**ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

**РАЗРАБОТКА БАЗЫ ДАННЫХ «Информационная система Аниме интернет-магазина»**

***КУРСОВОЙ ПРОЕКТ***

*СПЕЦИАЛЬНОСТЬ 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ*

Выполнил:

Студент группы 1912с

\_\_\_\_\_\_\_\_\_\_\_\_ / Заикин В.А.

\_\_\_\_\_\_\_\_\_\_\_\_ / Рюгин А.И.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024г.

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_/ Уляхин В.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024г.

Оценка «\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Томск 2024

Глава 1.

Описание проекта и задач

Общее описание проекта

Все пользователи системы подразделяются на две группы:

1. Администраторы
2. Пользователи

Категории товаров подразделяются на несколько групп

1. Манга
2. Кружки
3. Дакимакуры
4. Значки

Функционал пользователя:

* Регистрация
* Авторизация
* Просмотр категорий товаров
* Просмотр товаров определенной категории

Функционал авторизированного пользователя

* Просмотр своего профиля
* Выход
* Добавление товара в корзину
* Просмотр своей корзины
* Оформление заказа
* Оставление отзыва для определённого товара
* Редактирование корзины
* Редактирование отзыва
* Редактирование своего профиля
* Удаление отзыва

Функционал администратора

* Добавление категории
* Добавление товара
* Просмотр списка всех авторизированных пользователей
* Просмотр списка всех отзывов
* Просмотр всех заказов
* Редактирование категории
* Редактирование товара
* Изменение статуса заказа
* Редактирование отзыва
* Удаление категории
* Удаление товара
* Удаление пользователя
* Удаление отзыва

**Для пользователя**

**POST**

Регистрация пользователя

Запрос для регистрации нового пользователя в системе. При отправке запроса необходимо передать FormData со следующими полями:

* name – обязательное поле, строка;
* surname – обязательное поле, строка;
* patronymic – не обязательное поле, строка;
* login – обязательное и уникальное поле, строка;
* password – обязательное поле, строка;
* birth – обязательное поле, дата, формат Y-m-d;
* telephone – обязательное поле, формат число;
* email – обязательное поле, строка;

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/register  **Method:** POST  **Headers:** Content-Type: application/json  **Body:**  {  "surname": "Евген",  "name": "Евгений",  "patronymic": "Евгеньев",  "login": "log1",  "password": "pass1",  "birth": "12.02.2003",  "telephone": "+7930573565",  "email": "asdadda@mail.ru"  } | Успешная регистрация  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Регистрация прошла успешно"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  {  "error": {  "fields": [  "Поле 'login' должно содержать только буквы и цифры.",  "Поле 'birth' имеет неверный формат. Используйте ДД.ММ.ГГГГ."  ]  }  } |

Авторизация

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/login  **Method:** POST  **Headers:** Content-Type: application/json  **Body:**  {  "login": "log1",  "password": "pass1"  } | Успешная регистрация  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."  }  }  Ошибка аутентификации  **Status:** 401  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Неверные учетные данные"  }  } |

Просмотр категорий товаров

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/categories  **Method:** GET  **Headers:**  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "name": "Электроника"  },  {  "id": 2,  "name": "Одежда"  }  // Другие категории  ]  }  Ответ при отсутствии категорий  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Категории не найдены"  }  } |

Просмотр товаров определенной категории

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/category{id}  **Method:** GET  **Headers:**  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "name": "Смартфон",  "price": 10000,  "category": "Электроника"  },  {  "id": 2,  "name": "Наушники",  "price": 2000,  "category": "Электроника"  }  // Другие товары  ]  }  Ответ при отсутствии товаров  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Товары не найдены"  }  } |

Просмотр конкретного товара

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/category/product{id}  **Method:** GET  **Headers:**  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "name": "Берсерк",  "price": 10000,  "category": "Манга"  }  ]  }  Ответ при отсутствии товаров  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Товар не найден"  }  } |

**Для авторизированного пользователя**

GET

Просмотр своего профиля

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/profile  **Method:** GET  **Headers:** Authorization: Bearer {{user\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "name": "Евгений",  "surname": "Евген",  "email": "asdadda@mail.ru",  "role": "user"  }  } |

Выход

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/logout  **Method:** GET  **Headers:** Authorization: Bearer {{token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Вы успешно вышли из системы"  }  } |

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/cart  **Method:** GET  **Headers:** Authorization: Bearer {{token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "показ корзины (товары цена и кол-во)  }  } |

POST

Добавление товара в корзину

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/category/product{id}  **Method:** POST  **Headers:** Authorization: Bearer {{token}},  Content-Type: application/json  **Body:**  {  "product\_id": 1,  "quantity": 2  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Товар успешно добавлен в корзину"  }  }  Ответ при ошибке  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось добавить товар в корзину. Пожалуйста, проверьте данные запроса."  }  } |

Оформление заказа

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/cart  **Method:** POST  **Headers:** Authorization: Bearer {{token}},  Content-Type: application/json  **Body:**  {  "cart\_items": [  {  "product\_id": 1,  "quantity": 2  },  {  "product\_id": 3,  "quantity": 1  }  ],  "address": "ул. Новая 5-15",  "payment\_id": 1  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "order\_id": 123,  "dateOrder": 2023-09-16,  "message": "Заказ успешно оформлен"  }  }  Ответ при ошибке  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось оформить заказ. Пожалуйста, проверьте данные запроса."  }  } |

Оставление отзыва для определённого товара

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/category/product{id}/review  **Method:** POST  **Headers:** Authorization: Bearer {{token}},  Content-Type: application/json  **Body:**  {  "rating": 4,  "textReview": "Отличный товар!"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Отзыв успешно добавлен"  }  }  Ответ при ошибке  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Вы не можете оставить отзыв на товар, который вы не покупали."  }  } |

PATCH

Редактирование корзины

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/cart  **Method:** PATCH  **Headers:** Authorization: Bearer {{user\_token}},  Content-Type: application/json  **Body:**  {  "items": [  {  "product\_id": 1,  "quantity": 3  },  {  "product\_id": 2,  "quantity": 2  }  ]  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Корзина успешно отредактирована"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось отредактировать корзину. Пожалуйста, проверьте данные запроса."  }  } |

Редактирование отзыва

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/category/product{id}/review  **Method:** PATCH  **Headers:** Authorization: Bearer {{user\_token}},  Content-Type: application/json  **Body:**  {  "rating": 5,  "textReview": "Отличный товар! Рекомендую."  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Отзыв успешно отредактирован"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось отредактировать отзыв. Пожалуйста, проверьте данные запроса."  }  } |

Редактирование своего профиля

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/profile/  **Method:** PATCH  **Headers:** Authorization: Bearer {{user\_token}},  Content-Type: application/json  **Body:**  {  "name": "Новое Имя",  "surname": "Новая Фамилия"  Всё нахуй можно поменять  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Профиль успешно отредактирован"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось отредактировать профиль. Пожалуйста, проверьте данные запроса."  }  } |

DELETE

Удаление отзыва

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/category/product{id}  **Method:** DELETE  **Headers:** Authorization: Bearer {{user\_token}},  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Отзыв успешно удален"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось удалить отзыв. Пожалуйста, проверьте данные запроса."  }  } |

**Для Администратора**

POST

Добавление категории

Запрос для добавления новой категории в системе. При отправке запроса необходимо передать FormData со следующими полями:

* name – обязательное поле, строка;

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/category/create  **Method:** POST  **Headers:** Authorization: Bearer {{admin\_token}},  Content-Type: application/json  **Body:**  {  "name": "Новая Категория"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Категория успешно добавлена"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось добавить категорию. Пожалуйста, проверьте данные запроса."  }  } |

Добавление товара

Запрос для регистрации нового пользователя в системе. При отправке запроса необходимо передать FormData со следующими полями:

* name – обязательное поле, строка;
* description – необязательное поле, строка;
* price – обязательное поле, положительное вещественное число;
* quantity – обязательное поле, числовой;
* photo – необязательное поле, фото формата jpeg, png;
* category\_id – обязательное поле, число, категория с данным идентификатором должна быть.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/product/create  **Method:** POST  **Headers:** Authorization: Bearer {{admin\_token}},  Content-Type: application/json  **Body:**  {  "name": "Новый Товар",  "price": 1500,  "category\_id": 1  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Товар успешно добавлен"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось добавить товар. Пожалуйста, проверьте данные запроса."  }  } |

GET

Просмотр списка всех зарегистрированных пользователей

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/users  **Method:** GET  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "name": "Евгений",  "surname": "Евген",  "email": "asdadda@mail.ru",  "role": "user"  },  {  "id": 2,  "name": "Админ",  "surname": "Админов",  "email": "admin@mail.ru",  "role": "admin"  }  // Другие пользователи  ]  }  Ответ при ошибке  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Пользователи не найдены"  }  } |

Просмотр списка всех отзывов

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/reviews  **Method:** GET  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "user\_id": 1,  "product\_id": 1,  "rating": 4,  "comment": "Отличный товар!"  },  {  "id": 2,  "user\_id": 2,  "product\_id": 3,  "rating": 5,  "comment": "Супер!"  }  // Другие отзывы  ]  }  Ответ при ошибке  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Отзывы не найдены"  }  } |

Просмотр всех заказов

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/orders  **Method:** GET  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "order\_id": 123,  "user\_id": 1,  "total\_amount": 15000,  "status": "processing"  },  {  "order\_id": 124,  "user\_id": 2,  "total\_amount": 500,  "status": "shipped"  }  // Другие заказы  ]  }  Ответ при ошибке  **Status:** 404  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Заказы не найдены"  }  } |

PATCH

Редактирование категории

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/category{id}/edit  **Method:** PATCH  **Headers:** Authorization: Bearer {{admin\_token}}, Content-Type: application/json  **Body:**  {  "name": "Новая Электроника"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Категория успешно отредактирована"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось отредактировать категорию. Пожалуйста, проверьте данные запроса."  }  } |

Редактирование товара

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/product{id}/edit  **Method:** PATCH  **Headers:** Authorization: Bearer {{admin\_token}}, Content-Type: application/json  **Body:**  {  "name": "Новый Смартфон",  "price": 12000  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Товар успешно отредактирован"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось отредактировать товар. Пожалуйста, проверьте данные запроса."  }  } |

Изменение статуса заказа

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/order{id}/edit  **Method:** PATCH  **Headers:** Authorization: Bearer {{admin\_token}}, Content-Type: application/json  **Body:**  {  "status": "shipped"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Статус заказа успешно изменен"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось изменить статус заказа. Пожалуйста, проверьте данные запроса."  }  } |

Редактирование отзыва

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/review{id}/edit  **Method:** PATCH  **Headers:** Authorization: Bearer {{admin\_token}}, Content-Type: application/json  **Body:**  {  "rating": 5,  "comment": "Отличный товар! Рекомендую."  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Отзыв успешно отредактирован"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось отредактировать отзыв. Пожалуйста, проверьте данные запроса."  }  } |

DELETE

Удаление категории

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/categories/delete/{id}  **Method:** DELETE  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Категория успешно удалена"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось удалить категорию. Пожалуйста, проверьте данные запроса."  }  } |

Удаление товара

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/products/delete/{id}  **Method:** DELETE  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Товар успешно удален"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось удалить товар. Пожалуйста, проверьте данные запроса."  }  } |

Удаление пользователя

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/users/delete/{id}  **Method:** DELETE  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Пользователь успешно удален"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось удалить пользователя. Пожалуйста, проверьте данные запроса."  }  } |

Удаление отзыва

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api/admin/reviews/delete/{id}  **Method:** DELETE  **Headers:** Authorization: Bearer {{admin\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Отзыв успешно удален"  }  }  Ответ при ошибке  **Status:** 400  **Content-Type:** application/json  **Body:**  {  "error": {  "message": "Не удалось удалить отзыв. Пожалуйста, проверьте данные запроса."  }  } |

Глава 2.

# ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметной областью базы данных является информационная система аниме интернет-магазина. Эта система позволяет управлять и отслеживать различные аспекты работы магазина, включая информацию о доступных товарах, клиентах, заказов, сотрудниках. В магазине предоставляются услуги по продаже товаров (манга, фигурки, кружки и т. д), и данная база данных помогает магазину эффективно управлять своей деятельностью и взаимодействовать с клиентами.

# КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ДАННЫХ

Концептуальная модель – это отражение предметной области, для которой разрабатывается база данных. Не вдаваясь в теорию, отметим, что это некая диаграмма с принятыми обозначениями элементов. Все объекты, обозначающие вещи, обозначаются в виде прямоугольника. Атрибуты, характеризующие объект - в виде овала, а связи между объектами - ромбами. Мощность связи обозначаются стрелками (в направлении, где мощность равна многим - двойная стрелка, а со стороны, где она равна единице - одинарная).

Анализируя предметную область, можно выделить следующие сущности с их атрибутами (см. рисунок 1):

1. Сущность «Роль» с атрибутом «Название».
2. Сущность «Пользователи» с атрибутами «Имя», «Фамилия», «Отчество», «Логин», «Пароль», «Дата рождения», «Email», «Телефон», «Токен».
3. Сущность «Отзывы» с атрибутом «Текст».
4. Сущность «Заказы» с атрибутом «Адрес».
5. Сущность «Статус» с атрибутом «Название».
6. Сущность «Состав заказа» с атрибутами «Количество», «Общая стоимость».
7. Сущность «Способ оплаты» с атрибутами «Название».
8. Сущность «Товары» с атрибутами «Название», «Описание», «Цена», «Количество», «Фото».
9. Сущность «Категории» с атрибутами «Название».
10. Сущность «Корзины» с атрибутами «Количество», «Общая стоимость».

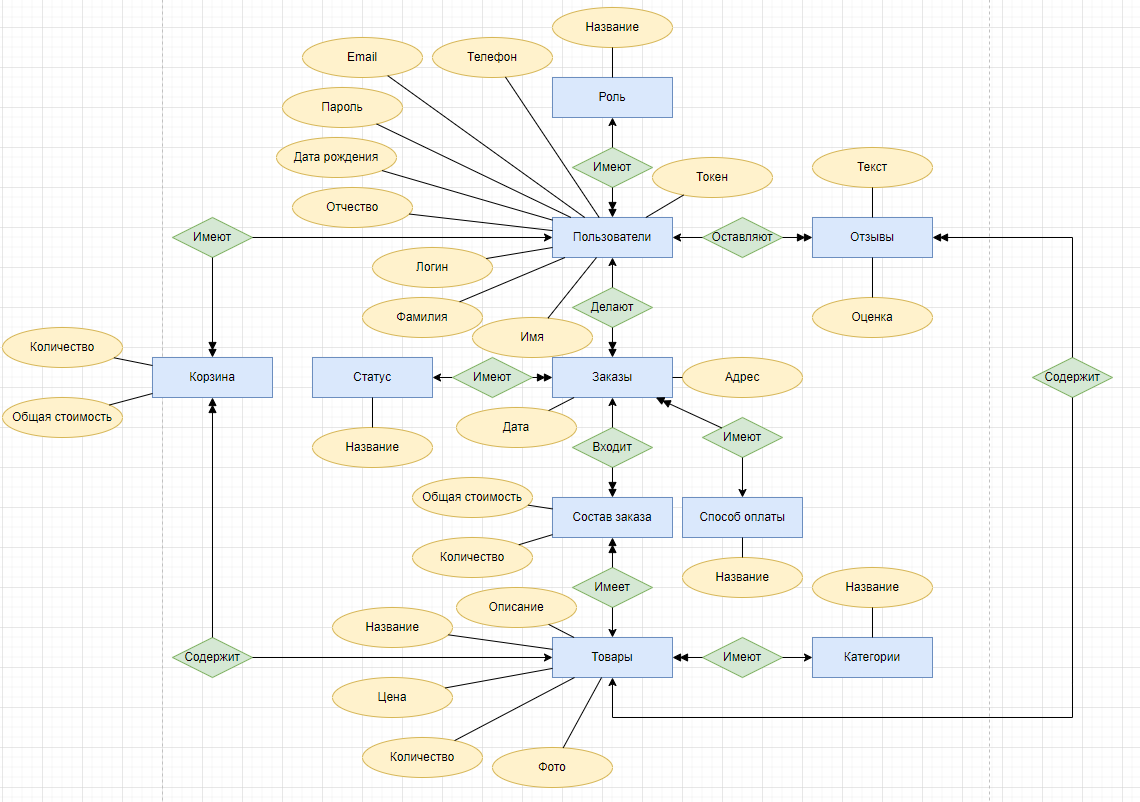


Рисунок 1 – Концептуальная модель данных

## НОРМАЛИЗАЦИЯ ДАННЫХ

Нормальная форма — свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение.

Процесс преобразования отношений базы данных к виду, отвечающему нормальным формам, называется нормализацией. Нормализация предназначена для приведения структуры БД к виду, обеспечивающему минимальную логическую избыточность, и не имеет целью уменьшение или увеличение производительности работы или же уменьшение или увеличение физического объёма базы данных. Конечной целью нормализации является уменьшение потенциальной противоречивости, хранимой в базе данных информации.

Устранение избыточности производится, как правило, за счёт декомпозиции отношений таким образом, чтобы в каждом отношении хранились только первичные факты (то есть факты, не выводимые из других хранимых фактов).

***Первая нормальная форма (1NF)***

Переменная отношения находится в первой нормальной форме (1НФ) тогда и только тогда, когда в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов.

***Вторая нормальная форма (2NF)***

Переменная отношения находится во второй нормальной форме тогда и только тогда, когда она находится в первой нормальной форме и каждый не ключевой атрибут неприводимо (функционально полно) зависит от её потенциального ключа. Функционально полная зависимость означает, что если потенциальный ключ является составным, то атрибут зависит от всего ключа и не зависит от его частей.

***Третья нормальная форма (3NF)***

Переменная отношения находится в третьей нормальной форме тогда и только тогда, когда она находится во второй нормальной форме, и отсутствуют транзитивные функциональные зависимости не ключевых атрибутов от ключевых.

При проектировании БД третья нормальная форма схем отношений достаточна в большинстве случаев, и приведением к третьей нормальной форме процесс проектирования реляционной базы данных обычно заканчивается. Однако иногда полезно продолжить процесс нормализации.

# ЛОГИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Логическая модель базы данных — схема базы данных, выраженная в понятиях модели данных. Этим отличается от концептуальной модели, описывающей семантику предметной области без указания технологии (конкретных методов реализации), и от физической модели, которая описывает конкретные физические механизмы, применяемые для хранения данных в накопителях.

Используя нашу концептуальную модель данных и выполнив нормализацию данных до 3 нормальной формы включительно, построим логическую модель данных, в которой мы можем выделить новые атрибуты, поля являющиеся первичными и внешними ключами, необходимые для построения связей.

Получившиеся логическая модель данных представлена на рисунке 2.

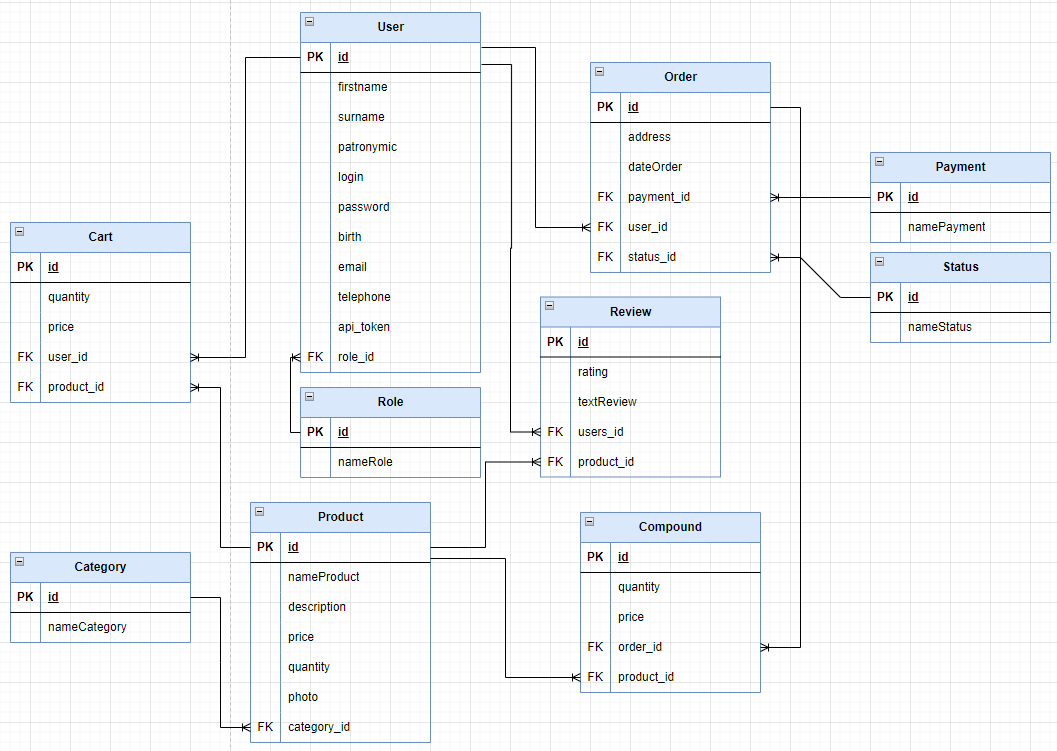


Рисунок 2 - Логическая модель данных

# ФИЗИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Физическая модель данных – это модель данных, описанная с помощью средств конкретной системы управления базами данных (СУБД). Физическая модель данных строится на базе даталогической путем добавления особенностей конкретной СУБД. К таким особенностям могут относиться поддерживаемые СУБД типы данных, соглашения о присвоении имен таблицам, атрибутам и т.д. Физическая модель данных фактически является готовым заданием на создание БД, имея которое можно реализовать БД в выбранной СУБД.

На рисунке 3 представленная физическая модель данных для информационной системы «Информационная система аниме интернет-магазина»

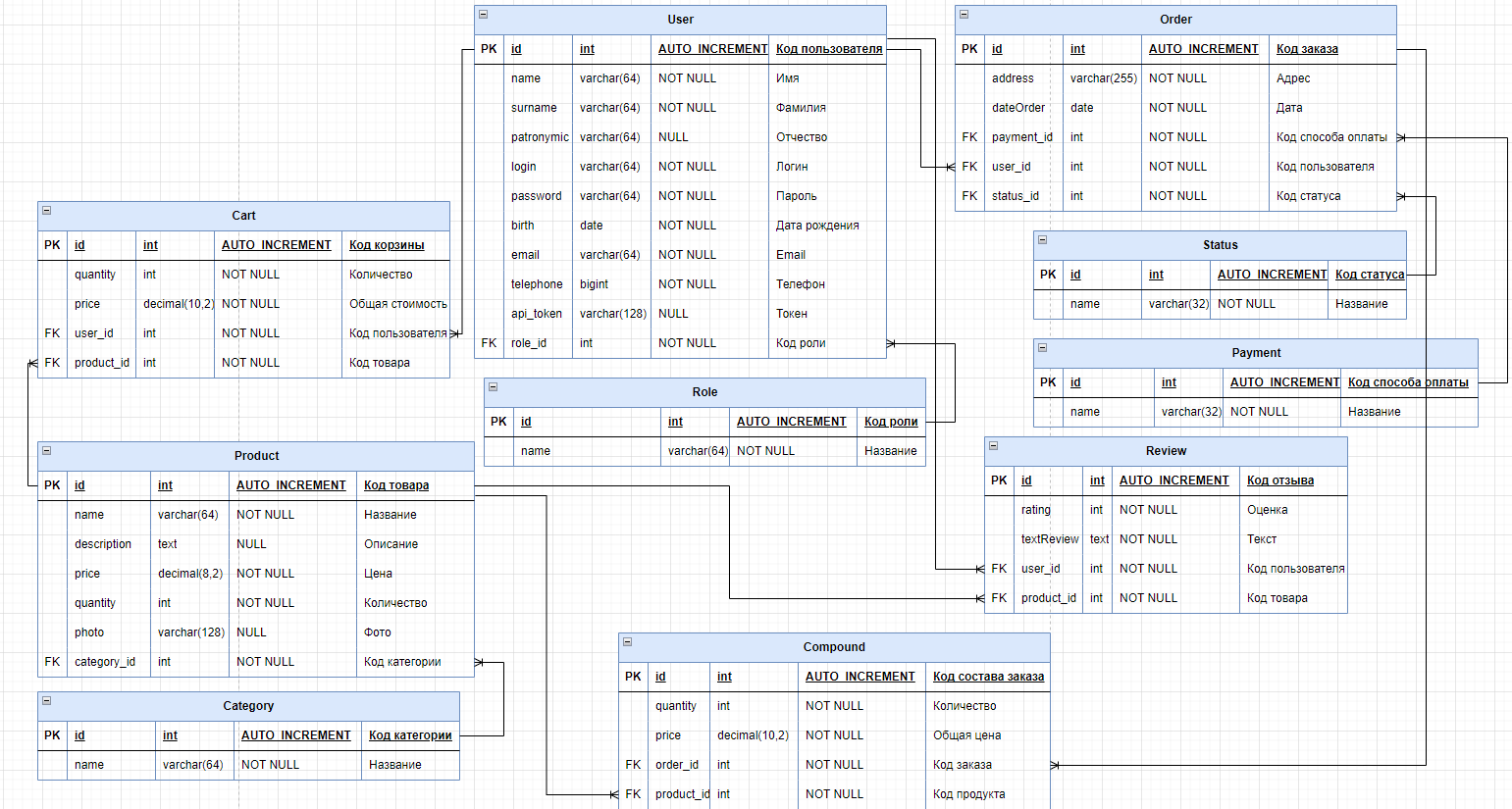


Рисунок 3 - Физическая модель данных

# СЛОВАРЬ ДАННЫХ

Таблица 1 – Словарь данных для БД «Информационная система аниме интернет-магазина»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Role** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID роли |
| name | varchar(64) | | Да | |  | Название |
| **User** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID пользователя |
| name | varchar(64) | | Да | |  | Имя |
| surname | varchar(64) | | Да | |  | Фамилия |
| patronymic | varchar(64) | | Нет | |  | Отчество |
| login | varchar(64) | | Да | |  | Логин |
| password | varchar(64) | | Да | |  | Пароль |
| birth | date | | Да | |  | Дата рождения |
| email | email | | Да | |  | Email |
| telephone | bigint | | Да | |  | Телефон |
| token | int | | Да | |  | Токен |
| role\_id | int | | Да | | Внешний | ID роли |
| **Order** | | | | | | |
| **Поле** | **Формат** | **Обязательное** | | **Ключ** | | **Примечание** |
| id | int | Да | | Первичный | | ID заказа |
| address | varchar(255) | Да | |  | | Адрес |
| dateOrder | date | Да | |  | | Дата |
| discount\_id | int | Да | | Внешний | | ID скидки |
| payment\_id | int | Да | | Внешний | | ID способа оплаты |
| user\_id | int | Да | | Внешний | | ID пользователя |
| status\_id | int | Да | | Внешний | | ID статуса |
| **Review** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID отзыва |
| textReview | text | | Да | |  | Текст |
| user\_id | int | | Да | | Внешний | ID пользователя |
| product\_id | int | | Да | | Внешний | ID товара |
| **Payment** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID способа оплаты |
| name | varchar(32) | | Да | |  | Название |
| **Compound** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID состава заказа |
| quantity | int | | Да | |  | ID количества |
| price | decimal(10, 2) | | Да | |  | ID общей цены |
| order\_id | int | | Да | | Внешний | ID заказа |
| product\_id | int | | Да | | Внешний | ID товара |
| **Product** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID товара |
| name | varchar(64) | | Да | |  | Название |
| description | text | | Нет | |  | Описание |
| price | decimal(8,2) | | Да | |  | Цена |
| quantity | int | | Да | |  | Количество |
| photo | image | | Нет | |  | Фото |
| category\_id | int | | Да | | Внешний | ID категории |
| **Category** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID категории |
| name | varchar(164) | | Да | |  | Название |
| **Cart** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID корзины |
| quantity | int | | Да | |  | Количество |
| price | decimal(10,2) | | Да | |  | Общая стоимость |
| user\_id | int | | Да | | Внешний | ID пользователя |
| product\_id | int | | Да | | Внешний | ID товара |
| **Status** | | | | | | |
| **Поле** | **Формат** | | **Обязательное** | | **Ключ** | **Примечание** |
| id | int | | Да | | Первичный | ID статуса |
| name | varchar(32) | | Да | |  | Название |

Глава 3.

Введение

Проект "Название проекта" представляет собой инновационное веб-приложение, разработанное для облегчения процесса онлайн-шопинга и управления заказами. Объединяя в себе мощь фреймворка Laravel для бэкенда и гибкость Vue.js для фронтенда, это приложение стало надежным инструментом как для пользователей, так и для администраторов.

Основные функции

Регистрация и вход:

Простой и интуитивно понятный процесс регистрации и входа позволяет пользователям быстро начать пользоваться приложением.

Безопасная аутентификация обеспечивает конфиденциальность данных пользователей.

Управление товарами:

Администраторы имеют возможность добавлять, редактировать и удалять товары с легкостью.

Пользователи могут просматривать описания товаров, а также оставлять отзывы и оценки.

Оформление заказов:

Интуитивно понятный интерфейс помогает пользователям легко и быстро оформить заказ, указав адрес доставки и способ оплаты.

Система уведомлений позволяет отслеживать статус заказов и получать информацию о доставке.

Управление пользователями:

Администраторы имеют возможность управлять пользователями, блокируя или удаляя аккаунты по необходимости.

Пользователи могут обновлять свои профили, изменяя информацию о себе и настройки аккаунта.

Технологии

Laravel:

Фреймворк Laravel обеспечивает мощный и гибкий инструментарий для разработки серверной части приложения.

Модульная структура и множество готовых компонентов упрощают процесс создания и поддержки функционала.

Vue.js:

Фронтенд приложения построен с использованием Vue.js, что обеспечивает быстрый и отзывчивый интерфейс для пользователей.

Компонентный подход позволяет создавать многоразовые элементы интерфейса и улучшает масштабируемость проекта.

phpMyAdmin:

В качестве СУБД используется phpMyAdmin, обеспечивающий удобное администрирование базы данных MySQL.

Установка и настройка

Клонирование репозитория:

Склонируйте репозиторий проекта с GitHub на ваш локальный компьютер.

Установка зависимостей:

Установите все необходимые зависимости, выполнив команды composer install для Laravel и npm install для Vue.js.

Настройка окружения:

Создайте файл .env на основе .env.example и настройте параметры подключения к базе данных и другие конфигурационные параметры.

Запуск сервера:

Запустите локальный сервер Laravel, выполнив команду php artisan serve.

Для разработки клиентской части выполните сборку Vue.js с помощью команды npm run watch.

Заключение

Проект "Название проекта" представляет собой мощный инструмент для управления заказами и улучшения процесса онлайн-шопинга. Используйте его функционал для создания неповторимого опыта покупок!