

Лабораторная работа №4 По курсу “Проектирование человеко-машинных интерфейсов”

**Проектирование информационной архитектуры и взаимодействия с пользователем**

“Bimbimbambam”  
4 курс, 12 группа

## Постановка задачи

Онлайн-магазин цифровой техники (каталог и управление заказами цифровой техники). Должны быть реализованы сценарии: покупка товара, поиск товара, добавление нового товара в базу данных магазина, просмотр и обработка заказов покупателей, регистрация нового покупателя. Фильтрация товаров по различным критериям.

### Задание 1. Проектирование модели требований и реестра вариантов использования

Актор	User Story (US)	Use Case (UC)
Покупатель	US-1.1 Как покупатель онлайн-магазина, я хочу искать товары, чтобы быстро находить нужную технику.	UC 1.1 Поиск товаров
Покупатель	US-1.2 Как покупатель онлайн-магазина, я хочу просматривать карточку товара, чтобы узнать подробную информацию о продукте (описание, цена, наличие, характеристики).	UC 1.2 Просмотр карточки товара
Покупатель	US-1.3 Как покупатель онлайн-магазина, я хочу добавлять товары в корзину, чтобы сформировать заказ.	UC 1.3 Добавление в корзину
Покупатель	US-1.4 Как покупатель онлайн-магазина, я хочу оформить заказ, чтобы приобрести выбранные товары.	UC 1.4 Оформление заказа
Покупатель	US-1.5 Как покупатель онлайн-магазина, я хочу оплатить заказ онлайн, чтобы завершить покупку.	UC 1.5 Оплата заказа
Покупатель	US-1.6 Как покупатель онлайн-магазина, я хочу зарегистрироваться, чтобы иметь возможность оформлять заказы и отслеживать их статус.	UC 1.6 Регистрация
Покупатель	US-1.7 Как покупатель онлайн-магазина, я хочу авторизоваться, чтобы получить доступ к своему профилю и истории заказов.	UC 1.7 Авторизация
Покупатель	US-1.8 Как покупатель онлайн-магазина, я хочу оставлять отзывы о товарах, чтобы поделиться своим мнением с другими пользователями.	UC 1.8 Оставить отзыв
Покупатель	US-1.9 Как покупатель онлайн-магазина, я хочу просмотреть отзывы о товарах, чтобы узнать больше о товаре от покупателей.	UC 1.9 Просмотреть отзыв
Менеджер	US-2.1.1 Как менеджер онлайн-магазина, я хочу видеть перечень всех товаров с их данными (название, поставщик, стоимость, количество единиц в наличии), чтобы управлять ассортиментом.	UC 2.1.1 Просмотр списка товаров
Менеджер	US-2.1.2 Как менеджер онлайн-магазина, я хочу добавлять новый товар, задавая данные о нем	UC 2.1.2 Добавить товар

	(название, поставщик, стоимость, количество единиц в наличии), чтобы пополнять ассортимент.	
Менеджер	US-2.1.3 Как менеджер онлайн-магазина, я хочу управлять конкретным товаром, изменяя его данные (название, поставщик, стоимость, количество единиц в наличии) или удаляя его, чтобы поддерживать актуальность каталога.	UC 2.1.3 Посмотреть товар UC 2.1.4 Изменить товар UC 2.1.5 Удалить товар
Менеджер	US-2.2.1 Как менеджер онлайн-магазина, я хочу видеть перечень клиентских заказов, чтобы контролировать процесс выполнения.	UC 2.2.1 Просмотр списка заказов
Менеджер	US-2.2.2 Как менеджер онлайн-магазина, я хочу просматривать детали конкретного заказа (товары, клиент, сумма, дата, способ оплаты), чтобы отслеживать процесс покупки.	UC 2.2.2 Просмотр заказа
Менеджер	US-2.2.3 Как менеджер онлайн-магазина, я хочу изменять состояние заказа (новый, в обработке, отправлен, доставлен, отменен), чтобы управлять его жизненным циклом и информировать клиента.	UC 2.2.3 Изменить состояние заказа
Менеджер	US-2.3 Как менеджер онлайн-магазина, я хочу авторизоваться, чтобы получить доступ к функциям управления товарами и заказами.	UC 2.3 Авторизация

## Задание 2. Проектирование физической модели базы данных

Сущность	Таблица БД	Поле	Смысл поля	Тип данных
Категория	categories	id	Уникальный идентификатор категории	UUID
		name	Название категории	VARCHAR
Товар	products	id	Уникальный идентификатор товара	UUID
		sku	Уникальный артикул товара	VARCHAR (UNIQUE)
		name	Название товара	VARCHAR
		category_id	Ссылка на категорию товара	UUID (FK → categories.id)
		stock_qty	Количество единиц товара на складе	INT
		price	Цена товара	NUMERIC(12,2)
		rating	Средний рейтинг товара	NUMERIC(4,1)

Отзыв	reviews	id	Уникальный идентификатор отзыва	UUID
		product_id	Ссылка на товар, к которому относится отзыв	UUID (FK → products.id)
		user_id	Ссылка на клиента, оставившего отзыв	UUID (FK → users.id)
		rating	Оценка товара	INT
		text	Текст отзыва	TEXT
		created_at	Дата и время создания отзыва	TIMESTAMP
Пользователь	users	id	Уникальный идентификатор пользователя	UUID
		email	Электронная почта пользователя	VARCHAR (UNIQUE)
		password_hash	Хэш пароля пользователя	TEXT
		name	Имя пользователя	VARCHAR
		phone	Контактный номер телефона	VARCHAR
Товар в корзине	cart_items	id	Уникальный идентификатор записи в корзине	UUID
		user_id	Ссылка на пользователя	UUID (FK → users.id)
		product_id	Ссылка на товар	UUID (FK → products.id)
		qty	Количество единиц товара	INT
		price	Цена за единицу товара	NUMERIC(12,2)
Заказ	orders	id	Уникальный идентификатор заказа	UUID
		user_id	Ссылка на покупателя	UUID (FK → users.id)
		address_id	Ссылка на адрес доставки	UUID (FK → addresses.id)
		status	Статус заказа (новый, обработан, доставлен, отменён и т.д.)	VARCHAR
		tracking_num	Номер для отслеживания заказа	
		created_at	Дата и время создания заказа	TIMESTAMP
		total	Общая сумма заказа	NUMERIC(12,2)

Товар в заказе	order_items	id	Уникальный идентификатор позиции в заказе	UUID
		order_id	Ссылка на заказ	UUID (FK → orders.id)
		product_id	Ссылка на товар	UUID (FK → products.id)
		qty	Количество единиц товара	INT
		price	Цена товара в заказе	NUMERIC(12,2)
Платёж	payments	id	Уникальный идентификатор платежа	UUID
		order_id	Ссылка на заказ	UUID (FK → orders.id)
		provider	Платёжная система или провайдер	VARCHAR
		status	Статус платежа (успешен, отклонён, ожидает)	VARCHAR
		amount	Сумма платежа	NUMERIC(12,2)
Адрес доставки	addresses	id	Уникальный идентификатор адреса	UUID
		order_id	Ссылка на заказ	UUID (FK → order.id)
		line1	Улица и дом	VARCHAR
		city	Город	VARCHAR
		postal_code	Почтовый индекс	VARCHAR
		country	Страна	VARCHAR

## ERD-диаграмма базы данных

@startuml

!theme plain

```
entity "categories" as categories {
    *id : UUID <<PK>>
    --
    name : VARCHAR
}
```

```
entity "products" as products {
    *id : UUID <<PK>>
    --
    sku : VARCHAR <<UNIQUE>>
    name : VARCHAR
    category_id : UUID <<FK>>
    stock_qty : INT
    price : NUMERIC(12,2)
    rating : NUMERIC(4,1)
}
```

```
entity "reviews" as reviews {
    *id : UUID <<PK>>
    --
    product_id : UUID <<FK>>
    user_id : UUID <<FK>>
    rating : INT
    text : TEXT
    created_at : TIMESTAMP
}
```

```
entity "users" as users {
    *id : UUID <<PK>>
    --
    email : VARCHAR <<UNIQUE>>
    password_hash : TEXT
    name : VARCHAR
    phone : VARCHAR
}
```

```
entity "cart_items" as cart_items {
    *id : UUID <<PK>>
    --
    user_id : UUID <<FK>>
    product_id : UUID <<FK>>
    qty : INT
    price : NUMERIC(12,2)
}
```

```

entity "orders" as orders {
  *id : UUID <<PK>>
  --
  user_id : UUID <<FK>>
  address_id : UUID <<FK>>
  status : VARCHAR
  tracking_num : VARCHAR
  created_at : TIMESTAMP
  total : NUMERIC(12,2)
}

entity "order_items" as order_items {
  *id : UUID <<PK>>
  --
  order_id : UUID <<FK>>
  product_id : UUID <<FK>>
  qty : INT
  price : NUMERIC(12,2)
}

entity "payments" as payments {
  *id : UUID <<PK>>
  --
  order_id : UUID <<FK>>
  provider : VARCHAR
  status : VARCHAR
  amount : NUMERIC(12,2)
}

entity "addresses" as addresses {
  *id : UUID <<PK>>
  --
  order_id : UUID <<FK>>
  line1 : VARCHAR
  city : VARCHAR
  postal_code : VARCHAR
  country : VARCHAR
}

```

#### ' Relationships

```

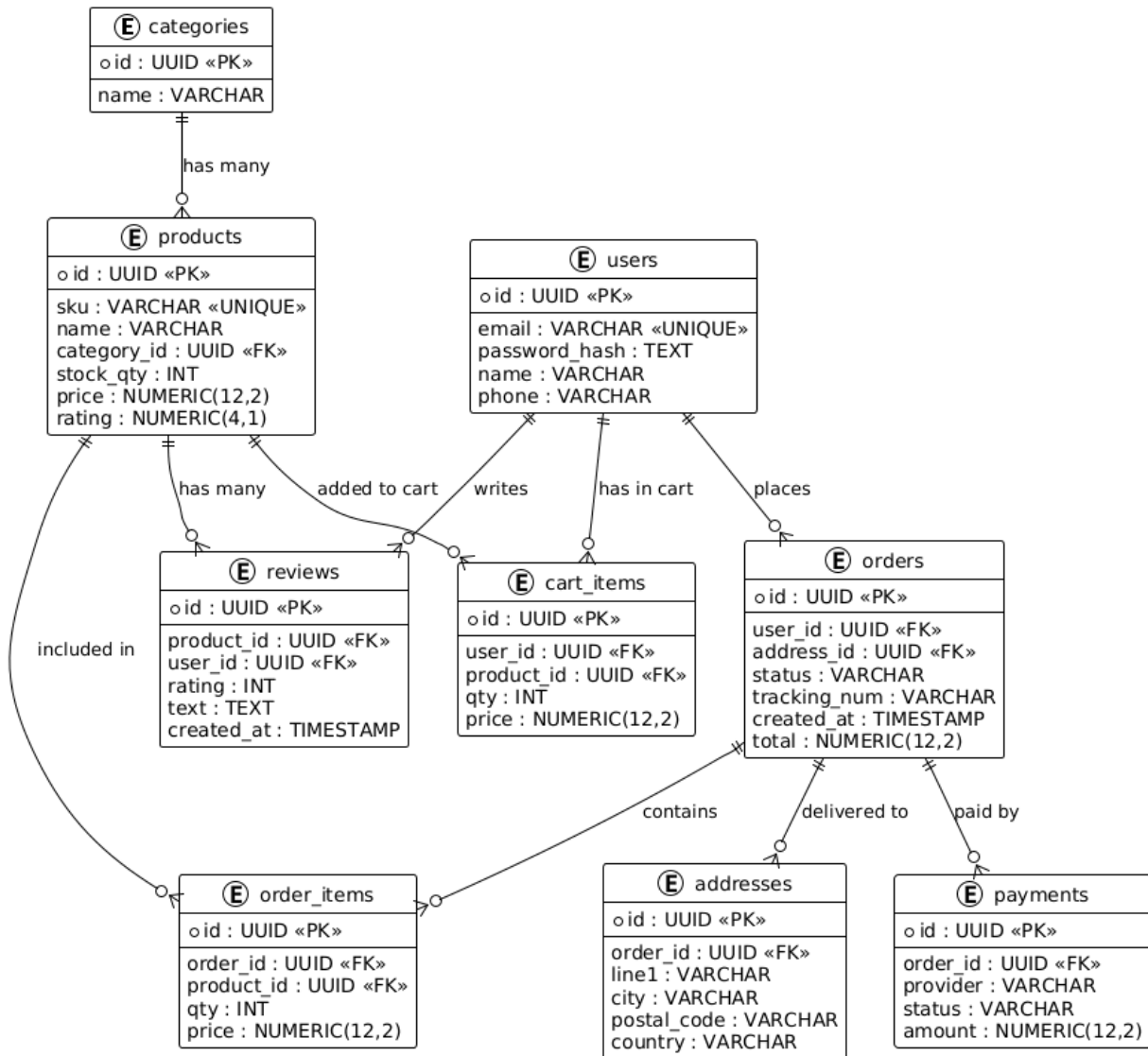
categories ||--o{ products : "has many"
products ||--o{ reviews : "has many"
users ||--o{ reviews : "writes"
users ||--o{ cart_items : "has in cart"
users ||--o{ orders : "places"
orders ||--o{ order_items : "contains"
orders ||--o{ payments : "paid by"
orders ||--o{ addresses : "delivered to"

```

```

products ||--o{ cart_items : "added to cart"
products ||--o{ order_items : "included in"
@enduml

```



Листинг файла в формате sql:

```

CREATE TABLE categories (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  name VARCHAR NOT NULL
);

CREATE TABLE products (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  sku VARCHAR UNIQUE NOT NULL,
  name VARCHAR NOT NULL,
  category_id UUID REFERENCES categories(id) ON DELETE SET NULL,

```



```
    stock_qty INT DEFAULT 0,  
    price NUMERIC(12,2) NOT NULL,  
    rating NUMERIC(4,1) DEFAULT 0  
);
```

```
CREATE TABLE users (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    email VARCHAR UNIQUE NOT NULL,  
    password_hash TEXT NOT NULL,  
    name VARCHAR NOT NULL,  
    phone VARCHAR  
);
```

```
CREATE TABLE reviews (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    product_id UUID NOT NULL REFERENCES products(id) ON DELETE CASCADE,  
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,  
    rating INT CHECK (rating BETWEEN 1 AND 5),  
    text TEXT,  
    created_at TIMESTAMP DEFAULT NOW()  
);
```

```
CREATE TABLE cart_items (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,  
    product_id UUID NOT NULL REFERENCES products(id) ON DELETE CASCADE,  
    qty INT NOT NULL CHECK (qty > 0),  
    price NUMERIC(12,2) NOT NULL  
);
```

```
-- Optional: ensure each user has one cart entry per product  
CREATE UNIQUE INDEX ux_cart_unique_item ON cart_items(user_id, product_id);
```

```
CREATE TABLE orders (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,  
    address_id UUID,  
    status VARCHAR NOT NULL DEFAULT 'new',  
    tracking_num VARCHAR,  
    created_at TIMESTAMP DEFAULT NOW(),  
    total NUMERIC(12,2) NOT NULL DEFAULT 0  
);
```

```
CREATE TABLE order_items (  
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),  
    order_id UUID NOT NULL REFERENCES orders(id) ON DELETE CASCADE,  
    product_id UUID NOT NULL REFERENCES products(id) ON DELETE CASCADE,  
    qty INT NOT NULL CHECK (qty > 0),  
    price NUMERIC(12,2) NOT NULL  
);
```

```
id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
order_id UUID NOT NULL REFERENCES orders(id) ON DELETE CASCADE,
product_id UUID NOT NULL REFERENCES products(id),
qty INT NOT NULL CHECK (qty > 0),
price NUMERIC(12,2) NOT NULL
);
```

```
CREATE TABLE payments (
id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
order_id UUID NOT NULL REFERENCES orders(id) ON DELETE CASCADE,
provider VARCHAR NOT NULL,
status VARCHAR NOT NULL CHECK (status IN ('успешен', 'отклонён', 'ожидает')),
amount NUMERIC(12,2) NOT NULL
);
```

```
CREATE TABLE addresses (
id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
order_id UUID REFERENCES orders(id) ON DELETE CASCADE,
line1 VARCHAR NOT NULL,
city VARCHAR NOT NULL,
postal_code VARCHAR,
country VARCHAR NOT NULL
);
```

### Задание 3. Проектирование маршрутов и конечных точек в виде спецификации API

#### 1. Проектирование, маршрутов и конечных точек

Сущность	Таблица БД	Маршрут (endpoint)
Категория	categories	/categorie
Товар	products	/product
Отзыв	reviews	/review
Корзина	carts	/cart
Заказ	orders	/order
Платёж	payments	/payment
Адрес доставки	addresses	/address
Доставка	shipments	/shipment
Пользователь	users	/registration /login
Покупатель	customers	/customer

#### 2. Сопоставить варианты использования с конечными точками, т.е. HTTP-методами, которые будут обращаться к маршрутам

Актор	Use Case	Маршрут	HTTP-запрос	Аутентификация
Покупатель / Менеджер	UC 1.1 Поиск товаров	/product?search={query}	GET	нет
Покупатель / Менеджер	UC 1.2 Просмотр карточки товара	/product/{id}	GET	нет
Покупатель	UC 1.3 Добавление в корзину	/cart/item	POST	да
Покупатель	UC 1.3 Просмотр корзины	/cart	GET	да
Покупатель	UC 1.3 Удаление товара из корзины	/cart/item/{id}	DELETE	да
Покупатель	UC 1.4 Оформление заказа	/order	POST	да

	UC 2.2.1 Просмотр списка заказов			
<b>Покупатель / Менеджер</b>	UC 1.4 Просмотр своих заказов	/order	GET	да
<b>Покупатель</b>	UC 1.5 Оплата заказа	/orders/{id}/pay	POST	да
<b>Покупатель</b>	UC 1.6 Регистрация	/register	POST	нет
<b>Покупатель / Менеджер</b>	UC 1.7 Авторизация	/login	POST	нет
<b>Покупатель</b>	UC 1.8 Оставить отзыв	/product/{id}/review	POST	да
<b>Покупатель</b>	UC 1.9 Просмотреть отзывы	/product/{id}/review	GET	нет
<b>Менеджер</b>	UC 2.1.2 Добавить товар	/product	POST	да
<b>Менеджер</b>	UC 2.1.4 Изменить товар	/product/{id}	PUT / PATCH	да
<b>Менеджер</b>	UC 2.1.5 Удалить товар	/product/{id}	DELETE	да
<b>Менеджер</b>	UC 2.2.2 Просмотр заказа	/orders/{id}	GET	да
<b>Менеджер</b>	UC 2.2.3 Изменить состояние заказа	/orders/{id}/status	PATCH	да

#### **Задание 4. Проектирование и тестирование API**

<https://app.swaggerhub.com/apis/bimbimbam/elecstore/1.0.0#/>

