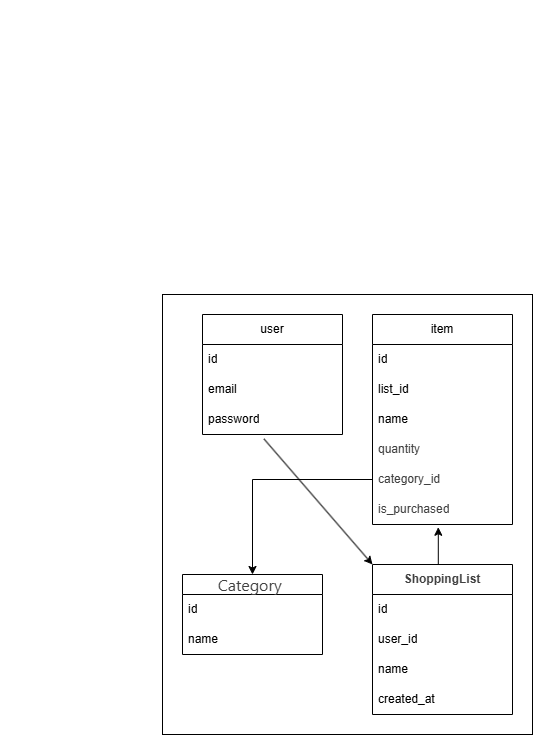
**Покупка/отмена покупки:**

PATCH /api/items/{itemId}

{

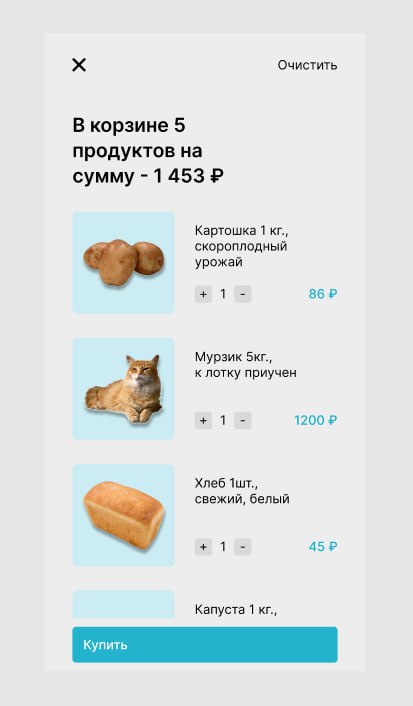
"is\_purchased": true

} 

ER-диаграмма сущностей:

Задание 4

https://www.figma.com/design/ErMBdfQyUgsGhd0grUxcxs/Untitled?node-id=0-1&t=QMwUrzXHwisaVVQf-1



Задание 5

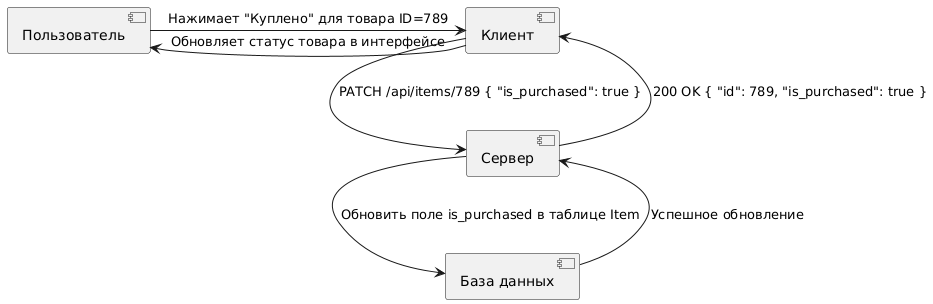
**1. Текстовое описание функции**

Функция позволяет пользователю отмечать товары в списке покупок как купленные или некупленные.

Сценарии использования:

* Пользователь отмечает товар как купленный после его покупки.
* Пользователь отменяет отметку, если товар не был куплен.

**2. UML-диаграмма последовательности**

****

**3. API-метод**

**Пример запроса**:

PATCH /api/items/789

{

"is\_purchased": true

}

**Ответы:**

**Успех (200 OK):**

{

"id": 789,

"is\_purchased": true,

"updated\_at": "2023-10-20T15:30:00Z"

}

**Ошибки**:

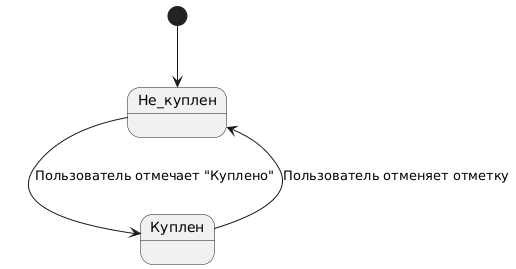
404 Not Found — товар не существует.

400 Bad Request — некорректный запрос

#### **4. Хранение информации о покупках**

**Структура таблицы Item в БД:**

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| id | INT (PK) | Уникальный ID товара. |
| list\_id | INT (FK) | ID списка покупок. |
| name | VARCHAR(255) | Название товара. |
| is\_purchased | BOOLEAN | Статус покупки (true/false). |
| updated\_at | TIMESTAMP | Время последнего изменения. |



Задние 6

**Основные сложности разработки приложения:**

1. **Синхронизация данных**
   * Реализация мгновенной синхронизации между устройствами.
   * Обработка конфликтов при одновременном редактировании списков.
2. **Безопасность**
   * Защита персональных данных (пароли, списки покупок).
   * Предотвращение утечек и DDoS-атак.
3. **Производительность**
   * Оптимизация запросов к базе данных при большом количестве пользователей.
   * Масштабируемость архитектуры.
4. **Интеграции**
   * Работа с API сторонних сервисов (если планируется интеграция с магазинами).
   * Обработка ошибок и ограничений сторонних систем.
5. **Юзабилити**
   * Создание интуитивно понятного интерфейса.
   * Адаптивность под мобильные устройства и десктопы.
6. **Тестирование**
   * Покрытие всех сценариев, включая краевые случаи (например, потеря соединения).
   * Автоматизация тестирования.

### Вопросы к заказчику:

1. **Целевая аудитория**
   * Кто основные пользователи? Физические лица, семьи, бизнес?
   * Нужна ли функция совместного редактирования списков?
2. **Масштабируемость**
   * Ожидается ли рост пользователей в ближайшие 1-2 года?
   * Какие требования к производительности (время отклика, нагрузка)?
3. **Интеграции**
   * Планируется ли интеграция с магазинами/доставкой в будущем?
4. **Безопасность**
   * Требуется ли соответствие GDPR или другим стандартам?
   * Нужна ли двухфакторная аутентификация?
5. **Мобильные платформы**
   * Нужны ли нативные приложения для iOS/Android или достаточно PWA?
6. **Монетизация**
   * Как будет зарабатывать приложение (подписка, реклама, платные функции)?
7. **Резервное копирование**
   * Как часто нужно делать бэкапы?
   * Сколько времени хранить данные удаленных списков?
8. **Локализация**
   * Планируется ли поддержка нескольких языков и региональных настроек?
9. **Технологический стек**
   * Есть ли предпочтения по языкам, фреймворкам, базам данных?

Задание 7

**1. Общая стоимость книг для каждого автора (сортировка по убыванию):**

SELECT

a.AuthorName,

SUM(b.Price) AS TotalPrice

FROM

Authors a

LEFT JOIN

Books b ON a.id = b.AuthorId

GROUP BY

a.AuthorName

ORDER BY

TotalPrice DESC;

**Результат:**

AuthorName | TotalPrice

Дж. Роулинг | 2900

Л.Н.Толстой | 1600

М.А.Булгаков | 2000

А.С.Пушкин | 1500

П.Коэльо | 700

Д.Браун | NULL

**2. Авторы, у которых общая стоимость книг превышает 1500:**

SELECT

a.AuthorName,

SUM(b.Price) AS TotalPrice

FROM

Authors a

LEFT JOIN

Books b ON a.id = b.AuthorId

GROUP BY

a.AuthorName

HAVING

SUM(b.Price) > 1500;

**Результат:**

AuthorName | TotalPrice

Дж. Роулинг | 2900

Л.Н.Толстой | 1600

М.А.Булгаков | 2000

**3. Авторы 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;

**Результат:**

AuthorName | BookCount

А.С.Пушкин | 2

М.А.Булгаков | 1

Л.Н.Толстой | 2

П.Коэльо | 1

Дж. Роулинг | 1

Д.Браун | 0

**4. Авторы без книг:**

SELECT

a.AuthorName

FROM

Authors a

LEFT JOIN

Books b ON a.id = b.AuthorId

WHERE

b.id IS NULL;

**Результат:**

AuthorName

Д.Браун