**Техническое задание**

**Модели данных**

**User поля:**

id - integer(> 0), unique;

email - varchar(255), unique;

password - varchar(255) ;

role - varchar(255) ;

**Cart поля:**

id - integer(> 0), unique;

userId - foreign key to User model;

**CartGadget поля:**

id - integer(> 0), unique;

gadgetId - foreign key to Gadget model;

cartId - foreign key to Cart model;

**Gaget поля:**

id - integer(> 0), unique;

name - varchar(255), unique;

price - varchar(255);

img - varchar(255), unique;

categoryId - foreign key to Category model;

brandId - foreign key to Brand model;

**GadgetProperty поля:**

id - integer(> 0), unique;

gadgetId - foreign key to Gadget model;

title - varchar(255);

description - varchar(255);

**Brand поля:**

id - integer(> 0), unique;

name - varchar(255), unique;

**Category поля:**

id - integer(> 0), unique;

name - varchar(255), unique;

**Роли**

**Гость. Возможности гостя:**

Создать аккаунт;

Просмотр списка товаров на странице товаров;

Просмотр товара на странице товара.

**Пользователь. Возможности пользователя:**

Все, что может гость;

Авторизироваться;

Выйти из аккаунта;

Имеет возможность добавить товар в понравившееся;

Добавлять товары в понравившееся и удалять их от туда;

**Админ. Возможности админа:**

Все, что может пользователь;

Добавлять товары, категории, бренды.

**Описание API методов:**

**POST("user/login")** - метод производит вход пользователя. Принимает тело с полями email и password. Если в теле запроса отсутствует одно из полей, указанного пользователя не существует или пароль не верный, то возвращается ошибка с соответсвтующим сообщением. В противном случае возвращается jwt токен.

**POST("user/registration")** - метод производит регистрацию пользователя. Принимает тело с полями email, password, role. Если в теле запроса отсутствует одно из полей или указанный пользователь уже существует, то возвращается ошибка с соответсвтующим сообщением.

**GET("user/auth")** - метод производит проверку на то, авторизован ли пользователь или нет. Проверка производится по JWT токену из хедера. В случае если токен валиден, то метод обновит токен, в противно случе вернется ошибка.

**POST("/gadget")** - есть проверка на администратора - метод создает товар. Принимает тело с полями name, price, brandId, categoryId и необязательное поле с массивом properties. Также получает фотографию. В случае отсутстивая полей или фотографии возвращает ошибку, как и в случае если такой товар уже существует. В случае успеха возвращает сам товар.

**GET("/gadget")** - метод возвращает список товаров. Может принимать как query параметры brandId, categoryId, limit и page. Возвращает количество гаджетов и их список.

**GET("/gadget/:id")** - метод возвращает товар с его свойствами по id.

**POST("/category")** - есть проверка на админитратора - метод создает категорию. Принимает тело с полем name. Если такого поля нет возвращает ошибку, как и в случае если такая категория уже существует. Возвращает созданную категорию.

**GET("/category")** - возвращает все категории.

**POST("/brand")** - есть проверка на администратора - метод создает бренд. Принимает тело с полем name. Если такого поле нет возвращает ошибку, как и в случае если такой бренд уже существует. Возвращает созданный бренд.

**GET("/brand")** - возвращает все бренды.

**POST('/cart')** - есть проверка на авторизацию - метод добавляет товар в избранное. Принимает тело с полем gadgetId. Если такого поле нет возвращает ошибку, как и в случае если такой това уже есть в корзине или пользователь не авторизован. Возвращает добавленый товар.

**GET('/cart')** - есть проверка на авторизацию - метод возвращает список товаров в корщине пользьзователя. Может принимать как query параметры limit и page. Возвращает ошибку, если пользователь не авторизован. Возвращает количество гаджетов и их список в корзине.

**GET('/cart/delete/:id')** - есть проверка на авторизацию - метод удаляет товар из корзины пользователя по id. Возвращает сообщение, подтверждающее успешную операцию.