Никонова Дарья Александровна

УлГТУ

ИСТбд-21

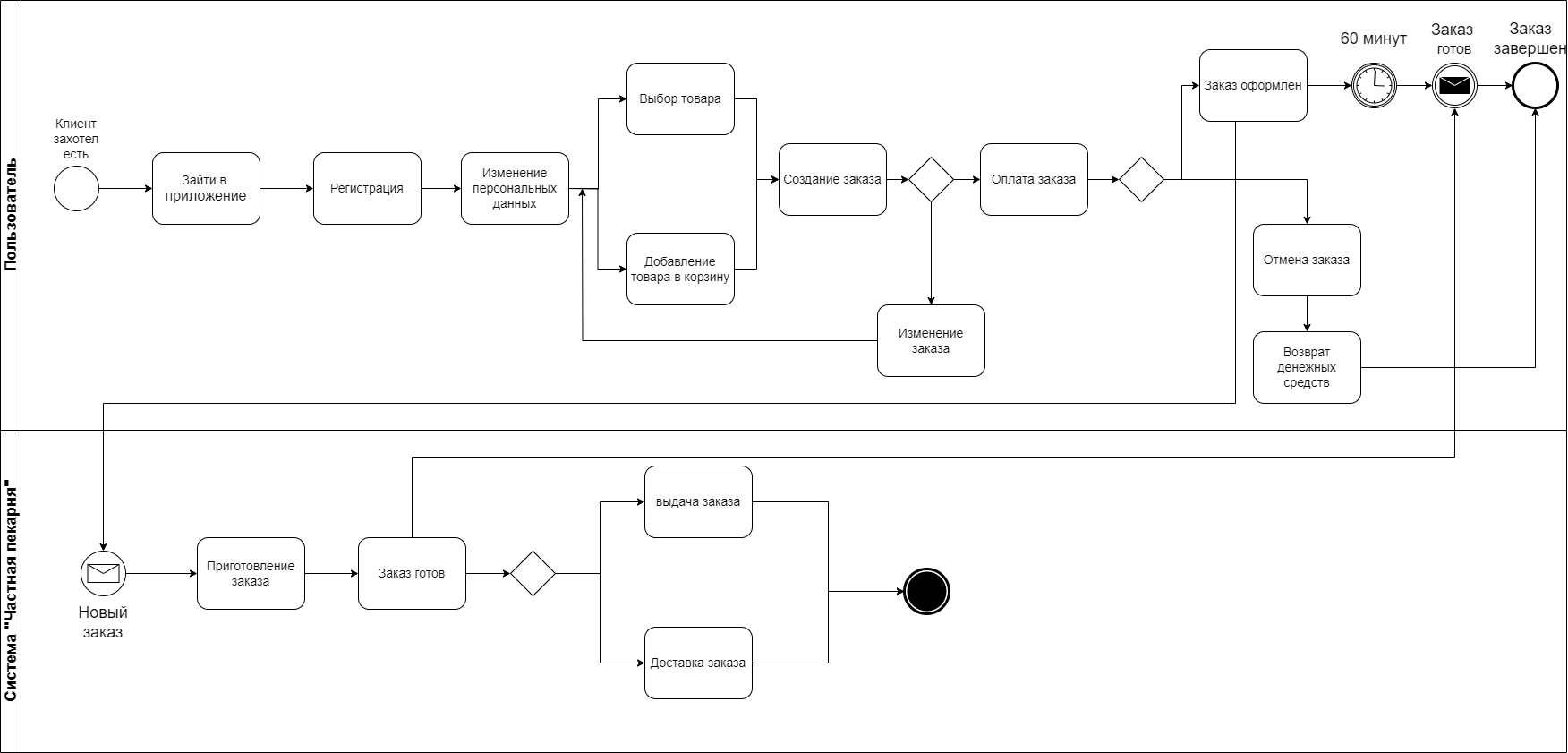
**Дано:**

Мобильное клиент-серверное приложение "Частная пекарня" (приложение для продажи выпечки)

**Необходимо описать:**

1. **Бизнес-процесс создания заказа (использовать любую удобную нотацию моделирования бизнес-процессов).**

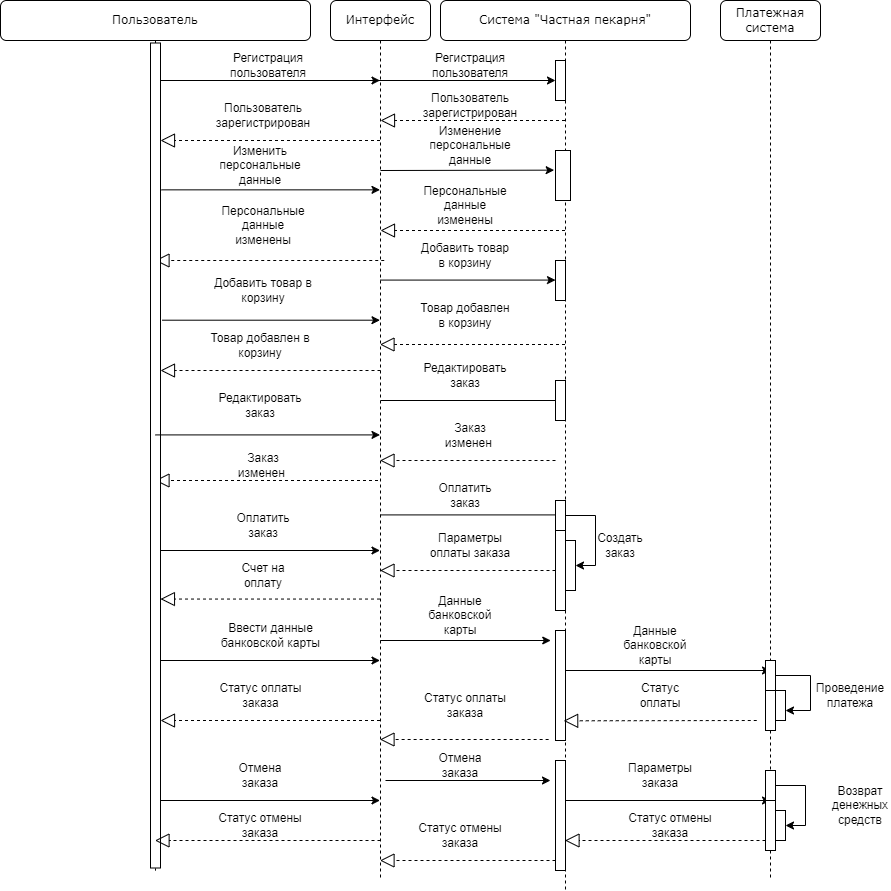
Нотация BPMN:



1. **Опишите процесс синхронизации данных между клиентом и сервером**

**(создание, редактирование и отмена заказа, изменение персональных данных, оплата заказа, добавление товара в корзину и регистрация пользователя). Представить все в диаграммах UML, API методах и других представлениях, также составить ER-диаграмму сущностей.**

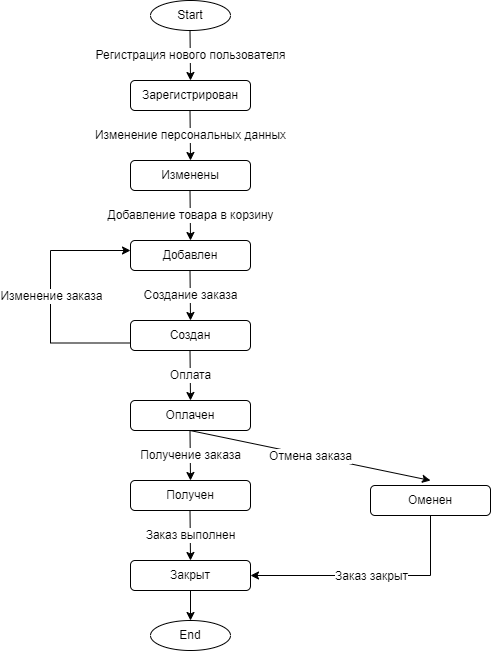
Диаграмма UML:



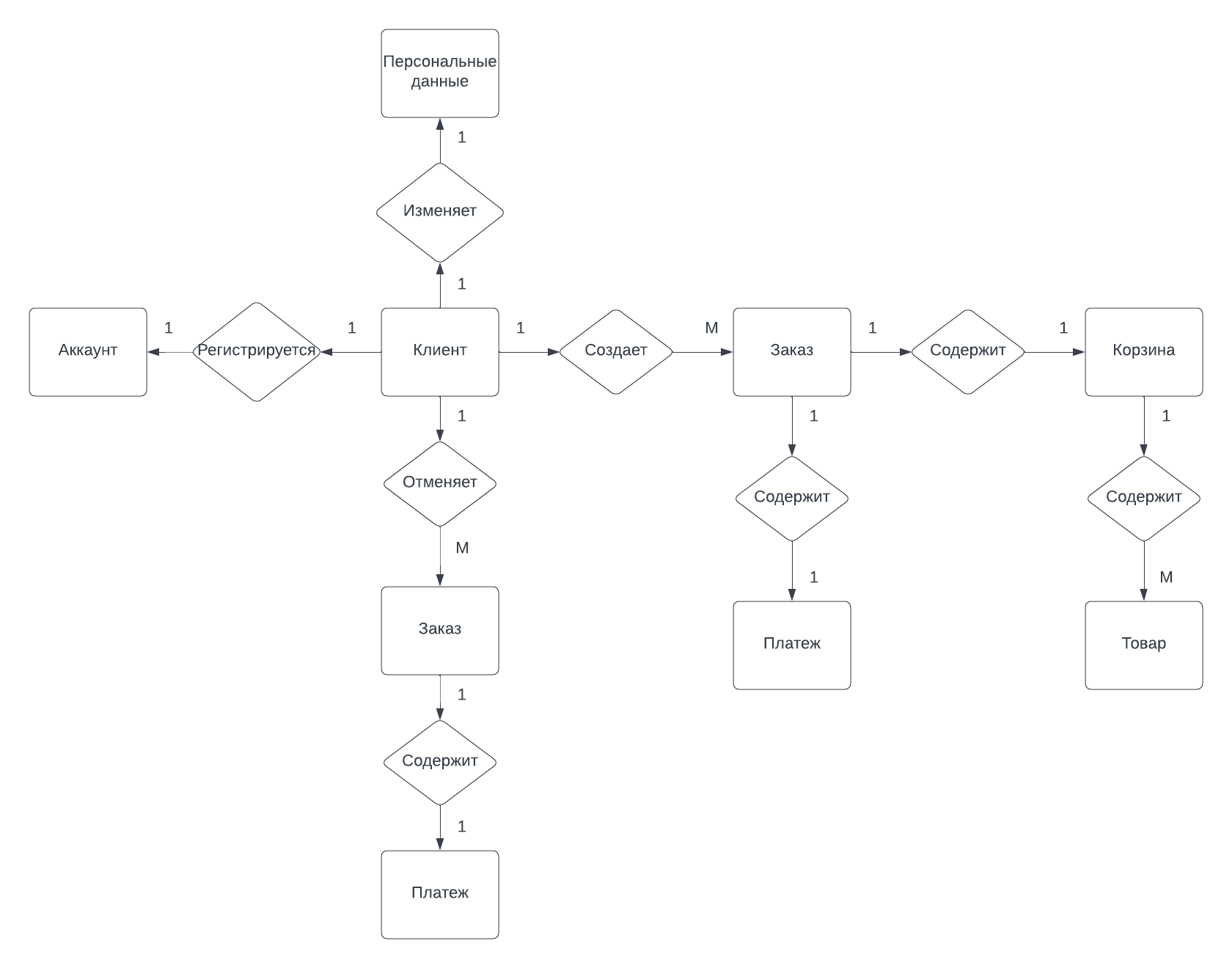
API методы:

|  |  |
| --- | --- |
| API методы | Значение |
| POST /api/register | Регистрация пользователя |
| PUT /api/user/{userId} | Изменение персональных данных |
| POST /api/cart/add | Добавление товара в корзину |
| POST /api/order/create | Создание заказа |
| PUT /api/order/edit/{orderId} | Редактирование заказа |
| DELETE /api/order/cance/{orderId} | Отмена заказа |
| POST /api/order/pay/{orderId} | Оплата заказа |

State Machine Diagram:

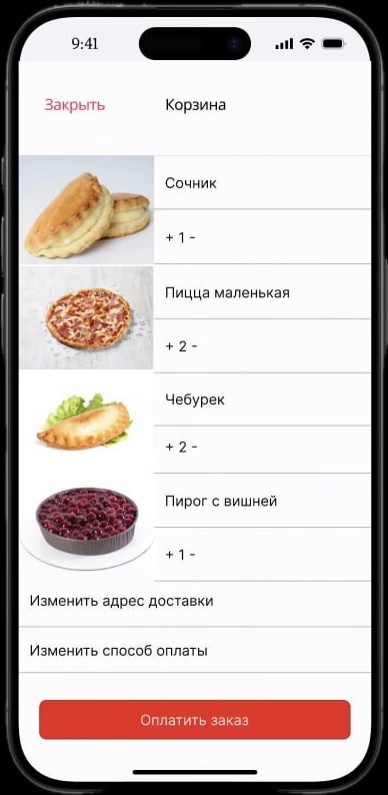


ER-диаграмма сущностей:



1. **Подготовить прототип одного из экранов данного мобильного приложения и описать пользовательский интерфейс для данного экрана (например, создание заказа).**

Экран корзины заказа:

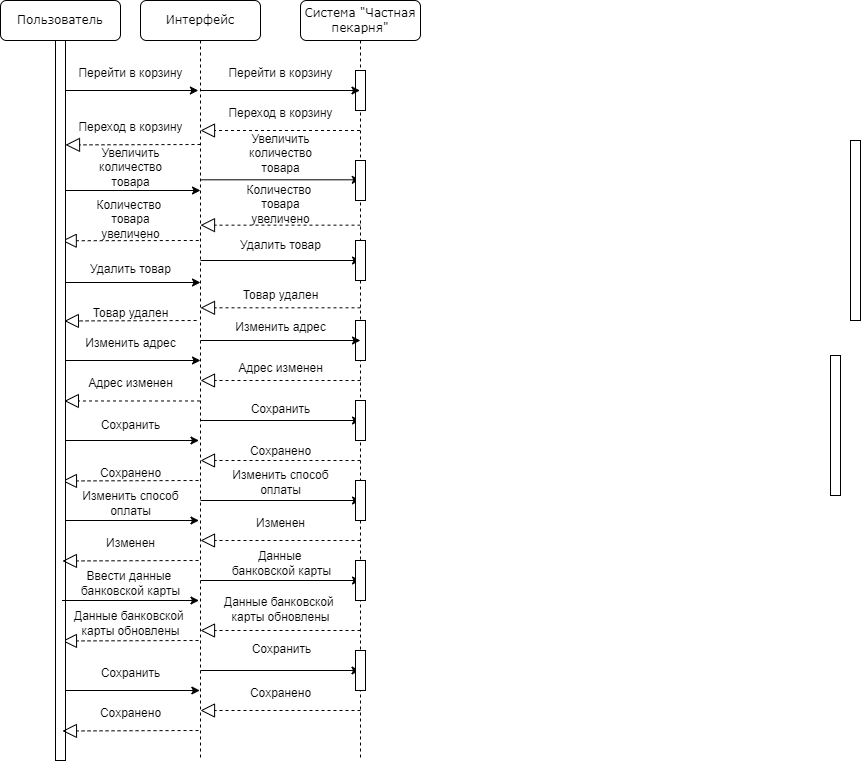


1. **Подготовить подробное описание функции редактирования заказа, которую можно было бы использовать в качестве постановки задачи для разработки (помимо текстового описания, использовать UML диаграммы, указать используемые API методы, передаваемые и получаемые параметры, описать процесс хранения информации о покупках пользователя).**

Описание задачи:

Предоставить возможность клиенту по изменять содержимое существующего заказа, добавлять или удалять товары, а также изменять адрес доставки и способ оплаты.

UML диаграмма:



API методы:

|  |  |
| --- | --- |
| API методы | Значение |
| GET /orders/{order\_id} | Получение информации о конкретном заказе по его идентификатору |
| *PUT /orders/{order\_id}* | Обновление информации о заказе |
| *PATCH /orders/{order\_id}* | Частичное обновление информации о заказе (например, изменение адреса доставки) |
| *POST /orders/{order\_id}/items* | Добавление товара в заказ |
| *DELETE /orders/{order\_id}/items/{item\_id}* | Удаление товара из заказа |

Передаваемые и получаемые параметры:

*order\_id****:*** Уникальный идентификатор заказа

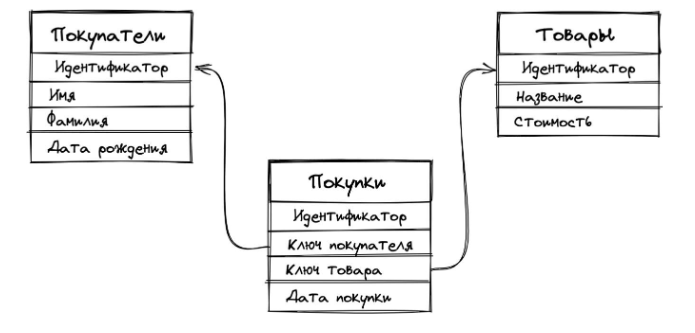
*item\_id*: Уникальный идентификатор товара

*Новые данные о заказе:* Данные о товарах, адресе доставки, способе оплаты.

Описание процесса хранения информации о покупках пользователя:

Для хранения информации о покупках пользователя можно использовать базу данных с таблицами "Заказы" и "Товары в заказе". В таблице "Заказы" хранятся основные данные о заказе (идентификатор, статус, адрес доставки, способ оплаты), а таблица "Товары в заказе" содержит связанные с заказом товары (идентификатор товара, количество, цена).

1. **Перед вами реляционная модель данных. Необходимо написать SQL-запросы.**



1. Вывести покупателей с количеством осуществленных покупок

SELECT п.имя, п.фамилия, COUNT(пк.идентификатор) AS количество\_покупок

FROM покупатели п

INNER JOIN покупки пк ON п.идентификатор = пк.идентификатор

GROUP BY п. идентификатор, п.имя, п.фамилия;

1. Общую стоимость товаров для каждого покупателя и отсортировать    результат в порядке убывания

SELECT п.имя, п.фамилия, SUM(т.стоимость) AS общая\_стоимость

FROM покупатели п

INNER JOIN покупки пк ON п.идентификатор = пк.идентификатор

INNER JOIN товары т ON пк.идентификатор = т. идентификатор

GROUP BY п.идентификатор, п.имя, п.фамилия

ORDER BY общая\_стоимость DESC;

1. Получить покупателей, купивших только один товар

SELECT п.имя, п.фамилия

FROM покупатели п

INNER JOIN покупки пк ON п.идентификатор = пк.идентификатор

GROUP BY п.идентификатор, п.имя, п.фамилия

HAVING COUNT(пк.идентификатор) = 1;