**Задание:**

Создайте web-приложение с применением библиотеки React для получения и отображения данных с тестового сервера, который расположен в корне архива с текстом задания. Также в архиве можно найти инструкцию для запуска тестового сервера.

В ходе выполнения тестового задания требуется разработать две страницы:

**1) Главная**

Представляет из себя веб-страницу, содержащую в себе

1. линейный график, который должен быть построен на основе информации, получаемой 1 раз в секунду с эндпоинта */current\_cpu\_usage*
2. таблицу с последними событиями, состоящую из трех столбцов: дата, время, событие, тип события. Рядом с таблицей должны располагаться фильтры по количеству отображаемых событий, по типу события (выпадающий список, варианты: critiсal, warning, info), а также по времени события (дата с / дата по) Для получения информации для отображения нужно воспользоваться эндпоинтом */recent\_events* с интервалом 3 с

**2) Взаимодействие с объектами**

Представляет из себя веб-страницу, содержащую в себе:

1. Таблицу со списком объектов в локальном хранилище, получаемым с эндпоинта */objects\_list*. Столбцы: id, имя, тип, описание. При нажатии на строку с объектом должно появляться всплывающее окно с возможностью удаления или редактирования объекта (эндпоинт /manage\_object)
2. Форму для добавления объекта (поле для ввода имени, всплывающий список для выбора типа, многострочное текстовое поле для описания (варианты: EMS, Network node, Data Element SNMP), кнопку «принять»). Добавление осуществляется с помощью эндпоинта /manage\_object
3. Окно с выводом JSON-ответа от сервера при добавлении, изменении или удалении объекта из списка

На обеих страницах должны присутствовать шапка с элементами навигации по веб-приложению, а также подвал, прижатый к нижней части страницы.

**Требования:**

* Ключевые моменты в коде должны быть хорошо задокументированы
* Приложение должно быть работоспособным, без ошибок в консоли браузера
* Плюсом будет использование Redux, а также использование CSS-препроцессора
* Классы должны быть названы в соответствии с правилами методологии БЭМ
* Цветовая палитра может быть выбрана самостоятельно

**Срок выполнения задания:** 7 дней с момента получения задания

Задание можно отправить архивом или ссылкой на публичный репозиторий в github на почту opr@micran.ru

**Описание API:**

1. **/current\_cpu\_usage**

Обрабатывает GET запросы, на вход не принимает ничего, возвращает текущую загрузку процессора в процентах

Пример ответа:

|  |
| --- |
| {      "cpu\_usage": 15.2  } |

2. **/recent\_events**

Обрабатывает GET запросы, на вход не принимает ничего, возвращает последние события. Каждые 10 секунд добавляется новое событие.

Пример ответа:

|  |
| --- |
| {      "events": [          {              "event": "Subsystem with id 713 fallen",              "type": "critical",              "date": "15-11-2024",              "time": "13:47"          },          {              "event": "Node #766 fallen with non zero error",              "type": "warning",              "date": "5-11-2024",              "time": "14:47"          },          {              "event": "Subsystem with id 713 fallen",              "type": "critical",              "date": "15-11-2024",              "time": "13:47"          }      ]  } |

3. **/objects\_list**

Обрабатывает GET запросы, на вход не принимает ничего, возвращает список объектов.

Пример ответа:

|  |
| --- |
| {      "objects": [          {              "object\_name": "ems1 starter",              "object\_type": "EMS",              "object\_description": "starter obj 1",              "object\_id": 1          },          {              "object\_name": "NN starter",              "object\_type": "Network node",              "object\_description": "starter obj 2",              "object\_id": 2          },          {              "object\_name": "DE starter",              "object\_type": "Data Element SNMP",              "object\_description": "starter obj 3",              "object\_id": 3          }      ]  } |

4. **/manage\_object**

Обрабатывает POST запросы, имеет 3 режима работы: создание нового объекта, редактирование, удаление

4.1 Создание нового объекта

Для создания нового объекта все поля должны быть заполнены (object\_name, object\_type, object\_description)

Пример запроса:

|  |
| --- |
| {      "operation\_type": "insert",      "data": {          "object\_name": "ems2 example",          "object\_type": "EMS",          "object\_description": "ems2 example description"      }  } |

Пример ответа:

|  |
| --- |
| {      "operation": "insert",      "object\_id": 4,      "object\_instance": {          "object\_name": "ems2 example",          "object\_type": "EMS",          "object\_description": "ems2 example description"      },      "timestamp": "2024-09-16T17:21:18.792903",      "errors": ""  } |

4.2 Изменение существующего объекта

Для изменения существующего объекта обязательно должен быть заполнен object\_id и одно из полей (object\_name, object\_type или object\_description)

Пример запроса:

|  |
| --- |
| {      "operation\_type": "update",      "data": {          "object\_id": 4,          "object\_name": "ems3 udpated example"      }  } |

Пример ответа:

|  |
| --- |
| {      "operation": "update",      "updated\_param": "object\_name",      "errors": ""  } |

4.3 Удаление существующего объекта

Для удаления существующего объекта обязательно должен быть заполнен object\_id

Пример запроса:

|  |
| --- |
| {      "operation\_type": "delete",      "data": {          "object\_id": 4      }  } |

Пример ответа:

|  |
| --- |
| {      "operation": "delete",      "object\_id": 4,      "object\_instance": {          "object\_name": "ems3 udpated example",          "object\_type": "EMS",          "object\_description": "ems2 example description"      },      "timestamp": "2024-09-16T17:26:27.819480",      "errors": ""  } |