**Тулапин Тимофей Андреевич Исд-33**

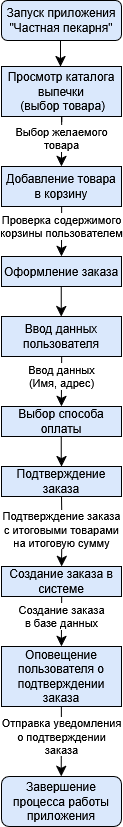
**Аналитика**

Задание 1

Для описания бизнес-процесса создания заказа я использую нотацию **BPMN**.

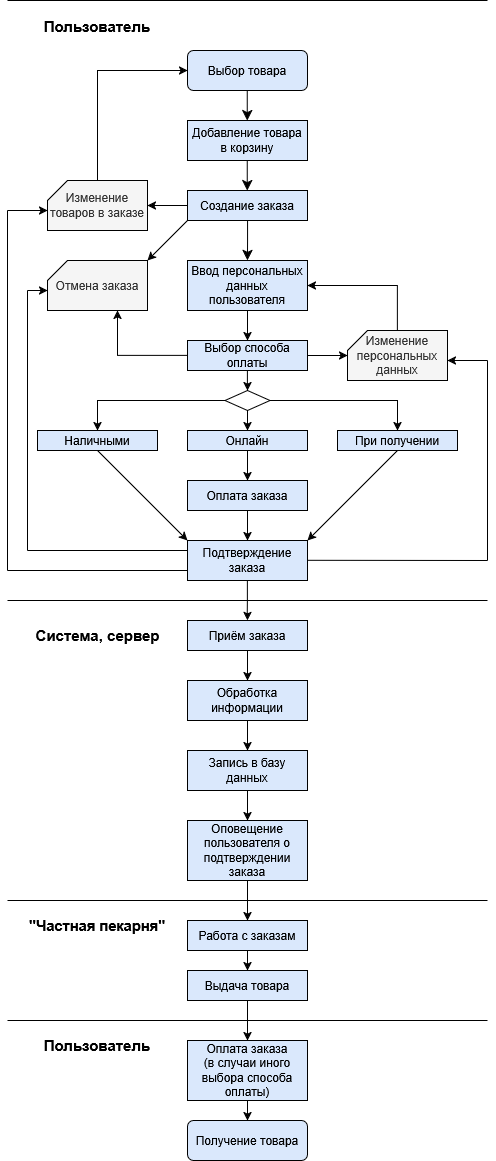
1. **Начало:** запуск приложения "Частная пекарня", пользователь решает сделать заказ.
2. **Выбор товаров:** пользователь выбирает товары, которые хочет приобрести.
3. **Добавление в корзину:** выбранные товары добавляются в корзину.
4. **Оформление заказа:** пользователь подтверждает заказ.
5. **Ввод данных пользователя:** пользователь вводит свои данные (имя, адрес, номер телефона).
6. **Выбор способа оплаты:** пользователь выбирает способ оплаты (онлайн, наличными при получении.).
7. **Подтверждение заказа:** успешной оплаты заказ подтверждается.
8. **Уведомление пользователю:** пользователь получает уведомление о подтверждении заказа.
9. **Завершение:** процесс завершается.

Также описания бизнес-процесса создания заказа можно изобразить в виде диаграммы.



Задание 2

Диаграмма UML описывающая процесс синхронизации данных между клиентом и сервером.



API методы использующиеся при синхронизации данных между клиентом и сервером.

**Создание заказа:** POST

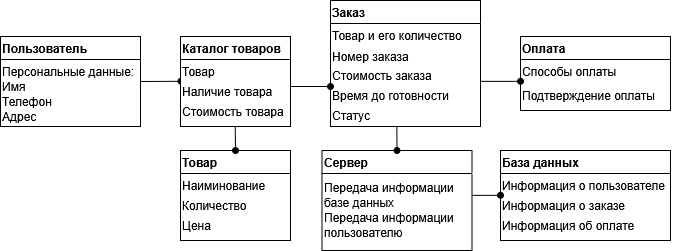
**Изменение кол-во товаров:** PUT

**Отмена заказа:** DELETE

**Изменение персональных данных:** PUT

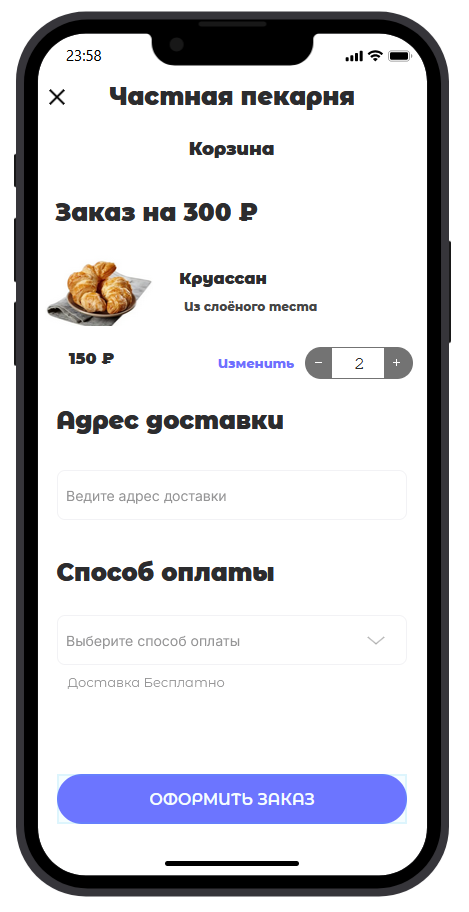
**Оплата заказа:** POST

ER-диаграмму сущностей.



Задание 3

Прототип экрана создания заказа.



**Описание пользовательского интерфейса**

В самом верху экрана находится название приложения.

Ниже находится корзина, в которой у нас отображается сумма, на которую будет сделан заказ, еще ниже название товара, фото товара, и цена, также присутствует кнопка изменения товара и его количества.

Далее идёт строка введения адреса доставки.

Затем находится кнопка выбора способа оплаты.

И в самом низу экрана находится кнопка оформления заказа.

Задание 4

Описание функций редактирования заказа.

**Идентификация заказа (orderID) -** Уникальный идентификатор заказа, связанный с пользователем.

**Редактирование списка товаров (items) -** каждый элемент списка должен содержать: уникальный идентификатор товара (itemID), название товара(name), количество единиц товара(quantity) и цену (price).

**Редактирование адреса(address)** - каждый адрес должен иметь уникальный идентификатор адреса(addresID).

**Выбор способа оплаты (PaymentMethod) -** уникальный метод способа оплаты(paymentID).

ER диаграмма процесса хранения информации о покупках пользователя.

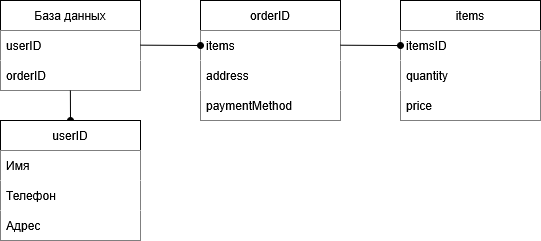
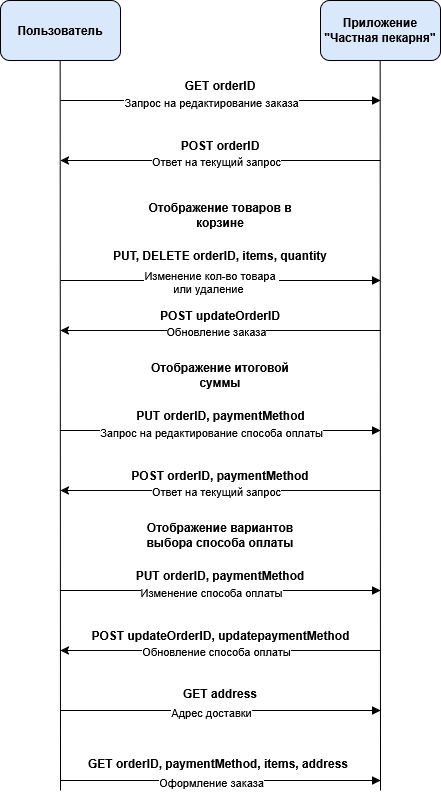
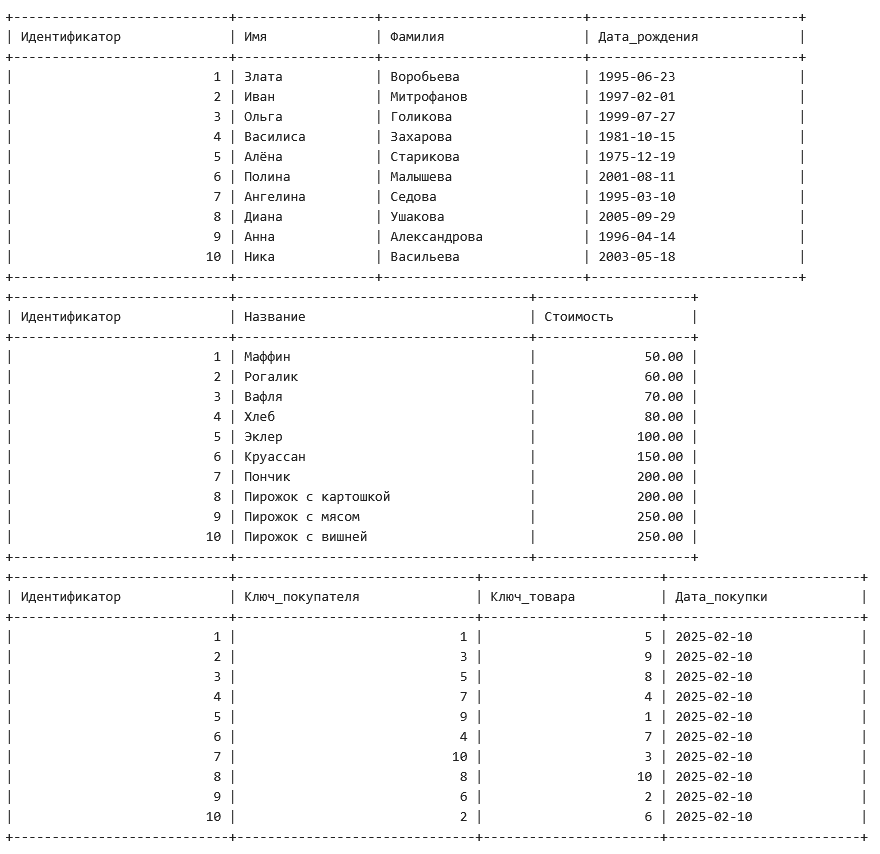


Диаграмма Описание функции редактирования заказа.



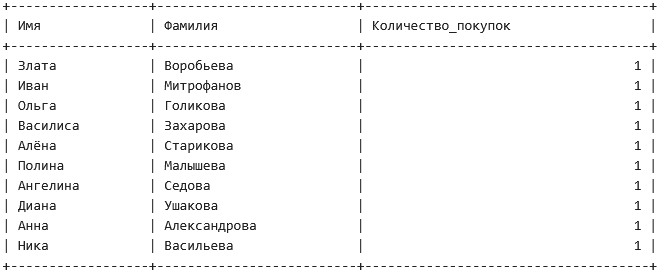
Задание 5

Реляционная модель данных.

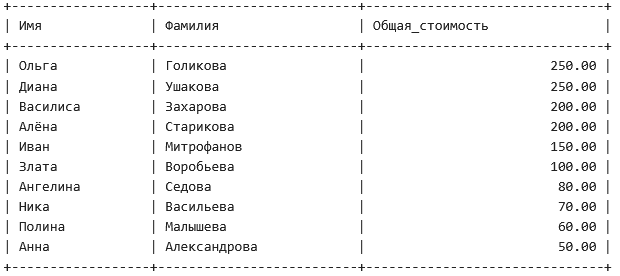


Задания к модели данных.

1. Покупателей с количеством осуществленных покупок



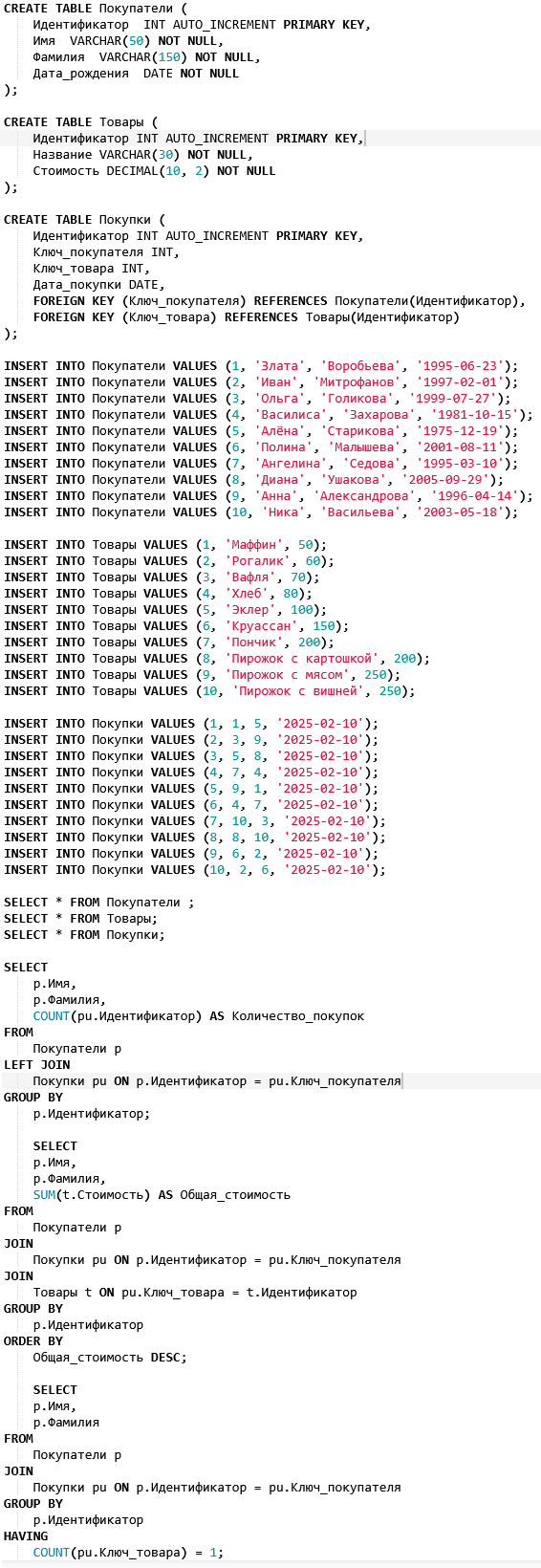
1. Общая стоимость товаров для каждого покупателя и отсортированная в порядке убывания.



1. Покупатели, купившие только один товар.



Скриншот кода базы данных:



Код базы данных:

CREATE TABLE Покупатели (

Идентификатор INT AUTO\_INCREMENT PRIMARY KEY,

Имя VARCHAR(50) NOT NULL,

Фамилия VARCHAR(150) NOT NULL,

Дата\_рождения DATE NOT NULL

);

CREATE TABLE Товары (

Идентификатор INT AUTO\_INCREMENT PRIMARY KEY,

Название VARCHAR(30) NOT NULL,

Стоимость DECIMAL(10, 2) NOT NULL

);

CREATE TABLE Покупки (

Идентификатор INT AUTO\_INCREMENT PRIMARY KEY,

Ключ\_покупателя INT,

Ключ\_товара INT,

Дата\_покупки DATE,

FOREIGN KEY (Ключ\_покупателя) REFERENCES Покупатели(Идентификатор),

FOREIGN KEY (Ключ\_товара) REFERENCES Товары(Идентификатор)

);

INSERT INTO Покупатели VALUES (1, 'Злата', 'Воробьева', '1995-06-23');

INSERT INTO Покупатели VALUES (2, 'Иван', 'Митрофанов', '1997-02-01');

INSERT INTO Покупатели VALUES (3, 'Ольга', 'Голикова', '1999-07-27');

INSERT INTO Покупатели VALUES (4, 'Василиса', 'Захарова', '1981-10-15');

INSERT INTO Покупатели VALUES (5, 'Алёна', 'Старикова', '1975-12-19');

INSERT INTO Покупатели VALUES (6, 'Полина', 'Малышева', '2001-08-11');

INSERT INTO Покупатели VALUES (7, 'Ангелина', 'Седова', '1995-03-10');

INSERT INTO Покупатели VALUES (8, 'Диана', 'Ушакова', '2005-09-29');

INSERT INTO Покупатели VALUES (9, 'Анна', 'Александрова', '1996-04-14');

INSERT INTO Покупатели VALUES (10, 'Ника', 'Васильева', '2003-05-18');

INSERT INTO Товары VALUES (1, 'Маффин', 50);

INSERT INTO Товары VALUES (2, 'Рогалик', 60);

INSERT INTO Товары VALUES (3, 'Вафля', 70);

INSERT INTO Товары VALUES (4, 'Хлеб', 80);

INSERT INTO Товары VALUES (5, 'Эклер', 100);

INSERT INTO Товары VALUES (6, 'Круассан', 150);

INSERT INTO Товары VALUES (7, 'Пончик', 200);

INSERT INTO Товары VALUES (8, 'Пирожок с картошкой', 200);

INSERT INTO Товары VALUES (9, 'Пирожок с мясом', 250);

INSERT INTO Товары VALUES (10, 'Пирожок с вишней', 250);

INSERT INTO Покупки VALUES (1, 1, 5, '2025-02-10');

INSERT INTO Покупки VALUES (2, 3, 9, '2025-02-10');

INSERT INTO Покупки VALUES (3, 5, 8, '2025-02-10');

INSERT INTO Покупки VALUES (4, 7, 4, '2025-02-10');

INSERT INTO Покупки VALUES (5, 9, 1, '2025-02-10');

INSERT INTO Покупки VALUES (6, 4, 7, '2025-02-10');

INSERT INTO Покупки VALUES (7, 10, 3, '2025-02-10');

INSERT INTO Покупки VALUES (8, 8, 10, '2025-02-10');

INSERT INTO Покупки VALUES (9, 6, 2, '2025-02-10');

INSERT INTO Покупки VALUES (10, 2, 6, '2025-02-10');

SELECT \* FROM Покупатели ;

SELECT \* FROM Товары;

SELECT \* FROM Покупки;

SELECT

p.Имя,

p.Фамилия,

COUNT(pu.Идентификатор) AS Количество\_покупок

FROM

Покупатели p

LEFT JOIN

Покупки pu ON p.Идентификатор = pu.Ключ\_покупателя

GROUP BY

p.Идентификатор;

SELECT

p.Имя,

p.Фамилия,

SUM(t.Стоимость) AS Общая\_стоимость

FROM

Покупатели p

JOIN

Покупки pu ON p.Идентификатор = pu.Ключ\_покупателя

JOIN

Товары t ON pu.Ключ\_товара = t.Идентификатор

GROUP BY

p.Идентификатор

ORDER BY

Общая\_стоимость DESC;

SELECT

p.Имя,

p.Фамилия

FROM

Покупатели p

JOIN

Покупки pu ON p.Идентификатор = pu.Ключ\_покупателя

GROUP BY

p.Идентификатор

HAVING

COUNT(pu.Ключ\_товара) = 1;