Модуль Б. Разработка серверной части приложения

Стек: PHP Laravel, Python Django, MySQL

Введение

Вам необходимо разработать **серверную часть интернет-магазина**. Это включает в себя:

● разработку **административной панели** по предоставленному шаблону;

● реализацию **REST API** по предоставленной спецификации.

Функциональность административной панели реализуется с использованием HTML + PHP внутри выбранного фреймворка. API должен обеспечивать взаимодействие с пользовательской частью системы.

Админ-панель

Общие требования

Панель должна быть реализована на основе предоставленного HTML-шаблона.

Весь функционал доступен **только авторизованному администратору**.

При работе с формами необходимо реализовать **валидацию**:

○ Поля с ошибками должны быть визуально выделены.

○ Под каждым некорректным полем должно отображаться текстовое сообщение об ошибке.

Все пользователи

**Экран авторизации**

Экран авторизации содержит поля для ввода email и пароля, а также кнопку отправки формы.

При вводе некорректных данных должно отображаться соответствующее сообщение об ошибке.

Оба поля являются обязательными для заполнения. *Работает правило валидации форм в админ панели.*

После успешной авторизации администратора перенаправляет на список всех пользователей

Администратор должен иметь возможность управлять пользователями, а именно:

● Банить

● Разбанивать

● Просматривать пользователей с возможностью поиска по нику

**Просмотр списка пользователей** должен включать в пользователя поля с идентификатором, ником, почтой, датой регистрации, статусом ( Забанен, Не забанен ), количеством постов

В списке должны быть кнопки бана или разбана ( если забанен кнопка разбана и наоборот )

Также должна отображаться статистика с общим кол-во пользователей, кол-во не забаненых пользователей, кол-во забаненых

API

Описание

API должно предоставлять пользователю следующий функционал:

1. регистрация
2. авторизация
3. просмотр всех постов
4. создание/редактирование/удаление постов
5. возможность ставить и убирать лайки у постов

Общие требования

Тело запросов должно быть формате json при отправке и при получении.

Авторизация должна работать по заголовку Authorization с использованием Bearer-токена.

Если данного маршрута не существует, надо ответить:

Status code: 404

Body:

{

“message”: “Not Found”

}

Вам необходимо придерживаться одного шаблона, если пользователь попытался отправил не валидные данные:

Status code: 422

Body:

{

“message”: “Validation errors”,

“errors”: {

“{имя поля}”: [  
 “{текст ошибки}”

]

}

}

Попытка получения доступа к функционалу который пользователь не должен иметь доступа:

Status code: 403

{

“message”: “GET OUT!!”,

“error\_code”: “4444”,

}

Регистрация

Запрос для регистрации пользователя в системе. При отправке запроса необходимо передать объект с логином и паролем. Если клиент отправил корректные данные, то необходимо вернуть сгенерированный токен, а иначе сообщение об ошибке.

Валидация:

nickname – обязательное, строка, до 20 символов

email – обязательное, уникальное, валидный емайл

password –обязательное, минимум 3 символа

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/register;  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  “nickname”: “user123“,  “email“: “123 “,  “password”: “asd”,  } | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  “data”: {  “user”: {  “nickname”: “user123“,  “email“: “123 “,  }  }  }  Ошибки валидации полей  **Формат ответа из общих требований** |

Авторизация

Запрос для авторизации пользователя в системе. При отправке запроса необходимо передать объект с логином и паролем. Если клиент отправил корректные данные, то необходимо вернуть сгенерированный токен, а иначе сообщение об ошибке.

Валидация

1. email – обязательное
2. password – обязательное

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of /login;  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  “email”: “user123“,  “password“: “123 “  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  “credentials”: {  “token”: <сгенерированный token>  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Неправильные логин или пароль  **Status:** 401  **Content-Type:** application/json  **Body:**  {  “message”: “failed”  } |

**Просмотр списка всех постов**

Запрос предназначен для получения списка всех постов в общей ленте

Посты отображаются только от не забаненных пользователей

Также должна быть реализована пагинация по 10 постов на странице и возможность выбора через query page

Поиск по названию поста с премерным совпадением query search

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/posts  **Method:** GET | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "title": "Bell",  "description": "admin",  "img": <url>,  “liked\_it”: false,  “count\_likes”: 7,  “post\_maker\_id”: 10,  “created\_at”: “2025-07-24”  },  {  "id": 2,  "title": "Bell",  "description": "admin",  "img": <url>,  “liked\_it”: true,  “count\_likes”: 2,  “post\_maker\_id”: 4,  “created\_at”: “2025-07-24”  },  ]  } |

**Просмотр информации пользователя**

Запрос предназначен для получения аккаунта пользователя со всеми его постами с возможностью пагинации по 10 постов на страницу   
query параметр – page

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/user/{user\_id}  **Method:** GET | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  “nickname”: “asd”,  “posts”: {  [  {  "id": 1,  "title": "Bell",  "description": "admin",  "img": <url>,  “liked\_it”: true,  “count\_likes”: 7,  “post\_maker\_id”: 5,  “created\_at”: “2025-07-24”  },  {  "id": 2,  "title": "Bell",  "description": "admin",  "img": <url>,  “liked\_it”: false,  “count\_likes”: 3,  “post\_maker\_id”: 5,  “created\_at”: “2025-07-24”    },  ]  }  }  Запрос осуществляется на забаненного юзера  **Status:** 404  **Content-Type:** application/json  **Body:**  {  “message”: “User has been banned”  }  Запрос осуществляется на юзера у которого роль админа  **Status:** 404  **Content-Type:** application/json  **Body:**  {  “message”: “Not Found”  } |

**Роуты которые доступны только авторизованным пользователям  
у всех не авторизованных должна быть ошибка из общих требований**

**Выход**

Запрос предназначен для очистки значение токена пользователя.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of /logout  **Method:** GET | Успех  **Status:** 204  **Content-Type:** application/json |

**Создание поста**

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

● title – обязательное поле, строка, мин 3 символа;

● description – не обязательное поле, строка, мин 10 символов;

● img – не обязательное поле, изображение, форматы jpg,jpeg,png размером до 4.5мб;

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/posts  **Method:** POST    Headers  - Content-Type:  formData    Body:  {  “title”: “title“,  “description“: “descrip“,  “img”: <img>  } | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "title": "Bell",  "description": "admin",  "img": <url>,  “liked\_it”: false,  “count\_likes”: 3,  “post\_maker\_id”: 10,  “created\_at”: “2025-07-12”  }  }  Ошибки валидации полей  **Формат ответа из общих требований** |

**Редактирование поста**

Валидация такая же как и при создании

При замене фотки надо старую удалять

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/posts/{post\_id}  **Method:** PATCH    **Headers**  - Content-Type: application/json    **- Content-Type: application/json**    **Body:**  **{**  **“title”: “title“,**  **“description“: “descrip“,**  **“img”: <img>**  **}** | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "title": "Bell",  "description": "admin",  "img": <url>,  “count\_likes”: 0,  “post\_maker\_id”: 10,  “created\_at”: “2025-07-12”  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка редактирования чужого поста  **Формат ответа из общих требований со status кодом 403** |

**Удаление поста**

При удалении поста должна удаляться фотка из файлов

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/posts/{post\_id}  **Method:** DELETE    **Headers**  - Content-Type: application/json | Успех  **Status:** 204  **Content-Type:** application/json  Попытка удаления чужого поста  **Формат ответа из общих требований со status кодом 403** |

**Поставить лайк на пост**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/posts/{post\_id}/like  **Method:** POST    **Headers**  - Content-Type: application/json | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  “message”: “success”  }  }  Попытка поставить лайк, когда он уже стоит  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  “message”: “There’s already a like”  }  } |

**Убрать лайк с поста**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-of/posts/{post\_id}/like  **Method:** DELETE    **Headers**  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  “message”: “success”  }  }  Попытка убрать лайк, когда он не стоит  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  “message”: “no likes”  }  } |

**Инструкция для конкурсанта**

Разработанное API должно быть доступно по адресу http://{your\_url}/api-of

Форматы запросов и ответов, а также форматы дат и времени должен соответствовать примерам из задания.

Вы должны создать аккаунт админа с данными

Email: admin@admin

Password: admin1

Если мы не сможем зайти в админ панель то она не будет проверяться (вранье)

Кто на питоне напишет - расстрел