

****Тестовое задание по аналитике

**Выполнил**: студент 4-го курса колледжа экономики и информатики.

**Группы**: ИСдо-42

Каравашкин Денис Константинович

**Задание 1**

Файл с бизнес-логикой приложения в репозитории. Название «Задание 1»

**Задание 2**

**1. Создание и управление списками покупок**

**Описание**: Возможность создавать, редактировать и удалять списки покупок.

**2. Добавление и редактирование товаров в списке**

**Описание**: Возможность добавлять товары в список, указывать их количество, а также редактировать или удалять их.

**3. Отметка купленных товаров**

**Описание**: Возможность отмечать товары как купленные (например, с помощью чекбоксов), чтобы отслеживать прогресс.

**4. Синхронизация данных между устройствами**

**Описание**: Возможность синхронизировать списки покупок между несколькими устройствами пользователя через учетную запись.

**7. Совместное использование списков**

**Описание**: Возможность делиться списками с другими пользователями (например, с членами семьи).

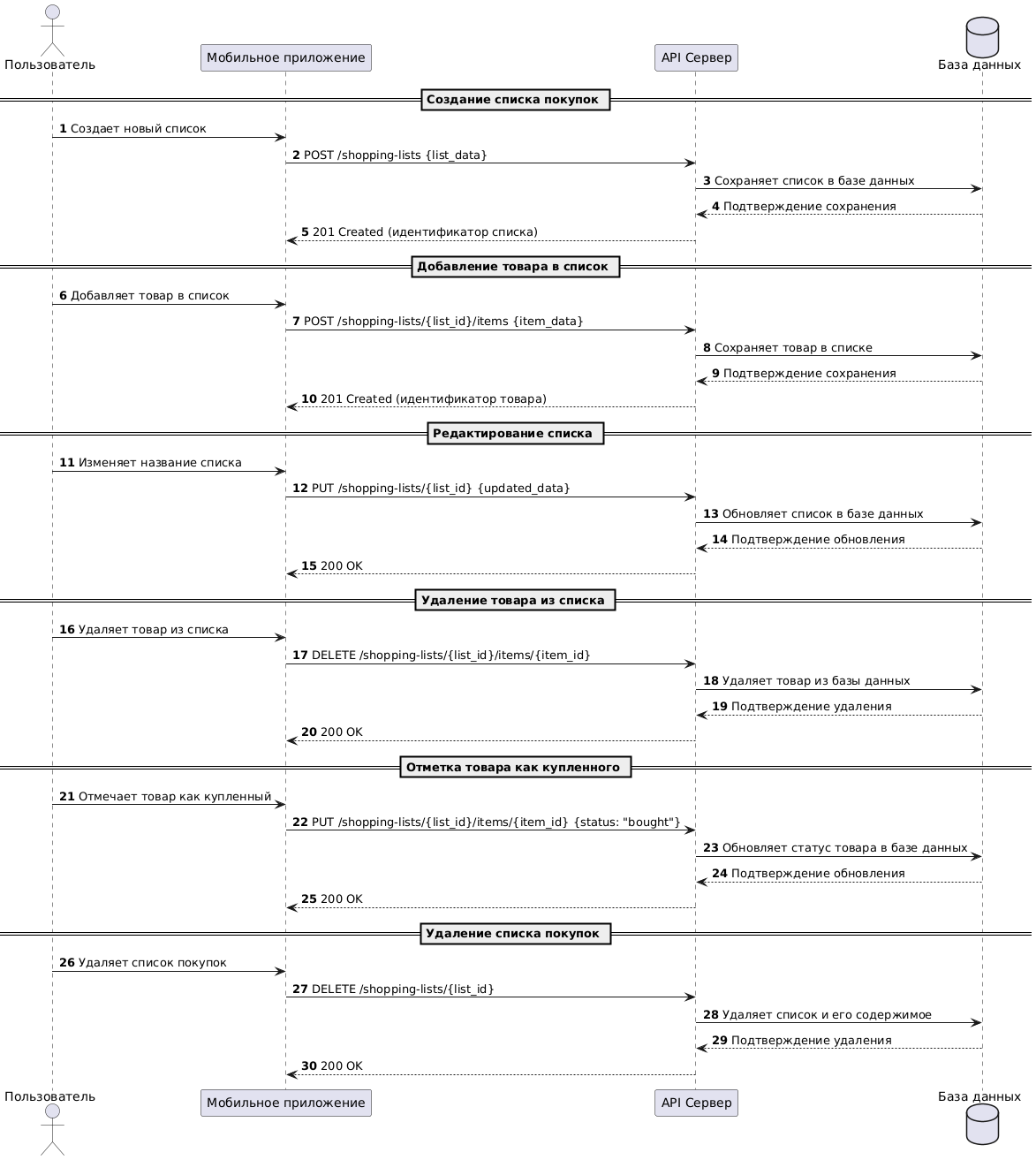
**8. Напоминания о покупках**

**Описание**: Возможность устанавливать напоминания о необходимости совершить покупки (например, перед выходом в магазин).

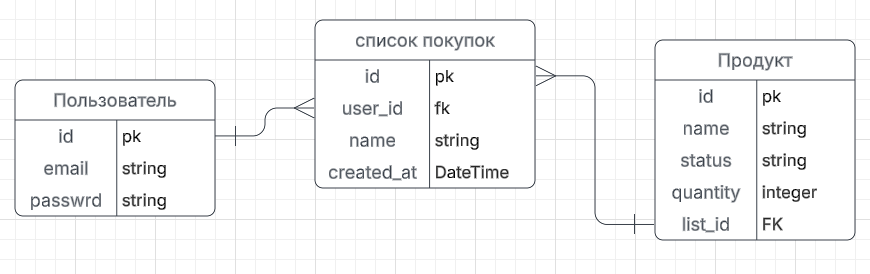
**9. Оффлайн-режим**

**Описание**: Возможность использовать приложение без подключения к интернету с последующей синхронизацией данных.

**Задание 3**



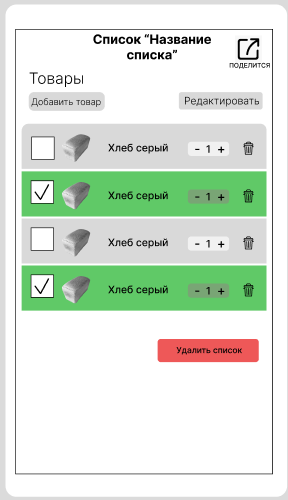
Ссылка[: https://github.com/spyrro/practika-for-Mediasoft-2025/blob/main/%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%203.png](:%20https:/github.com/spyrro/practika-for-Mediasoft-2025/blob/main/%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%203.png)



Синхронизация данных между клиентом (мобильным приложением) и сервером включает в себя:

1. **Добавление списка покупок**
2. **Удаление списка покупок**
3. **Добавление товара в список**
4. **Редактирование списка**
5. **Удаление товара из списка**
6. **Отметка товара как купленного**
7. **Обновление данных между клиентом и сервером**

**Задание 4**



Ссылка на макет: <https://www.figma.com/design/5dbaHv2S9BxFLeXBSSts25/Untitled?node-id=0-1&p=f&t=buS03lDIV0Km1stY-0>

**Задание 5**

**1. Общее описание функции**

Функция покупки товара позволяет пользователю отметить товар из списка покупок как купленный. Это помогает пользователю отслеживать, какие товары уже приобретены, а какие еще нужно купить. Функция должна быть реализована как на клиентской стороне (мобильное приложение), так и на серверной (синхронизация данных между устройствами).

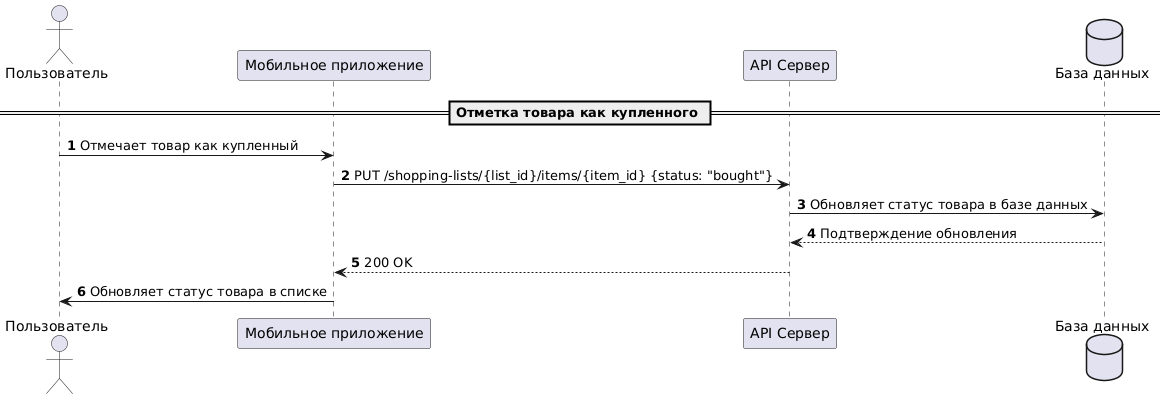
**2. Основные шаги функции**

1. **Отображение списка товаров:**
   * Пользователь видит список товаров, которые он добавил в список покупок.
   * Каждый товар отображается с названием, количеством (если указано) и статусом (куплен/не куплен).
2. **Выбор товара для отметки:**
   * Пользователь может выбрать товар из списка, нажав на него или на иконку/чекбокс рядом с названием товара.
3. **Отметка товара как купленного:**
   * При нажатии на товар или чекбокс, товар помечается как купленный.
   * Визуально это может быть отображено:
     + Зачеркиванием названия товара.
     + Изменением цвета текста или фона.
     + Иконкой "галочка" или другим визуальным индикатором.
4. **Сохранение статуса товара:**
   * Статус товара (куплен/не куплен) сохраняется локально на устройстве пользователя.
   * Данные о статусе товара синхронизируются с сервером, чтобы изменения отображались на всех устройствах пользователя.
5. **Отмена отметки (опционально):**
   * Пользователь может снять отметку о покупке, если товар был отмечен по ошибке.
   * При повторном нажатии на товар или чекбокс, статус товара изменяется на "не куплен".

**3. Требования к логике**

* **Клиентская часть:**
  + Приложение должно сохранять статус товара (куплен/не куплен) локально на устройстве.
  + При отсутствии интернета, изменения должны сохраняться локально и синхронизироваться с сервером при восстановлении соединения.
* **Серверная часть:**
  + Сервер должен принимать данные о статусе товара от клиента и обновлять их в базе данных.
  + Сервер должен отправлять обновленные данные на все устройства пользователя для синхронизации.
* **Обработка ошибок:**
  + Если синхронизация с сервером не удалась, приложение должно уведомить пользователя и повторить попытку позже.
  + При конфликте данных (например, если товар был изменен на другом устройстве), приложение должно предложить пользователю выбрать актуальный статус.

**UML Sequence диаграмма:**



Ссылка: <https://github.com/spyrro/practika-for-Mediasoft-2025/blob/main/%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%205.png>

**API методы**

**3.1. Отметка товара как купленного**

**Запрос:**

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

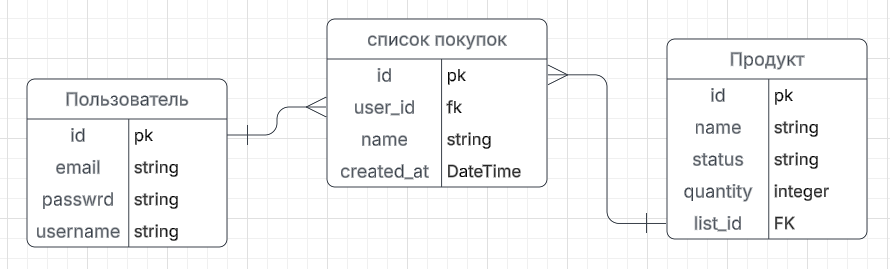
**4. Хранение на сервере**

Серверное хранение необходимо для синхронизации данных между устройствами пользователя и для резервного копирования.

**4.1. Структура данных на сервере**

На сервере данные могут храниться в реляционной или NoSQL базе данных. Пример структуры:

* **Таблица "Пользователи":**
  + user\_id (уникальный идентификатор пользователя).
  + Password (пароль пользователя).
  + username (имя пользователя).
  + email (электронная почта).
* **Таблица "Списки покупок":**
  + list\_id (уникальный идентификатор списка).
  + user\_id (связь с пользователем).
  + name (название списка).
  + created\_at (дата создания списка).
* **Таблица "Продукты":**
  + item\_id (уникальный идентификатор товара).
  + list\_id (связь со списком).
  + item\_name (название товара).
  + quantity (количество).
  + status (куплен/не куплен).



**4.2. Процесс работы с серверными данными**

1. **Синхронизация данных:**
   * При запуске приложения или изменении данных, клиент отправляет запрос на сервер для синхронизации.
   * Сервер возвращает актуальные данные, которые сохраняются локально.
2. **Обновление статуса товара:**
   * При отметке товара как купленного, клиент отправляет запрос на сервер для обновления статуса.
   * Сервер обновляет данные в базе и уведомляет другие устройства пользователя.

### **Пример процесса хранения**

1. Пользователь добавляет товар "Молоко" в список покупок.
2. Данные сохраняются в локальной базе.
3. При наличии интернета, данные отправляются на сервер через REST API.
4. Сервер сохраняет данные в БД и уведомляет другие устройства пользователя.
5. На другом устройстве пользователя данные синхронизируются, и товар "Молоко" отображается в списке.

**Задание 6**

**Вопросы заказчику для уточнения ТЗ**

Нужно ли совместное редактирование списка несколькими пользователями?

Должен ли список автоматически удаляться или архивироваться после покупки всех товаров?

Какие категории товаров будут предустановлены в системе?

Нужно ли сохранять историю покупок для последующего анализа?

Какие устройства и операционные системы должны поддерживаться (iOS, Android)?

Есть ли возможность пользоваться функционалом приложения не создавая аккаунт?

Нужно ли шифровать данные списка покупок?

Как реализовать доступ к списку между пользователями (например, передача через QR-код)?

Нужно ли распознавание текста из фото чеков для автоматического добавления товаров?

Требуется ли темная тема и адаптивный дизайн?

Как пользователи будут получать уведомления о незавершенных покупках?

**Задание 7**

1. **Общая стоимость книг для каждого автора, отсортированная 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;

1. **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;

1. **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;

1. **Получить автора без книг:**

SELECT a.AuthorName

FROM Authors a

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

WHERE b.Id IS NULL;