**Backend**

Необходимо реализовать **REST API** интерфейс для работы с сущностью «оборудование».

Обязательные методы:

| **#** | **Роут** | **Действие** |
| --- | --- | --- |
| 1 | GET:/api/equipment | Вывод пагинированного списка оборудования с возможностью поиска путем указания query параметров советующим ключам ответа |
| 2 | GET:/api/equipment/{id} | Запрос данных по id |
| 3 | POST:/api/equipment | Создание новой(ых) записи(ей) |
| 4 | PUT:/api/equipment/{id} | Редактирование записи |
| 5 | DELETE:/api/equipment/{id} | Удаление записи |
| 6 | GET:/api/equipment-type | Вывод пагинированного списка типов оборудования с возможностью поиска путем указания query параметров советующим ключам ответа |

Для хранения информации об оборудовании используется 2 таблицы в базе данных (использовать MySQL):

1. «Тип оборудования» поля: код, наименование типа, маска серийного номера.
2. «Оборудование» поля: код, код типа оборудования, серийный номер (уникальное поле), примечание.

При создании новых записей в поле «серийный номер» интерфейс принимает массив номеров каждый из которых валидируется на соответствие маске и отсутствие в таблице «оборудование» в качестве уникального ключа. Создание может быть как по одному объекту, так и коллекция (на входе json-массив).

Если «серийный номер» не проходит валидацию, по нему формируется соответствующие сообщение, которое возвращает интерфейс, после обработки всего массива номеров.

| **id** | **Тип оборудования** | **Маска SN** |
| --- | --- | --- |
| 1 | TP-Link TL-WR74 | XXAAAAAXAA |
| 2 | D-Link DIR-300 | NXXAAXZXaa |
| 3 | D-Link DIR-300 S | NXXAAXZXXX |

**Где:**

* **N** – цифра от 0 до 9;
* **A** – прописная буква латинского алфавита;
* **a** – строчная буква латинского алфавита;
* **X** – прописная буква латинского алфавита либо цифра от 0 до 9;
* **Z** –символ из списка: “-“, “\_”, “@”.

**Обязательные требования:**

1. использование фреймворка Laravel v.8+.
2. наличие PHPDoc для всех методов
3. использование Form Request Validation
4. все запросы API должны быть авторизованы (проверка Bearer Authentication)
5. все ответы API должны использовать API Resources & Resource Collections
6. вся бизнес-логика манипуляции с данными должна быть инкапсулирована в моделях и сервисном слое приложения (контроллеры должны быть плоскими)

**Материалы для решения задачи:**

* PHP: <https://www.php.net/>
* Laravel <https://laravel.com/>
* MySQL Server (<http://www.mysql.com/downloads/mysql/>)
* regex101 (<https://regex101.com/>)

**Frontend**

Для отображения сделать минимальное **SPA-приложение** с использованием **Vue.js**, работающее с разработанным API.

**Форма добавления записей, содержащая:**

1. Список «Тип оборудования» (id типа) select.
2. поле «Серийные номера» input-text/textarea.
3. Поле «Примечание» textarea
4. Кнопка «Добавить».

**Форма поиска, редактирования и удаления записей:**

1. Поисковая строка input-text, для поиск по серийному номеру/примечанию.
2. Поле «Код оборудования» (id) — input-text.
3. Поле «Тип оборудования» (наименование) — input-text/select.
4. Поле «Серийный номер» — input-text.
5. Поле «Примечание» — input-text/textarea.
6. Кнопки редактировать/сохранить/удалить в зависимости от выдранного режима (просмотр/редактирование).

В качестве **ui kit** можно использовать любой готовый фреймворк/библиотеку.

Редактирование может быть реализовано как в целом всех полей, так и отдельными полями.

Для всех форм обязательно наличие минимальной валидации данных на стороне клиента.