Лабораторная работа №36

Использование фреймворка для создания сайта

Цель: отработать на практике навыки разработки веб-приложения с использованием клиентских и серверных фреймворков.

Ход работы:

1. Выполнить задание в соответствии с указаниями.
2. Оформить отчет по лабораторной работе в соответствии с требованиями.

ВВЕДЕНИЕ

В колледже планируется открытие выставки «Digital Art», организатором которой стало фотосообщество Белгорода «Belgorod Digital». Принять участие в ней могут все желающие.

Выставка является подведением итогов жизни фотосообщества. На выставке будут выставлены работы, которые будут выбраны самими участниками сообщества. Данная выставка направлена на создание творческой среды для фотографов, и повышения уровня их работ.

Организация «Belgorod Digital» решила разработать фотосервис, который позволит людям регистрироваться в нем, загружать фотографии, модифицировать их и делиться ими с другими пользователями.

Вам необходимо разработать функциональное клиент-серверное приложение: клиентскую часть (верстка, дизайнерские решения, взаимодействие с серверной частью (RESTful API), а также RESTful API для сервиса.

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

На стороне клиента:

Документация с описанием API предоставлена вам в приложении к заданию.

В документации к API вместо {service} необходимо подставить localhost/photos/api

Ниже описан функционал, который вам необходимо реализовать.

Регистрация – любой человек должен иметь возможность зарегистрироваться в сервисе. При регистрации необходимо указать следующие обязательные поля:

- Имя
- Фамилия
- Номер телефона
- Пароль

Авторизация – для авторизации необходимо ввести номер телефона и пароль.

В случае неправильного заполнения данных в формах должны отображаться ошибки, которые вернул сервер.

Пользователь должен иметь возможность загрузить фотографию.

Редактирование фотографии – должна быть возможность изменить название для уже загруженной фотографии, а также обрезать фотографию.

Удаление фотографий – пользователь должен иметь возможность удалить уже загруженную фотографию. Также должна быть возможность удалить группу фотографий, отметив их.

Пользователи должны иметь возможность поделиться своими фотографиями с другими людьми. Для этого нужно реализовать следующий функционал:

1. Выбор фотографий, которыми нужно поделиться.
2. Поиск пользователей по Имени, Фамилии и (или) номеру телефона.
3. Выбор пользователей из поиска, с кем пользователь хочет поделиться фотографиями.

Пользователи должны иметь возможность увидеть фотографии, которыми он поделился с другими людьми. Для этого нужно реализовать следующий функционал:

1. Поиск пользователей по Имени, Фамилии и (или) номеру телефона.
2. Выбор одного пользователя из поиска.

3. Просмотр расшаренных ему фотографий.

Разработанный сервис должен быть удобным для использования на разных устройствах. Не забывайте об использовании анимации и микро-анимации для улучшения впечатления.

Вы можете использовать следующие библиотеки и фреймворки: jQuery, Vue.js, React, Angular, Tailwind, Bootstrap

На стороне сервера:

Вашей задачей является создание RESTful API для вышеописанного сервиса.

Заказчик хочет в дальнейшем использовать разработанное API совместно с другими сервисами. Поэтому при написании API вам необходимо разрешить кросс-доменные запросы (CORS) для обращения с другого домена.

Ниже описаны некоторые нюансы логики взаимодействия с API:

- При создании фото название по умолчанию должно быть «Untitled»
- Все загруженные фотографии сохраняются с уникальным именем файла на сервер в папку «localhost/photos»
- Измененная фотография на сервере конвертируется в png и сохраняется на сервере в ту же папку.
- Пользователь не может изменить, удалить и поделиться фотографией, если не является её владельцем.
- При поиске, пользователь не видит себя в списке.

Вы можете использовать следующие фреймворки: Laravel, Yii, Django.

ПРИЛОЖЕНИЕ

Все запросы отправляются с заголовком Content-Type: application/json, если в запросе не указан другой вариант.

Все передаваемые и получаемые параметры должны строго соответствовать тому, что описано ниже!

Регистрация

URL: {service}/signup

Method: POST

Data params:

- first_name - обязательное
- surname - обязательное
- phone - обязательное, уникальное, ровно 11 цифр, может быть с ведущими нулями
- password - обязательное

Success response:

- Code: 201 Created
- Content:
  - id - идентификатор созданного пользователя

Error Response:

- Code: 422 Unprocessable entity
- Content - объект, где ключи - это поля, которые не прошли валидацию, а их значения - текст ошибки валидации

Авторизация

URL: {service}/login

Method: POST

Data params:

- phone - обязательное
- password - обязательное

Success response

- Code: 200 OK
- Content:
  - token - сгенерированный токен для дальнейшего доступа к странице

Error Response (ошибка валидации):

- Code: 422 Unprocessable entity
- Content - объект, где ключи - это поля, которые не прошли валидацию, а их значения - текст ошибки валидации

Error Response (неверный логин или пароль):

- Code: 404 Not found
- Content:
  - login: “Incorrect login or password”

Все последующие запросы требуют авторизации с использованием Bearer-токена. Токен должен быть отправлен в заголовке Authorization.

При отправке запроса без заголовка авторизации сервер должен вернуть следующий ответ:

Error Response:

- Code: 403 Forbidden
- Content:
  - message: You need authorization

Выход

URL: {service}/logout
Method: POST
Success response

- Code: 200 OK

Загрузка фотографии

URL: {service}/photo
Method: POST
Content-Type: **FormData**
Data params:

- photo – обязательное, файл с изображением, только jpg, jpeg или png.

Success response

- Code: 201 Created
- Content:
  - id – уникальный идентификатор фотографии
  - name – название фотографии, по умолчанию “Untitled”
  - url – **абсолютная** ссылка на фотографию (с http://)

Error Response:

- Code: 422 Unprocessable entity
- Content - объект, где ключи - это поля, которые не прошли валидацию, а их значения - текст ошибки валидации

Изменение фотографии

URL: {service}/photo/{ID}
Method: POST

Data params:

- **_method: обязательное поле, со значением “patch”, без кавычек**
- name – необязательное
- photo – необязательное, измененная фотография в формате base64

Success response

- Code: 200 OK
- Content:
  - id – уникальный идентификатор фотографии
  - name – название фотографии
  - url – **абсолютная** ссылка на фотографию (с http://)

Error Response (ошибка валидации):

- Code: 422 Unprocessable entity
- Content - объект, где ключи - это поля, которые не прошли валидацию, а их значения - текст ошибки валидации

Error Response (Ошибка доступа) - В случае, если пользователь пытается изменить не свою фотографию, ему возвращаются следующие параметры:

- Code: 403 Forbidden

Получение фотографий

URL: {service}/photo

Method: GET

Success response

- Code: 200 OK
- Content: массив объектов, где каждый объект имеет следующие свойства:
  - id
  - name
  - url
  - owner_id – id пользователя, которому принадлежит фотография
  - users – массив с id пользователей, которые имеют доступ к этой фотографии

Получение одной фотографии

URL: {service}/photo/{ID}

Method: GET

Success response

- Code: 200 OK
- Content:
  - id
  - name
  - url
  - owner_id – id пользователя, которому принадлежит фотография
  - users – массив с id пользователей, которые имеют доступ к этой фотографии

Удаление фотографии

URL: {service}/photo/{ID}

Method: DELETE

Success response:

- Code: 204 Deleted

Error Response (Ошибка доступа) - В случае, если пользователь пытается изменить не свою фотографию, ему возвращаются следующие параметры:

- Code: 403 Forbidden

Шаринг фотографий
URL: {service}/user/{ID}/share
Method: POST
Data params:

- photos – массив с идентификаторами фотографий. В случае, если в массиве будет id фотографии которая уже была расшарена, то повторно она расшарена не будет.

Success response:

- Code: 201 Created
- Content:
  - existing_photos - массив с идентификаторами фотографий, которые уже есть на сервере.

Поиск пользователей
URL: {service}/user
Method: GET
Query parameters:

- search – строка запроса, в которой указывается имя(или часть имени) фамилия(или часть фамилии) и (или) номер телефона(или часть номера телефона). Например: Иван Иванов 7951, И Иван 7.

Success response:

- Code: 200 OK
- Content - массив объектов, которые содержат идентификатор, имя, фамилию и телефон пользователя:
  - id
  - first_name
  - surname
  - phone