**ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА РАЗРАБОТКУ API СЕРВИСА «ПОХОРОННОЕ БЮРО»  
*ЭКЗАМЕН ПО ПМ.04***

*СПЕЦИАЛЬНОСТЬ 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ*

Выполнил:

Студент группы 1912с

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

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

Томск 2024

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc169536528)

[Информация о проекте 3](#_Toc169536529)

[**ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА РАЗРАБОТКУ API СЕРВИСА.** 4](#_Toc169536530)

[Общее описание проекта 4](#_Toc169536531)

[Общие требования 4](#_Toc169536532)

[Специфические требования к API 6](#_Toc169536533)

[**Регистрация** 6](#_Toc169536534)

[**Авторизация** 6](#_Toc169536535)

[**Просмотр товаров** 7](#_Toc169536536)

[**Просмотр отзывов к товару** 8](#_Toc169536537)

[**Выход** 8](#_Toc169536538)

[**Просмотр своего профиля** 9](#_Toc169536539)

[**Создание отзыва для купленного товара** 9](#_Toc169536540)

[**Создание категории** 10](#_Toc169536541)

[**Создание товара** 11](#_Toc169536542)

[**Редактирование категории** 12](#_Toc169536543)

[**Редактирование товара** 12](#_Toc169536544)

[**Удаление категории** 13](#_Toc169536545)

[**Удаление товара** 14](#_Toc169536546)

[**Просмотр всех пользователей** 14](#_Toc169536547)

# **ВВЕДЕНИЕ**

## Информация о проекте

|  |  |
| --- | --- |
| Название проекта | «Похоронное бюро» |
| Предметная область проекта | Похоронное бюро — это специализированная организация, предоставляющая комплексные услуги по организации и проведению похоронных церемоний, а также занимающаяся продажей ритуальных товаров. Основная задача такого бюро — помочь родственникам усопшего в трудный период, обеспечив проведение всех необходимых ритуалов и процедур, связанных с погребением, кремацией и другими видами прощания, а также предоставление всех необходимых атрибутов и предметов. |
| Цель проекта | Целью проекта заключается в разработке полнофункциональной веб-платформы для похоронного бюро, включая создание и проектирование API. |
| Задачи проекта | 1. Разработать технического задания API сервиса. 2. Разработать API сервис на фреймворке Laravel 10. 3. Разработать автоматизированные тесты и провести тестирование API сервиса. |

# 

# **ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА РАЗРАБОТКУ API СЕРВИСА.**

## Общее описание проекта

В базе данных сервиса похоронного бюро имеются следующие таблицы:

* Роль
* Пользователь
* Категория
* Товар
* Заказ
* Состав заказа
* Корзина
* Отзыв

Все пользователи системы подразделяются на две группы:

1. Администраторы;
2. Пользователи;

Функционал пользователя:

1. Регистрация
2. Авторизация
3. Просмотр товаров
4. Просмотр отзывов к товару

Функционал авторизированного пользователя:

1. Выход
2. Создание отзыва для купленного товара
3. Просмотр своего профиля

Функционал администратора:

1. Создание/редактирование/удаление категорий;
2. Создание/редактирование/удаление товаров
3. Просмотр всех пользователей

## Общие требования

Идентификация пользователя организуется посредством Bearer Token.

При попытке доступа к защищенным авторизацией функциям системы во всех запросах необходимо возвращать ответ следующего вида:

Status: 401

Content-Type: application/json

Body:

{  
 "error": {  
 "code": 403,  
 "message": "Ошибка авторизации"  
 }

}

При попытке доступа авторизованным пользователем к функциям недоступным для своей группы во всех запросах необходимо возвращать ответ следующего вида:

Status: 403

Content-Type: application/json

Body:

{  
 "error": {  
 "code": 403,  
 "message": "Доступ запрещён"  
 }  
}

В случае ошибок связанных с валидацией данных во всех запросах необходимо возвращать следующее тело ответа:

{  
 "error": {  
 "code": <code>,  
 "message": <message>,  
 "errors": {  
 <key>: [<error message>]  
 }  
 }  
}

Вместо <code> и <message> необходимо указывать соответствующее значение, определенное в описании ответа на соответствующий запрос. В свойстве error.errors необходимо перечислить те свойства, которые не прошли валидацию, а в их значениях указать массив с ошибками валидации.

В случае ошибок связанных с валидацией данных во всех запросах необходимо возвращать следующее тело ответа:

{  
 "error": {  
 "code": <code>,  
 "message": <message>,  
 "errors": {  
 <key>: [<error message>]  
 }  
 }  
}

Вместо <code> и <message> необходимо указывать соответствующее значение, определенное в описании ответа на соответствующий запрос. В свойстве error.errors необходимо перечислить те свойства, которые не прошли валидацию, а в их значениях указать массив с ошибками валидации.

## Специфические требования к API

### **Регистрация**

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

* name – обязательное поле, строка, максимум 128 символов;
* surname – обязательное поле, строка, максимум 128 символов;
* patronymic – не обязательное поле, строка, максимум 128 символов;
* login – обязательное и уникальное поле, строка, минимум 5 и максимум 128 символов;
* password – обязательное поле, строка, минимум 5 и максимум 255 символов;
* telephone – обязательное и уникальное поле, формат число, от 1 до 20 цифр;
* email – обязательное и уникальное поле, строка, максимум 128 символов;

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/register  **Method:** POST  **Headers:** Content-Type: application/json  **Body:**  {  "surname": "Usertest",  "name": " Usertest",  "patronymic": " Usertest",  "login": " Usertest",  "password": " Usertest1!",  "telephone": "7930573565",  "email": "usertest@mail.ru"  } | Успешная регистрация  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Регистрация прошла успешно"  }  }  Ошибки валидации полей  **Формат ответа из общих требований** |

### **Авторизация**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/login  **Method:** POST  **Headers:** Content-Type: application/json  **Body:**  {  "login": "log1",  "password": "pass1"  } | Успешная авторизация  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "api\_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."  }  }  Ошибка аутентификации  **Status:** 401  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 401,      "message": "Ошибка аутентификации"  }  Ошибки валидации полей  **Формат ответа из общих требований** |

### **Просмотр товаров**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/products  **Method:** GET  **Headers:**  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,         "name": "Берсерк. Том 1.",         "description": "",        "price": "1174.00",       "quantity": 10,        "photo": "uploads/1/1.webp",        "category\_id": 1,  },  // Другие товары  ]  }  Ответ при отсутствии товаров  **Status:** 404  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 404,      "message": "Товары не найдены"  } |

### **Просмотр отзывов к товару**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/product/{id}/review  **Method:** GET  **Headers:**  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": int,         "product\_id ": int,  "rating": int,  "description": "string",  "photo": "path",  },  // Другие отзывы  ]  }  Ответ при отсутствии отзывов  **Status:** 404  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 404,      "message": "Отзывы не найдены"  } |

### **Выход**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/logout  **Method:** GET  **Headers:** Authorization: Bearer {{api\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Вы успешно вышли из системы"  }  }  Ошибка авторизации  **Формат ответа из общих требований** |

### **Просмотр своего профиля**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/profile  **Method:** GET  **Headers:** Authorization: Bearer {{api\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "surname": "Usertest",  "name": " Usertest",  "patronymic": " Usertest",  "login": " Usertest",  "password": " Usertest1!",  "telephone": "7930573565",  "email": "usertest@mail.ru"  }  }  Ошибка авторизации  **Формат ответа из общих требований** |

### **Создание отзыва для купленного товара**

При отправке запроса необходимо передать FormData со следующими полями:

* rating – обязательное поле, числовой формат, от 1 до 5 цифр
* textReview – обязательное поле, строка, максимум 255 символов

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/product/{id}/review  **Method:** POST  **Headers:** Authorization: Bearer {{api\_token}},  Content-Type: application/json  **Body:**  {  "rating": 4,  "textReview": "Отличный товар!"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {      "message": "Отзыв успешно сохранен"  }  Ошибка авторизации  **Формат ответа из общих требований**  Ответ при попытке оставить отзыв на не купленный товар  **Status:** 404  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 404,      "message": "Вы не можете оставить отзыв на товар, который вы не покупали"  }  Ошибки валидации полей  **Формат ответа из общих требований** |

### **Создание 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}}/admin/category/create  **Method:** POST  **Headers:** Authorization: Bearer {{admin\_api\_token}},  Content-Type: application/json  **Body:**  {  "name": "Новая Категория"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Категория успешно создана"  }  }  Ответ при попытке добавить категорию с уже существующем именем  **Status:** 422  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 422,      "message": "Категория с таким именем уже существует"  }  Ошибки валидации полей  **Формат ответа из общих требований**  Ошибка авторизации  **Формат ответа из общих требований**  Ошибка доступа  **Формат ответа из общих требований** |

### **Создание товара**

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

* name – обязательное поле, строка, от 1 до 64 символов;
* description – обязательное поле, строка;
* price – обязательное поле, положительное вещественное число с 2 знаками после запятой;
* quantity – обязательное поле, числовой, минимум 1;
* path – необязательное поле, фото формата jpeg, png, jpg, webp;
* category\_id – обязательное поле, число, категория с данным идентификатором должна быть.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/admin/product/create  **Method:** POST  **Headers:** Authorization: Bearer {{admin\_api\_token}},  Content-Type: application/json  **Body:**  {    "name": "Новый Товар",    "price": 1500,    "quantity": 10,    "description": "new product",    "category\_id": 1  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Товар успешно создан"  }  }  Ответ при попытке добавить продукт с уже существующем именем  **Status:** 422  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 422,      "message": "Продукт с таким именем уже существует"  }  Ошибки валидации полей  **Формат ответа из общих требований**  Ошибка авторизации  **Формат ответа из общих требований**  Ошибка доступа  **Формат ответа из общих требований** |

### **Редактирование категории**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/admin/category/{id}/edit  **Method:** PATCH  **Headers:** Authorization: Bearer {{admin\_api\_token}}, Content-Type: application/json  **Body:**  {    "name": "Измененная манга"  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "Категория успешно отредактирована"  }  }  Ответ при ошибке  **Status:** 422  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 422,      "message": "Категория с таким именем уже существует"  }  Ошибки валидации полей  **Формат ответа из общих требований**  Ошибка авторизации  **Формат ответа из общих требований**  Ошибка доступа  **Формат ответа из общих требований** |

### **Редактирование товара**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/admin/product/{id}/edit  **Method:** POST  **Headers:** Authorization: Bearer {{admin\_api\_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."  }  }  Ответ при попытке изменить продукт с уже существующем именем  **Status:** 422  **Content-Type:** application/json  **Body:**  {      "success": false,      "code": 422,      "message": "Продукт с таким именем уже существует"  }  Ошибки валидации полей  **Формат ответа из общих требований**  Ошибка авторизации  **Формат ответа из общих требований**  Ошибка доступа  **Формат ответа из общих требований** |

### **Удаление категории**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/admin/category/{id}/delete  **Method:** DELETE  **Headers:** Authorization: Bearer {{admin\_api\_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."  }  }  Ошибка авторизации  **Формат ответа из общих требований**  Ошибка доступа  **Формат ответа из общих требований** |

### **Удаление товара**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/admin/product/{id}/delete  **Method:** DELETE  **Headers:** Authorization: Bearer {{admin\_api\_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."  }  }  Ошибка авторизации  **Формат ответа из общих требований**  Ошибка доступа  **Формат ответа из общих требований** |

### **Просмотр всех пользователей**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/users  **Method:** GET  **Headers:** Authorization: Bearer {{api\_token}}  **Body:**  {  } | Успешный ответ  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  //Данные всех пользователей  }  }  Ошибка авторизации  **Формат ответа из общих требований** |