СОДЕРЖАНИЕ

ВВЕДЕНИЕ *2*

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ *3*

Интерфейсная часть **4**

Страница регистрации 4

Страница входа 4

Страница с проектами 4

Страница модулей 4

Страница добавления модулей 4

Страница создания проекта 5

Внешнее API для информирования об успешной сборке 7

Ключи доступа 7

API **8**

Запрос без ключа доступа 8

Получение списка проектов пользователя 8

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА *8*

СХЕМА ОЦЕНКИ *9*

ВВЕДЕНИЕ

Вам предстоит разработать веб-сервис для управления модульной системой автоматизации “IoT Manager” на базе микроконтроллеров ESP32/ESP8266.

Система автоматизации представляет собой некое приложение на Python и C++, исходный код которого вам будет предоставлен в медиафайлах. Вам НЕ нужно взаимодействовать с этим приложением, компилировать, запускать или работать с Python и C++. Вам необходимо работать со структурой проекта. Данный проект имеет следующую структуру:

```
data_svelte
include
lib
src/
  classes/
  modules/
    exec/
      moduleA
      moduleB
    sensors/
      moduleC
      moduleD
    virtual/
      moduleE
      moduleF
  utils/
tools
training
.gitignore
LICENSE
PrepareProject.py
README.md
myProfile.json
platformio.ini
```

Данная система содержит ряд модулей, которые располагаются в папке src/modules. Модули группируются по своему типу, например выполняемые (exev), сенсоры (sensors), виртуальные (virtual). Внутри группы располагаются папки с модулями – у папки, может быть, любое имя и она может содержать набор файлов и папок.

В корне приложения находится файл myProfile.json, который содержит настройки приложения, а также список модулей.

Заказчик хочет, чтобы будущая поддержка разрабатываемого сервиса была простой, поэтому он настаивает на использовании одного из серверных фреймворков, таких как Laravel или Yii.

Пример файла myProfile.json

```
{
    "iotmSettings": {
       "settings": "",
       "name": "IoTmanagerVer4",
       "apssid": "IoTmanager",
       "appass": "",
       "routerssid": "rise",
       "routerpass": "hostel3333",
       "timezone": 1,
       "ntp": "pool.ntp.org",
       "weblogin": "admin",
       "webpass": "admin",
       "mqttServer": "m2.wqtt.ru",
       "mqttPort": 8021,
       "mqttPrefix": "/risenew",
       "mqttUser": "rise",
       "mqttPass": "3hostel3",
       "serverip": "http://iotmanager.org",
```

```
        "log": 0,
        "mqttin": 0
    },
    "projectProp": {
        "platformio": {
            "default_envs": "esp8266_4mb",
            "data_dir": "data_svelte"
        }
    },
    "modules": {
        "virtual": [
            { "path": "src\\modules\\virtual\\Logging", "active": true },
            { "path": "src\\modules\\virtual\\Timer", "active": true }
        ],
        "sensors": [
            { "path": "src\\modules\\sensors\\Ads1115", "active": false },
            { "path": "src\\modules\\sensors\\Aht20", "active": true }
        ],
        "exec": [
            { "path": "src\\modules\\exec\\ButtonIn", "active": true },
            { "path": "src\\modules\\exec\\ButtonOut", "active": true }
        ],
        "display": [
            { "path": "src\\modules\\display\\Lcd2004", "active": true }
        ]
    }
}
```

Данный файл имеет ряд настроек, свойств и перечень модулей, которые имеются в системе. Каждый модуль в этом файле имеет путь к папке с модулем и свойство active, отвечающее за то попадает ли данный модуль в сборку или нет.

Разрабатываемый вами сервис также должен иметь свое собственное API, позволяющее внешним приложениям получать список ваших проектов и просматривать информацию о проекте.

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

Ваша задача – разработать веб-сервис, позволяющий пользователям создавать необходимые им сборки данной системы, но с разным набором модулей. Сборки не нужно компилировать, запускать или взаимодействовать как-либо с C++ и Python. Все что вам нужно – создавать zip архив для проекта, содержащий разные набор модулей и корректный myProfile.json файл.

Алгоритм работы:

1. Исходные файлы «IoT Manager» размещаются на сервере, в директории iot, в корне модуля.
2. Пользователь сервиса заходит в ваш сервис
3. Пользователь переходит на страницу загрузки модулей
4. Пользователь загружает zip-архив, содержащий минимум 1 папку с модулем
5. Пользователь переходит на страницу создания проекта
6. Пользователь указывает требуемые настройки, выбирает какие модули исходной системы (лежащей в директории iot) нужно включить в проект, выбирает какие из собственных модулей (загруженных ранее) необходимы и создает проект.
7. Информация о проекте сохраняется в БД, пользователь получает возможность скачать zip-архив с системой «IoT Manager», содержащей только те модули, которые были указаны. Файл myProfile.json содержит верную информацию о модулях, включенных в сборку и настроечные параметры системы.

Вам необходимо разработать веб-сервис со следующей структурой:

- Веб-интерфейс:
  - Регистрация
  - Вход
  - Страница модулей
  - Страница добавления модуля
  - Страница проектов
  - Страница создания проекта
  - Страница ключей доступа
- API:
  - Получение списка проектов

Интерфейсная часть

Страница регистрации

Любой желающий должен иметь возможность зарегистрироваться в системе используя адрес электронной почты и пароль.

Страница входа

Зарегистрированные пользователи должны иметь возможность войти в систему используя комбинацию из электронной почты и пароля, указанных при регистрации.

После успешного входа в систему пользователь должен увидеть страницу с проектами.

Страница с проектами

На данной странице пользователи должны видеть свои ранее созданные проекты.

Каждый проект должен содержать:

- Наименование проекта
- Кнопка удаления
- Кнопка скачивания архива с проектом

Страница модулей

На данной странице должны отображаться модули, которые загрузил пользователь.

Каждый модуль должен отображать следующую информацию:

- Наименование модуля – берется из файла modinfo.json расположенного в папке с модулем
- Кнопка скачивания – при нажатии должен быть скачан zip-архив с данным модулем
- Кнопка удаления – при нажатии данный модуль должен быть удален из системы

На данной странице также должна находиться кнопка для перехода на страницу добавления нового модуля.

Страница добавления модулей

Данная страница должна содержать форму для загрузки модулей. Форма должна состоять из 1 поля для выбора zip архива и кнопки для отправки формы.

Данные формы должны валидироваться на стороне сервера: файл с модулями должен быть обязательным zip-архивом.

В результате отправки формы, модули содержащиеся в загруженном архиве должны быть добавлены в систему и отображаться на странице модулей у пользователя, который их загрузил.

Подразумевается, что zip-архив всегда содержит минимум 1 папку с модулем. Если в zip-архиве находится 3 папки, то в систему должны быть добавлены 3 модуля.

Внутри каждой папки с модулем обязан находиться файл modinfo.json. Из него система берет название модуля.

Страница создания проекта

На данной странице необходимо отобразить форму, содержащую следующие поля:

- Название создаваемого проекта – обязательно для заполнения
- Логин от Wi-Fi – обязательно для заполнения, соответствует полю routerssid в файле myProfile.json
- Пароль от Wi-Fi – обязательно для заполнения, соответствует полю routerpass в файле myProfile.json
- Перечень модулей из базового шаблона – 0 и более модулей
- Перечень пользовательских модулей (загруженных в свой аккаунт) - 0 и более модулей
- Кнопка отправки

По умолчанию, все модули базового проекта должны быть отмечены.

В процессе заполнения формы пользователь может выбрать только те модули базового шаблона, которые ему нужны, а также свои собственные модули, которые также должны попасть в сборку.

Исходные модули базового шаблона приложения должны браться из папки базового шаблона, расположенного в директории «iot», в корне модуля. Если заменить файлы исходного шаблона на другие вручную, то все должно работать и должны отображаться те модули, которые находятся в данный момент в папке шаблона.

Пользователь может отметить все модули базового шаблона и не добавлять свои, тогда создаваемый проект будет точно таким же, как и базовый шаблон проекта (Рисунок 1).

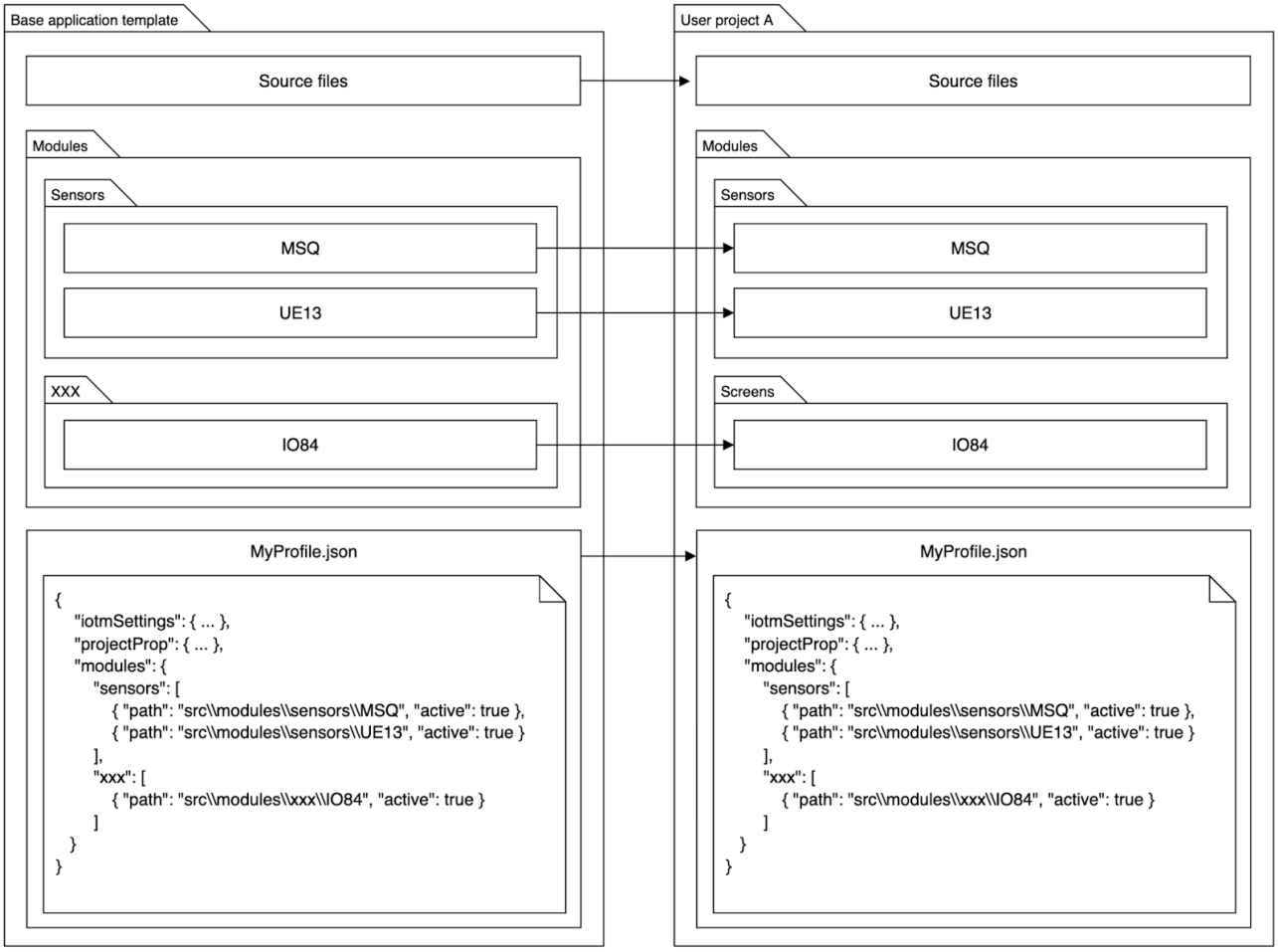

Рисунок 1 – Сборка проекта на основе полного базового шаблона

Если пользователь отметит только часть модулей базового проекта, то тогда не отмеченные модули не должны попадать в сборку (Рисунок 2).

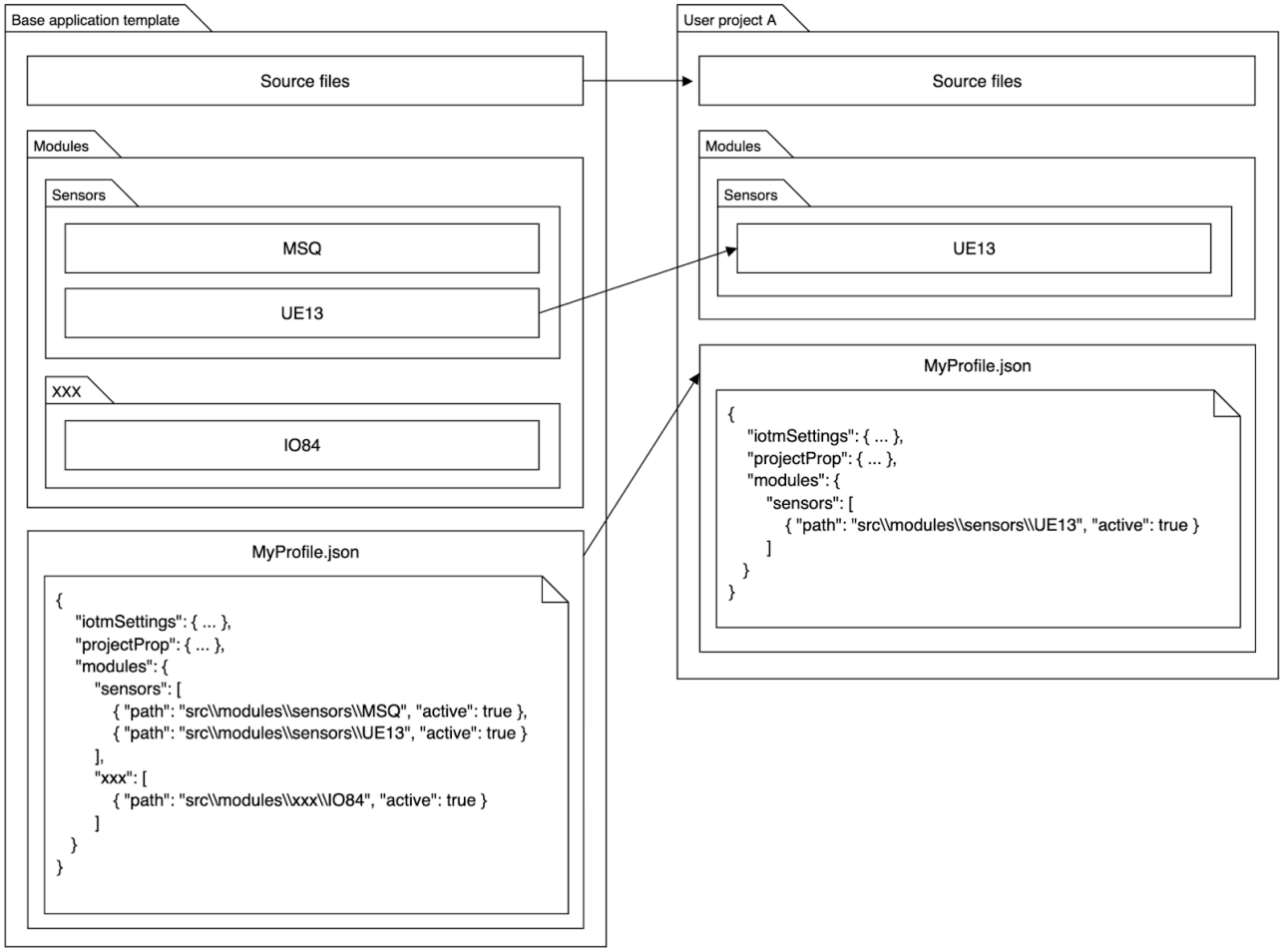

Рисунок 2 – Сборка проекта на основе части базового шаблона

Также пользователь может выбрать и свои модули тоже, тогда они также должны быть включены в сборку. Пользовательские модули должны попадать в папку src/modules/custom (Рисунок 3).

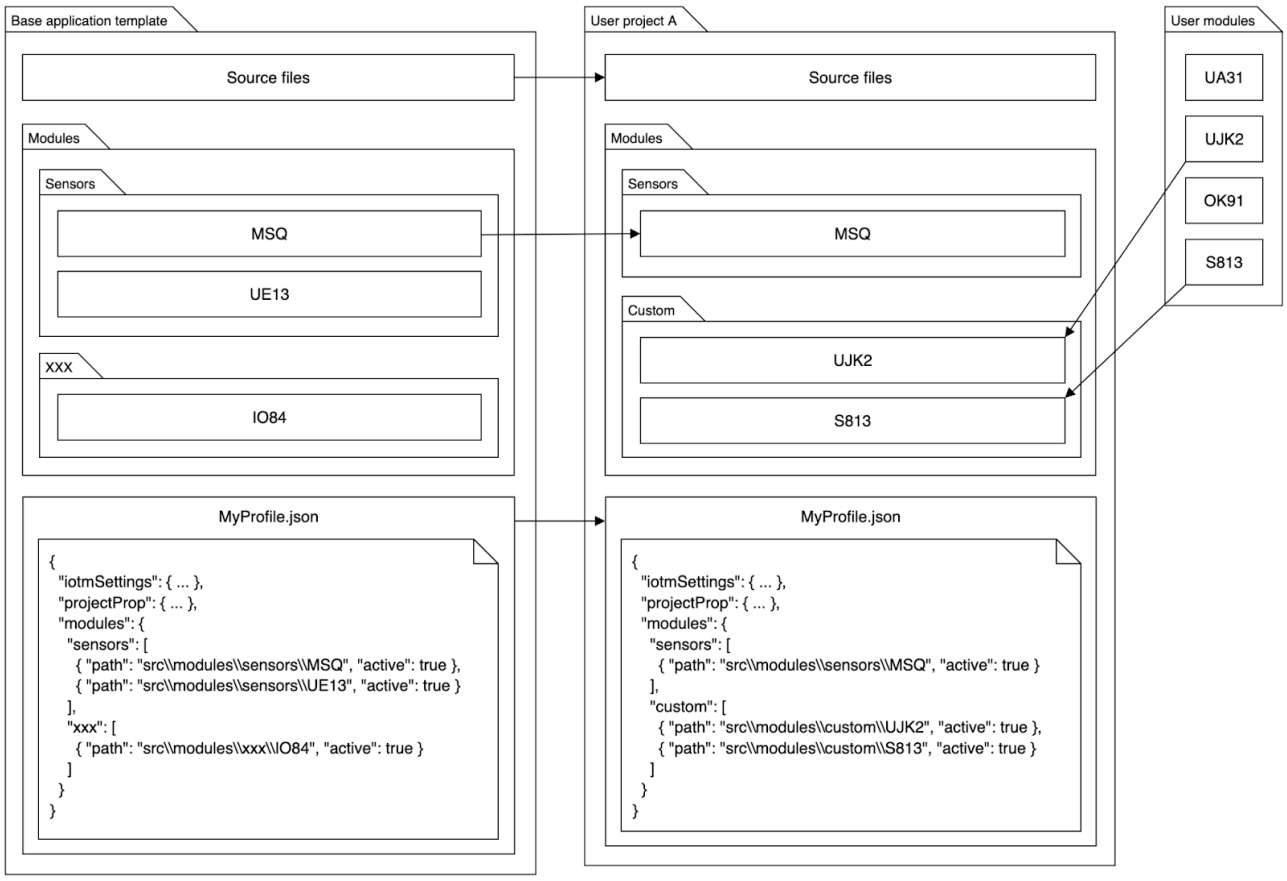

Рисунок 3 – Сборка проекта на основе части базового шаблона и пользовательских модулей

После отправки формы должен быть создан zip-архив содержащий все выбранные модули, а также корректный myProject.json файл. Пользователь должен быть перенаправлен на страницу со своими проектами.

Файл myProject.json формируется на базе уже готового файла из базового шаблона myProfile.json с заменой указанных данных на странице создания проекта.

Также, после создания проекта вы должны отправить запрос со своего сервера к предоставленному API для информирования об успешном создании новой сборки.

Внешнее API для информирования об успешной сборке

URL: `http://nvafmzc-m2.wsr.ru/api/builds`
Method: `POST`
Request Headers:

- `Accept: application/json`
- `Content-Type: application/json`

Request body:

```
{
    “competitor”: <your static name>,
    “project”: “Project name”,
    “url”: “http://<competitor_login>-m2.wsr.ru/files/<unique_filename>.zip”
}
```

Response status: `204`

Ключи доступа

На данной странице необходимо отобразить информацию (название и токен) о пользовательских ключах доступа, которые позволят взаимодействовать с вашим API. Ключ-доступа – это просто именованный API-токен, с помощью которого приложение будет получать доступ к системе.

У каждого ключа на странице должна быть кнопка для удаления.

На странице должна быть форма добавления нового ключа. Для добавления достаточно ввести только название ключа и отправить форму.

API

Запрос без ключа доступа

При попытке обратиться к любому запросу API без корректного токена клиенту должен быть возвращен следующий ответ:

Response Headers:
- Content-Type: application/json

Response status: 401
Response body:

```
{
   code: 401,
   message: “Authorization error”,
   error: “AUTH_TOKEN_INCORRECT”,
   errors: []
}
```

Получение списка проектов пользователя

URL: /api/projects
Method: GET
Request Headers:
- Accept: application/json
- Authorization: Bearer <token>

Response status: 200
Response body:

```
{
   data: [
       {
           “id”: 1,
           “name”: “Project A”,
           “file”: “http://<competitor_login>-m2.wsr.ru/files/<unique_filename>.zip”
       }
   ]
}
```

В поле “file” должна быть ссылка на получение zip-файла с архивом проекта.

Необходимо защитить файл от просмотра и скачивания путем использования API токена приложения.

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

Сохраните работу на сервере в папке второго модуля.

Разработанный вами сервис должен открываться по адресу http://xxx-m2.wsr.ru, где xxx – ваш логин.

Разработанное вами API должно отвечать по адресу http://xxx-m2.wsr.ru/api, где xxx – ваш логин.

Работы, не сохраненные на сервере, а также работы, которые были сохранены с ошибкой или были неправильно развернуты не будут проверяться. Поэтому убедитесь в работоспособности вашего сервиса.

Оценка веб-интерфейса будет проводиться в браузере Google Chrome, а для оценки API будет использоваться Postman.

СХЕМА ОЦЕНКИ

КРИТЕРИЙ	ИЗМЕРИМЫЕ	СУДЕЙСКИЕ	ВСЕГО
Верстка	6,00	0,00	6,00
Функционал	8,00	0,00	8,00
Создание проекта	5,30	0,00	5,30
Ключи доступа	1,00	0,00	1,00
API	2,70	0,00	2,70
Общее	2,00	1,00	3,00
Качество кода	0,00	3,00	3,00
Дизайн	0,00	6,00	6,00
Всего	**25,00**	**10,00**	**35,00**