# Содержание

[Содержание 1](#_Toc77964674)

[Введение 2](#_Toc77964675)

[Описание проекта и задач 2](#_Toc77964676)

[Аутентификация 3](#_Toc77964677)

[Создание сотрудника 4](#_Toc77964678)

[Получение списка сотрудников 4](#_Toc77964679)

[Создание точки контроля 5](#_Toc77964680)

[Список точек контроля 6](#_Toc77964681)

[Создание группы доступа 6](#_Toc77964682)

[Просмотр всех групп доступа 7](#_Toc77964683)

[Добавление точек контроля к группе доступа 8](#_Toc77964684)

[Добавление сотрудника к группе доступа 8](#_Toc77964685)

[Проверка права доступа сотрудника 8](#_Toc77964686)

[Просмотр всех логов доступа 10](#_Toc77964687)

[Добавление сотруднику права на доступ к точке контроля на N секунд 11](#_Toc77964688)

[База данных 11](#_Toc77964689)

[Инструкция для конкурсанта 12](#_Toc77964690)

[Система оценки 12](#_Toc77964691)

# Введение

**Технологии этого модуля:** REST API

**Время на выполнение**: 3 часа

Вам необходимо использовать все имеющиеся навыки в серверной разработке для создания REST API, которое будет отвечать за контроль доступа к пропускным пунктам на территории заказчика.

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

# Описание проекта и задач

Ваша задача – реализовать REST API, которое будет отвечать требованиям заказчика.

Для вашего удобства, во всех URL будет использоваться псевдоним {host} который обозначает адрес http://xxxxxx-m1.wsr.ru/, где xxxxxx – ваш логин участника.

Заказчик хочет организовать контроль доступа на своей территории с помощью точек контроля. Для этого ему требуется API с соответствующим функционалом.

Точка контроля – это физический контроллер, который считывает какую-то информацию с карточки или QR-кода и проверяет по средствам запроса на сервер есть ли у сотрудника доступ или нет. Точка контроля может иметь родительскую точку, например:

* Полигон 1
  + Кабинет 1001;
  + Кабинет 1002;
* Полигон 2

В системе должен быть предусмотрен только один тип пользователя – это администратор.

Функционал администратора должен быть следующим:

* Вход в систему;
* Точки контроля;
  + Создание точек контроля;
  + Просмотр списка точек контроля;
* Добавление сотрудников в систему;
* Группы доступа;
  + Создание групп доступа;
  + Просмотр списка групп доступа;
  + Добавление точки контроля к группе доступа;
  + Удаление точки контроля из группы доступа;
  + Добавление сотрудника к группе доступа;
* Проверка права сотрудника на доступ к точке контроля;
* Просмотр всех логов доступа;
* Добавление сотруднику права на доступ к точке контроля на N секунд;

### Аутентификация

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

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/login  **Method:** POST  **Headers**  **- Content-Type:** application/json  **Body:**  {  “login”: “admin”,  “password”: “admin”  }  **login** – обязательный параметр  **password** – обязательный параметр | **------------------- Successful --------------------**  **Status:** 200  **Content-Type:** application/json  **Body:** {  “data”: {  “token”: <сгенерированный token>,  “full\_name”: “Alex Adm”  }  }  **---------------- Validation error ----------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <сообщение об ошибке>  }  }  }  **------------------ Unauthorized -----------------**  **Status:** 401  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 401,  “message”: “Unauthorized”,  “errors”: {  “login”: “invalid credentials”  }  }  } |

**Во всех** **последующих запросах необходима аутентификация** с помощью заголовка Authorization**.** Если запрос будет отправлен без данного заголовка или токен будет не действительный, то в ответ сервер должен вернуть следующий ответ:

**Status:** 401

**Content-Type:** application/json

**Body:** {

“error”: {

“code”: 401,

“message”: “Unauthorized”

}

}

### Создание сотрудника

C помощью данного запроса должна быть возможность создавать сотрудника.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/staff  **Method:** POST  **Headers**  **- Content-Type:** multipart/form-data  **- Authorization:** Bearer <token>  **Body:**  **-** **full\_name:** “Ivan Ivanov”  **- photo:** <файл фотографии>  **full\_name** ­– обязательный параметр  **photo** ­– обязательный параметр, jpg | **--------------------- Successful --------------------**  **Status:** 201  **Content-Type:** application/json  **Body:** {  “data”: {  “id”: 1,  “full\_name”: “Ivan Ivanov”,  “code”: <уникальный код сотрудника из 32 символов (генерируется рандомно, закрепляется к сотруднику навсегда)>  }  }  **------------------ Validation error ----------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <error message>  }  }  } |

### Получение списка сотрудников

C помощью данного запроса должна быть возможность получить список сотрудников.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/staff  **Method:** GET  **Headers**  **- Authorization:** Bearer <token> | **-------------------------------- Successful ----------------------------------**  **Status:** 200  **Content-Type:** application/json  **Body:** {  “data”: {  “items”: [  {  “id”: 1,  “full\_name”: “Ivan Ivanov”,  “code”: <уникальный код сотрудника из 32 символов>,  “photo”: <прямая ссылка на фотографию>  }  ]  }  } |

### Создание точки контроля

Под точкой контроля понимается физический контроллер проверяющий права сотрудника на доступ. Данный запрос должен создавать новую точку контроля в системе и возвращать информацию о ней. Точка контроля может иметь родительскую точку контроля.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/points  **Method:** POST  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer <token>  **Body:**  {  “name”: “Кластер A”,  “parent”: 1  }  **name** – обязательный параметр  **parent** ­– не обязательный параметр, идентификатор существующей точки контроля | **-------------------- Successful ---------------------**  **Status:** 201  **Content-Type:** application/json  **Body:** {  “data”: {  “id”: 2,  “name”: “Кластер А”,  “parent”: <id / null>,  }  }  **---------------- Validation error ----------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <error message>  }  }  } |

### Список точек контроля

Данный запрос должен возвращать все точки доступа во вложенном виде.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/points  **Method:** GET  **Headers:**  **- Authorization:** Bearer <token> | **------------------------- Successful --------------------------**  **Status:** 200  **Content-Type:** application/json  **Body:** {  “data”: {  “items”: [  {  “id”: 1,  “name”: “House 1”,  “points”: [  {  “id”: 2,  “name”: “Office 1”,  “points”: []  },  {  “id”: 4,  “name”: “Office 2”,  “points”: [  {  “id”: 5,  “name”: “Chief office”,  “points”: []  }  ]  }  ]  },  {  “id”: 6,  “name”: “House 2”,  “points”: [  {  “id”: 7,  “name”: “Office”,  “points”: []  }  ]  }  ]  }  } |

### Создание группы доступа

Группа доступа – это группа, которая объединяет сотрудников и позволяет предоставить доступ к точкам контроля сразу для всех членов группы.

Данный запрос должен создавать новую группу доступа и возвращать информацию о ней.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/groups  **Method:** POST  **Headers:**  **- Content-Type:** application/json  **- Authorization:** Bearer <token>  **Body:** {  “name”: “Security”  }  **name** – обязательный параметр | **--------------------- Successful ---------------------**  **Status:** 201  **Content-Type:** application/json  **Body:** {  “data”: {  “id”: 1,  “name”: “Security”  }  }  **------------------- Validation error -----------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  “name”: <error message>  }  }  } |

### Просмотр всех групп доступа

Данный запрос должен возвращать список всех групп доступа.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/groups  **Method:** GET  **Headers:**  **- Authorization:** Bearer <token> | **--------------------- Successful ---------------------**  **Status:** 200  **Content-Type:** application/json  **Body:** {  “data”: {  “items”: [  {  “id”: 1,  “name”: “Security”  },  {  “id”: 2,  “name”: “Restaurant”  }  ]  }  } |

### Добавление точек контроля к группе доступа

Данный запрос должен добавлять переданные в запросе точки контроля к группе доступа. В теле запроса необходимо передать свойство points с идентификаторами точек контроля. Идентификаторы точек контроля должны проверяться валидацией на существование.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/groups/<id>/points  **Method:** POST  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer <token>  **Body:**  {  “points”: [1,2,3]  }  **points** – обязательный параметр, массив существующих идентификаторов, которые должны быть добавлены в группу | **---------------------- Successful ------------------------**  **Status:** 201  **-------------------- Validation error --------------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  “points”: <error messages separated by commas>  }  }  } |

### Добавление сотрудника к группе доступа

Данный запрос должен добавлять переданных в запросе сотрудников к группе доступа. В теле запроса необходимо передать свойство staff с идентификаторами сотрудников.

Идентификаторы сотрудников должны проверяться валидацией на существование.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/groups/<id>/staff  **Method:** POST  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer <token>  **Body:**  {  “staff”: [1,2,3]  }  **staff** – обязательный параметр, массив существующих идентификаторов сотрудников, которые должны быть добавлены в группу доступа | **----------------------- Successful -----------------------**  **Status:** 201  **-------------------- Validation error --------------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  “staff”: <error messages separated by commas>  }  }  } |

### Проверка права доступа сотрудника

Данный запрос должен проверять право доступа сотрудника к точке контроля. В запросе передается уникальный код сотрудника (не ID) и идентификатор точки контроля, а в ответе возвращается фото сотрудника и true/false в зависимости от наличия права на доступ.

Право на доступ определяется по следующим условиям:

* Если сотрудник принадлежит группе доступа, которая имеет доступ к точке контроля, то сотрудник должен имеет доступ.
* Если вышеуказанное условие не выполняется, но администратор предоставил сотруднику право на доступ в течении X секунд, то сотрудник должен иметь доступ к указанной контрольной точке X секунд с момента предоставления права на доступ.

Так как точки контроля могут иметь родительские точки, то это означает, что, если сотруднику предоставляется право на доступ к данной точке, то сотрудник автоматически должен иметь доступ и ко всем родительским точкам.

Например, представим следующую структуру точек:

* Вход
  + Полигон 1
    - Кабинет 1001;
    - Кабинет 1002;
  + Полигон 2
    - Кабинет 2001;

Если сотрудник будет иметь доступ к точке контроля «Кабинет 1002», то он также должен иметь доступ к «Полигон 1» и «Вход» соответственно.

Любая попытка доступа должна фиксироваться в системе для дальнейшего просмотра.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/access  **Method:** POST  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer <token>  **Body:**  {  “staff”: <уникальный код сотрудника из 32 символов, не ID>,  “point”: <id точки контроля>  }  **staff** – обязательное поле, код сотрудника  **point** – обязательное поле, существующий идентификатор точки контроля | **----------------------- Successful -----------------------**  **Status:** 200  **Content-Type:** application/json  **Body**: {  “data”: {  “photo”: <ссылка на фото сотрудника>,  “access”: <true/false>  }  }  **-------------------- Validation error --------------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <error message>  }  }  } |

Также во время проверки доступа вы должны сохранить ссылку на фотографию того сотрудника, кто инициировал проверку доступа. Для этого вам необходимо обратиться к внешнему API и передать туда ID точки доступа. В ответ вы получите ссылку на фотографию с камеры возле данной точки контроля.

**Внешнее API**

|  |  |
| --- | --- |
| Request | Response |
| **URL:** http://xkesryp-m1.wsr.ru/vcs/points/<ID>  **Method:** POST  **Query params:**  **- ID**: идентификатор точки контроля | **Status:** 200  **Content-Type:** application/json  **Body**: {  “data”: {  “url”: <ссылка на фото с камеры>,  }  } |

### Просмотр всех логов доступа

Данный запрос должен возвращать данные о всех попытках доступа к контрольным точкам.

Логи должны быть отсортированы по новизне (сначала новые).

Должна быть реализована возможность фильтровать логи используя GET-параметр **type.** Параметр должен принимать следующие значения:

* Параметр не указан – должны вернуться все логи.
* staff – должны вернуться логи сотрудника (id сотрудника передается GET-параметром id).
* point – должны вернуться логи точки контроля (id точки передается GET-параметром id).

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/logs  **Method:** GET  **Headers**  **- Authorization:** Bearer <token>  **Query params:**  **- type:** staff | point  **- id:** существующий идентификатор сотрудника или точки контроля (если значение параметра **type** равно **staff** или **point**) | **Status:** 200  **Content-Type:** application/json  **Body**: {  “data”: {  “items”: [  {  “access”: <true/false>,  “staff”: {  “id”: 1,  “full\_name”: “Alex”,  “photo”: <ссылка на фото профиля>,  “camera”: <ссылка на фото с камеры>  },  “point”: {  “id”: 1,  “name”: “Security”  },  “timestamp”: 4239213213  }  ]  }  } |

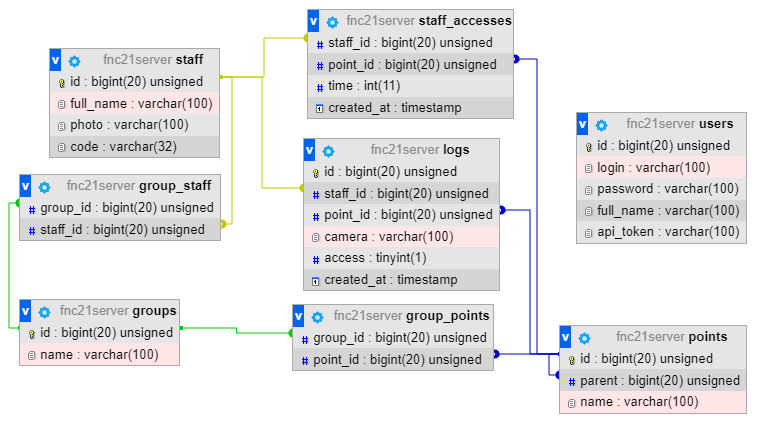
### Добавление сотруднику права на доступ к точке контроля на N секунд

С помощью данного запроса администратор должен иметь возможность предоставить сотруднику право на доступ к точке контроля на указанный период времени.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/staff/<id>/access  **Method:** POST  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer <token>  **Body:**  {  “point\_id”: 7, // обязательное поле, существующий ID точки контроля  “time”: 3600 // обязательное поле, больше 0, кол-во секунд, на которое предоставляется доступ  } | **-------------------- Successful -------------------**  **Status:** 201  **---------------- Validation error ----------------**  **Status:** 422  **Content-Type:** application/json  **Body:** {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <error message>  }  }  } |

### База данных

Вам предоставляется готовая база данных. Вы можете добавлять и удалять данные по своему усмотрению, но вы не можете менять структуру базы данных. При проверке все ваши данные, **кроме таблицы users** будут заменены на другие тестовые данные. Поэтому обязательно заведите пользователя с соответствующим логином и паролем как сказано далее.



# Инструкция для конкурсанта

Разработанное API должно быть доступно по адресу http://xxxxxx-m1.wsr.ru/, где xxxxxx - логин участника.

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

Вы должны **обязательно** **завести учетную запись** **пользователя** в своей системе со следующими данными для входа:

* Логин: admin
* Пароль: 1234

**Проверяться будут только работы, загруженные на сервер!**

**Работы будут проверяться автоматически!**

# Система оценки

|  |  |  |
| --- | --- | --- |
| **Секция** | **Критерий** | **Сумма** |
| A | Организация работы и управление | 4,00 |
| B | Коммуникация и навыки межличностного общения | 4,00 |
| C | Графический дизайн | 0,00 |
| D | Верстка | 0,00 |
| E | Программирование на стороне клиента | 0,00 |
| F | Программирование на стороне сервера | 14,00 |
| G | CMS | 0,00 |
| **Всего** |  | 22,00 |