ВВЕДЕНИЕ

Технологии этого модуля: Клиентское программирование
Время на выполнение: 3 часа

К вам обратилась организация, оказывающая помощь в поиске потерянных домашних животных с просьбой разработать сервис GET PET BACK, на котором пользователи могут оставлять информацию о найденных животных, а владельцы, у которых пропал питомец, могут их искать.

Заказчик предоставляет вам полностью готовую верстку со всеми страницами и рабочее API. Вам необходимо использовать все имеющиеся навыки в клиентской разработке для создания Single Page Application, далее SPA.

Заказчик хочет, чтобы api можно было легко поддерживать, поэтому использование JavaScript фреймворков будет плюсом.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

Ваша задача – реализовать SPA приложение, которое будет работать с уже разработанным API.

Для вашего удобства во всех URL будет использоваться переменная {host} которая обозначает хост адрес API: http://??????.wsr.ru/api – где ?????? Ваш логин.

Для тестирования работы запросов, требующих авторизации пользователя используйте:

Телефон: 89111234567

Пароль: Password123

Ваше SPA должно состоять из следующих экранов (сверстанные страницы предоставляются):

- Главная
- Страница входа в личный кабинет
- Страница регистрации в личном кабинете
- Страница личного кабинета пользователя
- Страница добавления информации о найденном животном
- Страница поиска животных
- Страница с карточкой найденного животного

Приложение должно обладать следующим функционалом:

1. Главная

Слайдер с объявлениями с животными, у которых были найдены хозяева, должен работать корректно. До загрузки объявлений вместо слайдера отображается прелоадер. В случае отсутствия найденных хозяев блок не отображается.

Request	Response

URL: {host}/api/pets/slider
Method: GET
 	
Для получения пустого массива вы можете воспользоваться запросом
 	
URL: {host}/api/pets/slider/empty
Method: GET
	----------Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“pets”: [
	{
	“id”: 1,
	“kind”: “Кошка”,
	“description”: “Найдена маленькая кошечка”,
	“image”: “url”
	}
	]
	}
	}

	---------- No results ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“ pets”: []
	}
	}

Быстрый поиск по объявлениям

Поиск осуществляется по объявлениям по полю description. Возвращаются объявления, в описании которых присутствует переданное строковое значение.

При вводе больше 3-х символов должны появится подсказки по вводу, загруженные с сервера.

Также необходимо реализовать функцию debounce (запрос на сервер уходит с задержкой 1000ms после окончания ввода последнего символа для поиска)

Request	Response

URL: {host}/api/search
Method: GET
 	
Query string (GET parameters):
- query
 	
1. Без параметров - вернет все объявления.
2. ?query=кошка
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“orders”: [
	{
	“id”: 1,
	“kind”: “кошка”,
	“photos”: ’*{url}*/img1.png’,
	“description”: “Найдена кошка, порода Сфинкс, очень грустная”,
	“mark”: “VL-0214”,
	“district”: “Василеостровский”,
	“date”: “01-01-1970”,
	}
	]
	}
	}
	
	---------- No results ----------
	Status: ~~204~~ 200
	Content-Type: application/json
	Body:

	{
	“data”: {
	“ orders”: []
	}
	}

mark - клеймо

Если результатов нет, появляется соответствующее сообщение.

Карточки найденных животных

Выводится информация о 6 найденных животных, найденных в последнее время.
Данные в блоке выводятся с сортировкой по убыванию даты.

Request	Response

URL: {host}/api/pets
Method: GET
 	
Query string (GET parameters):
Headers
- Content-Type: application/json
 	
Body:
{
}
	---------- **Successful** ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“orders”: [
	{
	“id”: 1,
	“phone”: “89112345678”,
	“name”: “Иван”,
	“kind”: “кошка”,
	“photo”: ‘*{url}*/img1.png’,
	“description”: “Найдена кошка, порода Сфинкс, очень грустная”,
	“mark”: “VL-0214”,
	“district”: “Василеостровский”,
	“date”: “01-01-1970”,
	“registred”: true
	}
	]
	}
	}

registred - true, если информация о найденном животном была добавлена зарегистрированным пользователем, иначе - false

Подписка на новости

Форма для подписки на новости, в случае успешной подписки, заменяется на сообщение о успешной подписке, в случае ошибки появляется сообщение об ошибке с текстом ошибки.

email: обязательное

Request	Response

URL: {host}/api/subscription
Method: POST
Body:
{
"email": “email”
}
	---------- **Successful** ----------
	Status: ~~200~~ 204
	---------- **Validation error** ----------
	Status: 422
	Content-Type: application/json
	Body:
	{
	“error”: {
	“code”: 422,
	“message”: “Validation error”,
	“errors”: {
	<key>: <массив ошибок>
	}
	}
	}

Регистрация

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

- name – обязательное поле, строка, допустимые символы - кириллица, пробел, дефис
- phone – обязательное, строка, только цифры, и знак +
- email – обязательное поле, строка, формат - email
- password – обязательное поле, строка, не менее 7 символов, обязательно: 1 цифра, 1 строчная, 1 заглавная буквы
- password_confirmation – обязательное поле, строка, значение должно совпадать со значением password
- confirm – обязательное поле, integer 0 или 1 - для элемента, подтверждающего согласие пользователя на обработку персональных данных. Если нет подтверждения согласия на обработку персональных данных, то необходимо вывести соответствующее сообщение в массиве ошибок

Вся валидация проводится средствами JS!

Отправка некорректных данных на сервер невозможна. В случае успешной регистрации отображается соответствующее сообщение, в случае ошибки появляется сообщение с текстом ошибки.

Request	Response
URL: {host}/api/register **Method:** POST **Headers** **- Content-Type:** application/json	---------- **Successful** ---------- **Status:** 204 ---------- **Validation error** ---------- **Status:** 422 **Content-Type:** application/json

Body:
{
“name”: “Иван”,
“phone”: “89001234567”,
“email”: “user@user.ru”
“password”: “paSSword1”,
“password_confirmation”: paSSword1”,
“confirm”: 1,
}
	Body:
	{
	“error”: {
	“code”: 422,
	“message”: “Validation error”,
	“errors”: {
	<key>: <массив ошибок>
	}
	}
	}

Страница аутентификации

Запрос для аутентификации пользователя в системе. При отправке запроса необходимо передать объект с логином и паролем.

- email – обязательное
- password – обязательное

Отправка некорректных данных на сервер невозможна. В случае успешной регистрации отображается соответствующее сообщение, в случае ошибки появляется сообщение с текстом ошибки.

Request	Response

URL: {host}/api/login
Method: POST
 	
Headers
- Content-Type: application/json
 	
Body:
{
“email”: “user@user.ru”,
“password”: “paSSword1”
}
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“token”: <сгенерированный token>
	}
	}
	
	---------- Validation error ----------
	Status: 422
	Content-Type: application/json
	Body:
	{
	“error”: {
	“code”: 422,
	“message”: “Validation error”,
	“errors”: {
	<key>: <массив ошибок>
	}
	}
	}

	---------- **Unauthorized** ----------
	Status: 401
	Content-Type: application/json
	Body:
	{
	“error”: {
	“code”: 401,
	“message”: “Unauthorized”,
	“errors”: {
	“phone”: [“phone or password incorrect”]
	}
	email email
	}
	}

Страница поиска по объявлениям

Поиск осуществляется по объявлениям по переданным значениям: район, вид животного.
Должно быть передано на сервер хотя бы одно из указанных значений.
Поиск по полю “район” - строгий (полное соответствие, например, “Центральный”),
по полю “вид” - нестрогий (например, для “ко” должны отобразиться объявления, в которых указан вид “кошка”, “кот”, “котик”, “собачко” и т.д.)
Необходимо отобразить все объявления, в полях которых присутствуют введенные параметры.

Вам следует реализовать пагинацию для отображения результатов поиска, позволяющую просматривать другие результаты поиска - не более 10 животных на одной странице.

Если результатов поиска нет, необходимо отобразить соответствующую информацию.

Request	Response

URL: {host}/api/search/order/
Method: GET
 	
Query string (GET parameters):
- query (/?district=Василеостровский&kind=ко)
	Response
	---------- **Successful** ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“orders”: [
	{
	“id”: 1,
	“phone”: “89112345678”,
	“email”: “user@user.ru”,
	“name”: “Иван”,
	“kind”: “кошка”,
	“photos”: ‘*{url}*/img1.png’,
	“description”: “Найдена кошка, порода Сфинкс, очень грустная”,
	“mark”: “VL-0214”,

	“district”: “Василеостровский”,
	“date”: “01-01-1970”,
	}
	]
	}
	}
	
	---------- No results ----------
	Status: 204
	Content-Type: application/json
	Body:
	{
	“data”: {
	“ orders”: []
	}
	}

Личный кабинет

Должна быть возможность выйти из личного кабинета.

Необходимо позаботиться об уведомлении пользователей о каких-либо действиях (ошибки валидации, подтверждения и т.п).

В личный кабинет возможен вход только авторизованного пользователя. Если пользователь не авторизован, должна происходить переадресация на страницу с аутентификацией.

Функциональные возможности личного кабинета:

- *Информация о пользователе*

Только для авторизованного пользователя.
Количество дней, которое прошло с момента регистрации пользователя на сайте рассчитывается на стороне клиента относительно текущей даты ит полученного значения registrationDate
В случае запроса неавторизованным пользователем должна происходить переадресация на страницу с аутентификацией.
При запросе информации о другом пользователе необходимо отобразить сообщение об ошибке доступа.

Request	Response

URL: {host}/api/users/{id}
Method: GET
 	
Headers
- Content-Type: application/json
- Authorization: Bearer {token}
 	
Body:
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“user”: [

{
 	
}
	{
	“id”: 1,
	“phone”: “89112345678”,
	“email”: “mail@email.ru”,
	“name”: “Иван”,
	“registrationDate”:”01-01-1970”,
	“ordersCount”:4,
	“petsCount”: 2
	}
	]
	}
	}
	---------- Unauthorized ----------
	Status: 401
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 401,
	"message": "Unauthorized"
	}
	}

registrationDate - дата регистрации пользователя
ordersCount - количество объявлений, добавленных пользователем
petsCount - количество животных, у которых нашлись хозяева из числа тех, объявления о которых были добавлены текущим пользователем

- *Изменение номера телефона*

- phone – обязательное

Request	Response

URL: {host}/api/users/~~{id}~~/phone
Method: PATCH
 	
Headers
- Content-Type: application/json
- Authorization: Bearer {token}
 	
Body:
{
“phone”: “+79112345678”,
}
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“status”: “ok”
	}
	}
	---------- Unauthorized ----------
	Status: 401
	Content-Type: application/json
	Body:
	{

	"error": {
	"code": 401,
	"message": "Unauthorized"
	}
	}
	---------- **Validation error** ----------
	(Если phone пустое)
	Status: 422
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 422,
	"message": "Validation error",
	"error": errors: [
	phone: "The phone should not be empty."]
	}
	}

- *Изменение адреса электронной почты*
- email – обязательное, изменение на строку, не являющуюся адресом электронной почты, недопустимо

Request	Response

URL: {host}/api/users/~~{id}~~/email
Method: PATCH
 	
Headers
- Content-Type: application/json
- Authorization: Bearer {token}
 	
Body:
{
“email”: “newmail@user.ru”,
}
	---------- **Successful** ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“status”: “ok”
	}
	}
	---------- **Unauthorized** ----------
	Status: 401
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 401,
	"message": "Unauthorized"
	}
	}

	---------- Validation error ----------
	(Если email пустое)
	Status: 422
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 422,
	"message": "Validation error",
	"error": "errors": [
	"email" "The email should not be empty."
	}
	}

- *Объявления, добавленные пользователем*

Статусы объявлений
active - активное
wasFound - хозяин найден
onModeration - на модерации
archive - в архиве

Request	Response

URL: {host}/api/users/orders/~~{id}~~
Method: GET
 	
Headers
- Content-Type: application/json
- Authorization: Bearer {token}
 	
Body:
{
 	
}
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“orders”: [
	{
	“id”: 1,
	“kind”: “кошка”,
	“photos”: ‘*{url}*/img1.png’,
	“description”: “Найдена кошка, порода Сфинкс, очень грустная”,
	“mark”: “VL-0214”,
	“district”: “Василеостровский”,
	“date”: “01-01-1970”,
	}, "status": "onModeration"
	…
	],
	}
	}

	---------- No results ----------
	Status: 204
	Content-Type: application/json
	Body:
	{
	“data”: {
	“ orders”: []
	}
	}

Удаление пользователем объявления

Удаление возможно только своих объявлений со статусом active и onModeration

Перед удалением необходимо подтверждение пользователем.

После запроса на сервер, объявление на страницах сайта не отображается

Request	Response

URL: {host}/api/users/orders/{id}
Method: DELETE
 	
Headers
- Content-Type: application/json
- Authorization: Bearer {token}
 	
Body:
{
 	
}
-------order not found --------
{
"error": {
"code": 404,
"message": "Order doesn't exist"
}
}
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“status”: “ok”
	}
	}
	---------- Unauthorized ----------
	Status: 401
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 401,
	"message": "Unauthorized"
	}
	}
	---------- Validation error ----------
	(например, для объявлений с другими статусами)
	Status: 403
	Content-Type: application/json
	Body:
	{
	"error": {

	"code": 403, "message": "Access denied", } }

Редактирование пользователем объявления

Редактирование возможно только объявлений со статусом active и onModeration.
Пользователь может редактировать только свои объявления.

Поля, доступные для редактирования:

- photo1, photo2, photo3 - изображения, в формате png, photo1 - обязательное поле
- mark - клеймо, необязательное строковое поле в произвольном формате
- description - текст, описание, дополнительная информация

Request	Response

URL: {host}/api/pets/{id}
Method: ~~PATCH~~ POST
 	
Headers
- Content-Type: multipart/form-data
- Authorization: Bearer {token}
 	
Body:
{
“kind”: “кошка”,
“photo1”: FILE,
“photo2”: FILE,
“photo3”: FILE,
“mark”:”VL-1250”,
“description”:”Веселая собачка породы бультерьер”,
}
 	
-------order not found --------
{
"error": {
"code": 404,
"message": "Order doesn't exist"
}
}
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“status”: “ok”
	}
	}
	---------- Unauthorized ----------
	Status: 401
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 401,
	"message": "Unauthorized"
	}
	}
	---------- Validation error ----------
	(например, для объявлений с другими статусами)
	Status: 403
	Content-Type: application/json
	Body:
	{
	"error": {
	"code": 403,
	"message": "Access denied",
	}
	}

	---------- Validation error ----------
	Status: 422
	Content-Type: application/json
	Body:
	{
	“error”: {
	“code”: 422,
	“message”: “Validation error”,
	“errors”: {
	<key>: <массив ошибок>
	}
	}
	}

Страница с карточкой одного животного

Request	Response

URL: {host}/api/pets/{id}
Method: GET
 	
Headers
- Content-Type: application/json
 	
Body:
{
}
	---------- Successful ----------
	Status: 200
	Content-Type: application/json
	Body:
	{
	“data”: {
	“pet”: [
	{
	“id”: 1,
	“phone”: “89112345678”,
	“email”: “user@user.ru”,
	“name”: “Иван”,
	“kind”: “кошка”,
	“photos”:
	[‘*{url}*/img1.png’,’*{url}*/img2.png’,
	’*{url}*/img3.png’],
	“description”: “Найдена кошка, порода Сфинкс, очень грустная”,
	“mark”: “VL-0214”,
	“district”: “Василеостровский”,
	“date”: “01-01-1970”,
	}
	]
	}
	}
	---------- No results ----------

	----- 	
	Status: ~~204~~ 404
	Content-Type: application/json
	Body:
	{
	“data”: {
	“pets”: []
	}

Страница добавления нового объявления

Добавление нового объявления. При отправке запроса необходимо передать объект со следующими свойствами:

- name – имя пользователя - обязательное поле, строка, допустимые символы - кириллица, пробел, дефис
- phone – обязательное, строка, только цифры, и знак +
- email – обязательное поле, строка, формат - email
- register - Элемент выбора опции автоматической регистрации пользователя при добавлении объявления о найденном животном. Если пользователь выбрал вариант с регистрацией, то появляется блок с полями для ввода пароля и его подтверждения (по умолчанию он скрыт). Если пользователь решил после этого отказаться от регистрации, блок с полями для ввода паролей скрывается (также без использования js).
    - password – обязательное поле, строка, не менее 7 символов, обязательно: 1 цифра, 1 строчная, 1 заглавная буквы
    - password_confirmation – обязательное поле, строка, значение должно совпадать со значением password
- photo1, photo2, photo3 - изображения, photo1 - обязательное поле Формат - только png.
- mark - клеймо, необязательное строковое поле в произвольном формате
- description - текст, описание, дополнительная информация
- confirm – обязательное поле, integer 0 или 1 - для элемента, подтверждающего согласие пользователя на обработку персональных данных. Если нет подтверждения согласия на обработку персональных данных, то необходимо вывести соответствующее сообщение в массиве ошибок
  Если пользователь авторизован, то поля с телефоном, именем, e-mail заполняются автоматически.

Вся валидация проводится средствами JS!

Отправка некорректных данных на сервер невозможна. В случае успешного добавления объявления отображается соответствующее сообщение, в случае ошибки появляется сообщение с текстом ошибки.

Request	Response
URL: {host}/api/pets/~~new~~ **Method:** POST	---------- **Successful** ---------- **Status:** 200 **Content-Type:** application/json

Headers
- Content-Type: multipart/form-data
 	
Body:
{
“name”: “Иван”,
“phone”: “89001234567”,
“email”: “user@user.ru”
“password”: “paSSword1”,
“password_confirmation”: paSSword1”,
“confirm”: 1,
“kind”: “собака”,
“photo1”: FILE,
“photo2”: FILE,
“photo3”: FILE,
“mark”:”VL-1250”,
“description”:”Веселая собачка породы сенбернар”,
“confirm”:1,
} "district": "Адмиралтейский"
	Body:
	{
	“data”: {
	“status”: “ok”,
	“id”: 10,
	}
	}
	
	---------- Validation error ----------
	Status: 422
	Content-Type: application/json
	Body:
	{
	“error”: {
	“code”: 422,
	“message”: “Validation error”,
	“errors”: {
	<key>: <массив ошибок>
	}
	}
	}

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

Разработанное приложение должно быть доступно по адресу http://xxxxxx-m2.wsr.ru/, где xxxxxx - логин участника (указан на индивидуальной карточке).

Проверка работы будет производиться в браузере Google Chrome.

На страницах должны использоваться анимации и микроанимации, реализованные с помощью JS. Созданные анимированные эффекты должны способствовать формированию положительного впечатления о сайте.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

СИСТЕМА ОЦЕНКИ

Секция	Критерий	Сумма
A	Организация работы и управление	1,00
B	Коммуникация и навыки межличностного общения	1,00
C	Графический дизайн	1,00
D	Верстка	0,00
E	Программирование на стороне клиента	18,00

F	Программирование на стороне сервера	0,00
G	CMS	0,00
Всего		21,00