Разработка системы управления установками вентиляции Comeforta

Ноябрь 18 , 2020

Сокращения

СУУ - система управления установками вентиляции
API - сервис для управления данными БС.

Общие требования

Необходимо реализовать сервис API для работы с установками очистки воздуха с рекуперацией.

API должен служить единственной точкой(endpoint) для управления с данными СУУ. То есть веб сайт (приложение) также будет работать с данными СУУ как клиент.

API состоит из следующих модулей:

- Модуль работы с сущностями СУУ
- Модуль работы с сущностями пользователей
- Модуль работы с сущностями логов

Основным типом обмена информации должен служить JSON, сам сервис API должен быть реализован учитывая подходы REST.

Описание СУУ

СУУ - система для обслуживания установок очистки воздуха. Данная установка обеспечивает приток свежего воздуха с улицы и вытяжку отработанного воздуха из помещения. СУУ позволяет добавлять новые установки в систему, управлять ими (удалять, редактировать), а также назначать пользователя на установку.

Пользователь скачивает приложение с маркетов Google Play и Apple Store и с его помощью регистрирует установку в СУУ и далее использует ее по назначению.

Различают следующие типы пользователей СУУ:

- Пользователь - человек, который приобрел установку (сам пользователь не имеет доступа к сайту и взаимодействует только через API).
- Администратор - человек, который имеет право на добавление (удаление, редактирование) установок в СУУ.

Ключевой сущностью СУУ является установка. Установка содержит:

- UIN (обязательное поле) - данное поле необходимо сгенерировать при создании сущности. Поле должно иметь вид ‘zCO2-[0-1A-Z]{12}’ (случайная комбинация из 12 букв либо чисел и префиксом ‘zCO2-’)
- Name (не обязательное поле)
- Description (не обязательное поле)
- Serial number (обязательное поле)
- Firmware version (обязательное поле)

- Firmware last update date (не обязательное поле)
- Device enabled (обязательное поле, по умолчанию false)
- Device mode (обязательное поле) - поле может принимать следующие значения: Off, One, Two, Three, Four
- Network mode (обязательное поле) - поле может принимать следующие значения: Home, OneDevice (в будущем возможно дополнение новыми значениями)
- Last CO2 value (не обязательное поле)
- In use (обязательное поле, по умолчанию false) - если установка не назначены ни какому пользователю, то поле принимает значение false.
- Night mode enabled (обязательное поле, по умолчанию false)
- Night mode auto (обязательное поле, по умолчанию false)
- Night mode from (не обязательное поле)
- Night mode to (не обязательное поле)
- Date created (обязательное поле)
- Date updated (обязательное поле)

Сущность Пользователя содержит следующие поля:

- Email (обязательное поле)
- Password (обязательное поле)
- Is Confirmed (обязательное поле)
- Role (обязательное поле) - Admin, User
- Date created (обязательное поле)
- Date updated (обязательное поле)

Так же есть сущность Логов:

- Log Content (обязательное поле)
- Date created (обязательное поле)

Каждому пользователю в один и тот же момент может принадлежать только одна СУУ.

Каждой СУУ может соответствовать неограниченное количество логов.

Описание модулей API

1. Модуль работы с сущностями СУУ
2. Модуль работы с сущностями пользователей
3. Модуль работы с сущностями логов

Модуль работы с сущностями СУУ

Данный модуль должен предоставлять возможности управления данными сущностей в виде следующих методов:

1.1 Метод создания установки

По данном методу клиенты API могут добавить новую установку в систему. При этом данный метод могут вызвать только пользователи, имеющие роль **Admin**. Обычный пользователь не может его использовать.
На вход метод будет принимать следующие данные:

- Name - текст
- Description - текст
- SerialNumber - текст
- Firmware version - число

1.2 Метод удаления установки

По данном методу клиенты API могут удалить установку из систему. При этом данный метод могут вызвать только пользователи, имеющие роль **Admin**. Обычный пользователь не может его использовать.

На вход метод будет принимать следующие данные:

- ID - идентификатор установки

1.3 Метод изменения установки

По данном методу клиенты API могут изменить данные установки из систему. При этом данный метод могут вызвать только пользователи, имеющие роль **Admin**. Обычный пользователь не может его использовать.

На вход метод будет принимать все возможные из схемы сущности данные. Так же, метод должен позволять изменять связь между пользователями и установками.

1.4 Метод получения списка установок (общий список)

По данном методу клиенты API могут получить список всех установоки в системе. При этом данный метод могут вызвать только пользователи, имеющие роль **Admin**. Обычный пользователь не может его использовать. Список должен отображать связь пользователя и установки. Так же, метод должен поддерживать пагинацию.

1.5 Метод назначения СУУ определенному пользователю

Пользователь, который смог авторизоваться в системе использую API (обычный пользователь, не Admin), использую этот метод может добавить к своей учетной записи купленную установку. При этом, как было написано выше, он может управлять только одной установкой, поэтому данный метод должен не давать возможности добавления установки, если пользователь уже имеет в учетной записи назначенную на себя установку. Метод также должен уведомлять, если установка занята другим пользователем.

На вход метод будет принимать следующие данные:

- UIN - идентификатор установки

1.6 Метод получения данных установки (текущей для пользователя)

По данном методу обычный пользователь может получить данные установки, назначенной для него в текущий момент.

Метод должен возвращать следующие данные:

- ID
- UIN
- Name
- Description
- Firmware version
- Device mode
- Network mode

- Last CO2 value
- Night mode enabled (обязательное поле, по умолчанию false)
- Night mode auto (обязательное поле, по умолчанию false)
- Night mode from (не обязательное поле)
- Night mode to (не обязательное поле)

1.7 Метод изменения данных установки (текущей для пользователя)

По данном методу обычный пользователь может изменить данные установки, назначенной для него в текущий момент.

На вход метод будет принимать следующие данные:

- ID
- Name
- Description
- Device mode
- Network mode
- Last CO2 value
- Night mode enabled (обязательное поле, по умолчанию false)
- Night mode auto (обязательное поле, по умолчанию false)
- Night mode from (не обязательное поле)
- Night mode to (не обязательное поле)

Модуль работы с сущностями пользователей

Данный модуль должен предоставлять возможности управления данными сущностей в виде следующих методов:

1.1 Метод регистрации пользователя

По данном методу обычные пользователи могут создать себе учетную запись в системе.

На вход метод будет принимать следующие данные:

- Email
- Password (обязательное поле)

Метод должен проводить валидацию введенных пользователем данных. Не пропускать слишком короткие и простые пароли, проверять на валидность email. Так же, метод должен возвращать корректные описания возникших проблем (такие как пользователь уже существует в системе)

1.2 Метод удаления пользователя администратором

По данном методу администраторы могут удалить обычного пользователя из системы. Метод доступен только для роли Admin.

1.3 Метод получения списка пользователей (общий список)

По данном методу клиенты API могут получить список всех пользователей в системе. При этом данный метод могут вызвать только пользователи, имеющие роль **Admin**. Обычный пользователь не может его использовать. Так же, метод должен поддерживать пагинацию.

1.4 Метод аутентификации пользователя

По данном методу клиенты API могут зайти в систему. Для текущей версии API используем **Basic Authentication**. Метод не должен возвращать роль пользователю. Роль должна учитываться только на сервере.

Модуль работы с сущностями логов

Данный модуль должен предоставлять возможности управления данными сущностей в виде следующих методов:

1.1 Метод создания лога

По данном методу пользователи могут добавлять логи к СУУ, которая назначена на его учетную запись. При этом, метод должен не давать добавлять логи к другим установкам пользователей (то есть можно добавлять логи только для текущей установки).

На вход метод будет принимать следующие данные:

- ID установки
- Log Content

1.2 Метод получения списка логов для установки (список)

По данном методу клиенты API могут получить список всех логов для определенной СУУ в системе. Если метод вызывает пользователь с ролью **Admin**, то в любом случае возвращаем логи для установки. Если метод вызывает пользователь с ролью **User**, то возвращаем логи только в том случае, если установка является текущей для данного пользователя. Метод должен поддерживать пагинацию.

На вход метод будет принимать следующие данные:

- ID установки

1.3 Метод удаления списка логов для установки

Используя этот метод, пользователь с ролью **Admin**, может почистить логи для установки.

На вход метод будет принимать следующие данные:

- ID установки

Технические требования

Все методы API должны следовать следующим набором правил

- Стараться использовать единый адрес (endpoint) для работы с сущностями (использовать соответствующие HTTP headers - GET, PUT, POST DELETE)
- Методы не должны (по возможности) ‘падать’. Все ошибки сервера должны быть отработаны и клиент должен видеть стандартный HTTP ответ - а не страницу ошибки сервера.
- Методы должны максимально использовать стандартные HTTP коды при формировании ответов API. Эти коды должны возвращаться наряду с ‘человеческим’ описанием ошибки. Например вместе с сообщением о неправильном входе пользователя, ответ должен идти с кодом 401.
- Методы должны быть задокументированы.

Разарботку вести на github. Проект должен иметь грамотно составленный readme, с инструкцией по развертыванию приложения. Ну и понятно, обязательным является использование виртуальных окружений (virtual env и т.д.)