WEB-ДИЗАЙН И РАЗРАБОТКА

VII Региональный чемпионат "Молодые профессионалы" (WorldSkills Russia) Республики Мордовия

День 1. Модуль RESTful API

СОДЕРЖАНИЕ

Данный тестовый проект состоит из следующих файлов:

1. Задание.pdf

ВВЕДЕНИЕ

К вам обратился представитель кастинг-агентства с просьбой вступить в команду, которая разрабатывает для него блог. Блог должен стать своеобразным информационным посредником между агентством и клиентами – зрителями будущих различных передач и ток-шоу.

От вас требуется написать RESTful API для блога, всем остальным будет заниматься другие разработчики из вашей команды.

Функционал блога будет разделен на две роли:

- администратор
- гость

Функциональные возможности гостя:

- авторизация
- просмотр записей блога и комментариев
- комментирование записей блога
- поиск записей по тегу

Функциональные возможности администратора включают в себя функциональные возможности гостя, а также:

- создание, редактирование и удаление записей блога
- удаление комментариев к записи блога

Данное задание состоит из одного модуля и рассчитано на 3 часа. Распределите свое время таким образом, чтобы успеть выполнить все поставленные задачи.

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

API авторизации должен быть доступен POST-запросом на адрес `{API}/auth`, а также принимать следующие обязательные параметры:

- `login`
  логин администратора
- `password`
  пароль администратора

При успешной авторизации должен возвращаться ответ в следующем виде:

- status code: 200
- status text: Successful authorization
- body:
  - status
    true
  - token
    bearer-токен для использования API администратора

При безуспешной авторизации или ошибке в запросе должен возвращаться ответ в следующем виде:

- status code: 401
- status text: Invalid authorization data
- body:

- `status`
  false
- `message`
  Invalid authorization data

API создания постов должен быть доступен только **администратору** POST-запросом на адрес `{API}/posts`, а также принимать следующие обязательные параметры:

- `title`
  название поста, не пустой, уникальный
- `anons`
  анонс поста, не пустой
- `text`
  текст поста, не пустой
- `tags`
  тэги поста через запятую, не обязательное поле
- `image`
  изображение поста, не пустой, разрешенные форматы: jpg, png. Максимальный размер: 2 мегабайта

При успешном создании поста должен возвращаться ответ в следующем виде:

- status code: 201
- status text: Successful creation
- body:
  - `status`
    true
  - `post_id`
    уникальный идентификатор поста

После создания поста изображение должно загружаться на сервер в папку `{API}/post_images`.

При безуспешном добавлении поста должен возвращаться ответ в следующем виде:

- status code: 400
- status text: Creating error
- body:
  - `status`
    false
  - `message`
    ассоциативный массив из параметров, которые содержат ошибку. В значении должно быть описание ошибки

Пример ответа при безуспешном добавлении поста:

```
{
"status": false,
"message": {
      "title": "already exists",
      "image": "invalid file format"
      }
}
```

API редактирования постов должен быть доступен только **администратору** POST-запросом на адрес `{API}/posts/<POST_ID>`, а также принимать следующие параметры:

- `title`
  название поста, уникальный
- `anons`
  анонс поста
- `text`
  текст поста
- `tags`
  тэги поста через запятую
- `image`
  изображение поста, разрешенные форматы: jpg, png. Максимальный размер: 2 мегабайта

При успешном редактировании поста должен возвращаться ответ в следующем виде:

- status code: 201
- status text: Successful creation
- body:
  - `status`
    true
  - `post`
    - `title`
      название поста
    - `datatime`
      дата и время создания поста в формате чч:мм дд.мм.гггг (12:35 06.08.2018)
    - `anons`
      анонс поста
    - `text`
      текст поста
    - `tags`
      массив из тэгов поста ([“tag1”, “tag2”])
    - `image`
      ссылка на изображение поста

При безуспешном добавлении поста должен возвращаться ответ в следующем виде:

- status code: 400
- status text: Editing error
- body:
  - `status`
    false
  - `message`
    ассоциативный массив из параметров, которые содержат ошибку. В значении должно быть описание ошибки

При попытке редактирования несуществующего поста должен возвращаться ответ в следующем виде:

- status code: 404
- status text: Post not found
- body:
  - `message`
    Post not found

API удаления поста должен быть доступен только **администратору** DELETE-запросом на адрес `{API}/posts/<POST_ID>`.

При успешном удалении поста должен возвращаться ответ в следующем виде:

- status code: 201
- status text: Successful delete
- body:
  - `status`
    true

При попытке удаления несуществующего поста должен возвращаться ответ в следующем виде:

- status code: 404
- status text: Post not found
- body:
  - `message`
    Post not found

API просмотра записей блога должен быть доступен GET-запросом на адрес `{API}/posts`. Ответ должен содержать массив объектов постов, содержащие следующие параметры:

- `title`
  название поста
- `datatime`
  дата и время создания поста в формате чч:мм дд.мм.гггг (12:35 06.08.2018)
- `anons`
  анонс поста
- `text`
  текст поста
- `tags`
  массив из тэгов поста ([“tag1”, “tag2”])
- `image`
  ссылка на изображение поста

Ответ должен иметь status code “200” и status text “List posts”.

API просмотра одной записи блога должен быть доступен GET-запросом на адрес `{API}/posts/<POST_ID>`.

Ответ должен содержать следующие параметры:

- `title`
  название поста
- `datatime`
  дата и время создания поста в формате чч:мм дд.мм.гггг (12:35 18.12.2018)
- `anons`
  анонс поста
- `text`
  текст поста
- `tags`
  массив из тэгов поста (`[“tag1”, “tag2”]`)
- `image`
  ссылка на изображение поста
- `comments`
  массив из объектов комментариев содержащие следующие параметры:
  - `comment_id`
    уникальный идентификатор комментария

    - `datatime`
      дата и время в формате чч:мм дд.мм.гггг (12:35 18.12.2018)
    - `author`
      имя автора, если автором комментария является администратор, то парметр содержит текст “admin”
    - `comment`
      текст комментария

Ответ должен иметь status code “200” и status text “View post”.

При попытке просмотра несуществующего поста должен возвращаться ответ в следующем виде:

- status code: 404
- status text: Post not found
- body:
    - `message`
      Post not found

API добавления комментария к посту должен быть доступен POST-запросом на адрес `{API}/posts/<POST_ID>/comments` и содержать следующие обязательные параметры:

- `author`
  имя комментатора, обязательное только для гостя
- `comment`
  обязательное, максимум 255 символов

При успешном добавлении комментария должен возвращаться ответ в следующем виде:

- status code: 201
- status text: Successful creation
- body:
    - `status`
      true

При безуспешном добавлении комментария должен возвращаться ответ в следующем виде:

- status code: 400
- status text: Creating error
- body:
    - `status`
      false
    - `message`
      ассоциативный массив из параметров, которые содержат ошибку. В значении должно быть описание ошибки

При попытке добавить комментарий к несуществующему посту должен возвращаться ответ в следующем виде:

- status code: 404
- status text: Post not found
- body:
    - `message`
      Post not found

API удаления комментария к посту должен быть доступен только **администратору** DELETE-запросом на адрес `{API}/posts/<POST_ID>/comments/<COMMENT_ID>`.

При успешном удалении комментария должен возвращаться ответ в следующем виде:

- status code: 201

- status text: Successful delete
- body:
    - `status`
      true

При попытке удаления комментария к несуществующему посту должен возвращаться ответ в следующем виде:

- status code: 404
- status text: Post not found
- body:
    - `message`
      Post not found

При попытке удаления несуществующего комментария должен возвращаться ответ в следующем виде:

- status code: 404
- status text: Comment not found
- body:
    - `message`
      Comment not found

API поиска постов по тегу должен быть доступен GET-запросом на адрес `{API}/posts/tag/<TAG_NAME>`.
В ответ должен возвращаться ответ в следующем виде:

- status code: 200
- status text: Found posts
- body
  массив из объектов постов, содержащие данный тэг:
    - `title`
      название поста
    - `datatime`
      дата и время создания поста в формате чч:мм дд.мм.гггг (12:35 06.08.2018)
    - `anons`
      анонс поста
    - `text`
      текст поста
    - `tags`
      массив из тэгов поста ([“tag1”, “tag2”])
    - `image`
      ссылка на изображение поста

Авторизация пользователя при использовании административных функциональных возможностей будет проверяться через bearer-token авторизацию. Т.е. к каждому такому запросу в headers будет дописываться bearer-token:

```
Authorization: Bearer xxxxxxxx
```

При безуспешной попытке пройти авторизацию по токену на любой из запросов, который должен быть доступен только администратору, должен возвращаться ответ в следующем виде:

- status code: 401
- status text: Unauthorized
- body:
    - `message`
      Unauthorized

Под {API} подразумевается адрес http://xxxxxx-m1.wsr.ru/api, где xxx - ваш логин.
Логин администратора `admin`, пароль - `saransk2018`.

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

Готовая система должна быть доступна по адресу: http://xxxxxx-m1.wsr.ru/

Вам доступны фреймворки Yii2 и Laravel для работы с этим модулем.

Все ответы должны приходить в JSON формате.

Во время выполнения конкурсного задания можно использовать Postman, предложенный в медиа данных.

СИСТЕМА ОЦЕНКИ

СЕКЦИЯ	КРИТЕРИЙ	СУДЕЙСКАЯ	ОБЪЕКТИВНАЯ	СУММА
A	Организация работы и управление	0.80	0.00	0.80
B	Коммуникация и навыки межличностного общения	1.00	0.00	1.00
C	Дизайн	0.00	0.00	0.00
D	Верстка	0.00	0.00	0.00
E	Программирование на стороне клиента	0.00	0.00	0.00
F	Программирование на стороне сервера	0.00	11.75	11.75
G	CMS	0.00	0.00	0.00
Всего		**1.80**	**11.75**	**13.55**