**Министерство образования Республики Беларусь**

**Учреждение образования**

**«Национальный детский технопарк»**

**Проектная работа**

**Программно-аппаратный комплекс «HomeBrain» для управления и мониторинга IoT-системы**

**Работу выполнил:** Хренков Даниил Денисович

**Руководители:** Листопадов Сергей Александрович, Карпович Дарья Валериевна

Минск 2023

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc151385924)

[1. Постановка задач 4](#_Toc151385925)

[1.1 Обзор существующих аналогов 4](#_Toc151385926)

[1.2 Функциональное назначение 5](#_Toc151385927)

[2. Проектирование веб-ресурса 6](#_Toc151385928)

[2.1 Выбор и обоснование инструментов разработки 6](#_Toc151385929)

[2.2 Структура базы данных 6](#_Toc151385930)

[2.3 Структура веб-ресурса 7](#_Toc151385931)

[3. Тестирование 16](#_Toc151385932)

[4. Инструкция по установке 16](#_Toc151385933)

[5. Руководство 17](#_Toc151385934)

[ЗАКЛЮЧЕНИЕ 25](#_Toc151385935)

[Список использованных источников 26](#_Toc151385936)

# ВВЕДЕНИЕ

В настоящее время IoT-системы являются одним из наиболее актуальных и востребованных направлений в области информационных технологий. Данные системы позволяют управлять различными устройствами, такими как освещение, отопление, кондиционирование, безопасность, и др.

IoT-системы становятся все более распространенными в современном мире. Они доступны для большинства пользователей, и технологии, используемые в этих системах, становятся все более доступными и дешевыми. Таким образом, пользователи могут применить данные системы в областях раннее не доступных из – за высоких издержек приобретения оборудования и его обслуживания.

Однако, потенциал IoT-систем еще не полностью раскрыт. Каждый день появляются новые устройства и технологии, которые могут быть включены в IoT-системы. IoT-системы могут улучшить качество жизни людей, сделать ее более удобной и безопасной, а также помочь экономить энергию и ресурсы.

Таким образом, создание MQTT-приложения для управления такими системами имеет большой потенциал и может быть очень полезным для пользователей. WEB-приложение позволит управлять IoT-системой с любого устройства с доступом в Интернет, что сделает управление еще более удобным и эффективным.

Целью проекта является разработка программного комплекса, позволяющего организовывать удобное управление IoT-системами.

Задачами проекта являются:

* анализ существующих приложений для управления IoT-системами и их особенностей;
* разработать серверную часть приложения, продумать его архитектуру;
* создать user-friendly интерфейс приложения;
* создать безопасный и быстрый канал связи между IoT-системой и сервером.

Актуальность проекта обусловлена тем, что IoT-технологии продолжают набирать популярность и становятся более доступными с каждым годом. Данный проект позволит удобно и гибко управлять этими системами, упрощая и улучшая жизнь его пользователей.

Проект включает специальные разделы:

* постановка задачи;
* проектирование веб - ресурса;
* тестирование.

# 1. Постановка задач

# 1.1 Обзор существующих аналогов

Для создания веб - сайта, необходимо изучить существующие аналоги и выделить их основные недостатки и преимущества.

Аналог представляет собой подобие какому-либо предмету, явлению, понятию. У большинства веб-сайтов существуют аналоги, и для того, чтобы не дублировать уже существующие программные средства, необходимо подробно ознакомиться с ними.

Аналоги:

* Domotix.pro.
* iRidium.

**domotix.pro**

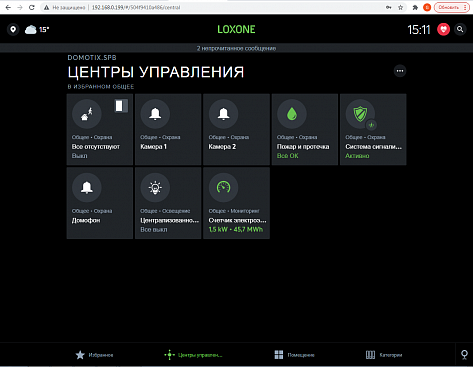


Рис. 1.1.1. – domotix.pro

Недостатки:

* Староватый и непонятный интуитивно дизайн.

Преимущества:

* Наличие встроенного плеера.
* Автоматический режим.

**iRidium**

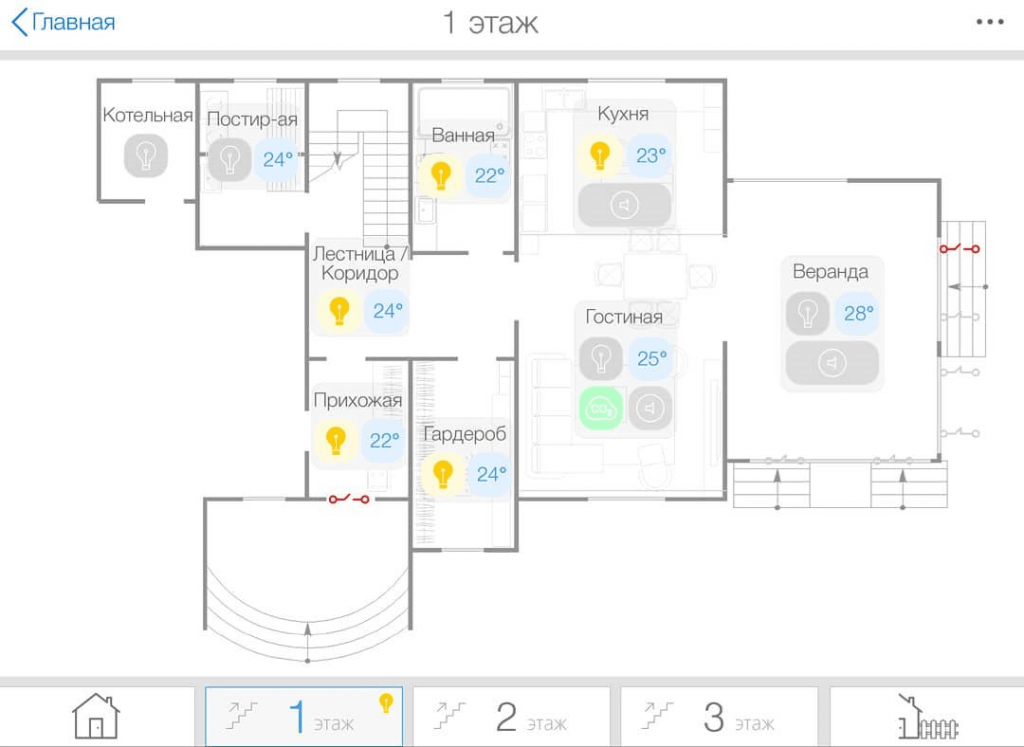


Рис. 1.1.2. – iRidium

Недостатки:

* Неудобный дизайн.

Преимущества:

* Наличие удобного режима просмотра состояний.
* Кроссплатформенность.

# 1.2 Функциональное назначение

Необходимо разработать программно-аппаратный комплекс «HomeBrain».

«HomeBrain» – это программно-аппаратный комплекс состоящий из веб-приложения, выступающего в роли интерфейса системы, сервера обработки входящих и исходящих запросов, связанных с IoT-системой, и логического модуля на базе Arduino, выступающего посредником между издателями и сервером обработки данных.

Издатель – датчик (сенсор) или устройство (девайс), подключённый к логическому модулю.

В последние годы каждый день появляется множество сайтов, поэтому для составления конкуренции другим нужно, чтобы сайт совмещал в себе функционал нескольких сайтов одновременно, что позволит привлечь максимально количество пользователей.

При работе с пользователем программное средство должно:

* быть простым и удобным в использовании;
* иметь современный дизайн и интерфейс;
* безошибочно работать;
* откликаться на действия пользователя.

В готовом виде веб-приложение должно быть максимально эргономичной и интуитивно понятным в использовании. Главной задачей любого сайта является удержание внимания пользователя, предоставление возможности применения умственных способностей и возможности отдыха при использовании сайта. Пользовательский интерфейс — совокупность способов взаимодействия пользователя с системой, обеспечивающая передачу информации между пользователем и программно-аппаратными компонентами компьютерной системы. Пользовательский интерфейс часто понимают только как внешний вид программы

К пользовательскому интерфейсу веб-приложения и веб-сайта предъявляются следующие требования:

* понятность и логичность расположения элементов интерфейса;
* эксплуатационное удобство;
* обеспечение защиты от человеческих ошибок;
* интуитивная ясность;
* современный дизайн и интерфейс.

При выполнении поставленных условий к функционалу и интерфейсу веб-приложения и веб-сайта, пользователю будет комфортно использовать его.

# 2. Проектирование программно-аппаратного комплекса

2.1 Выбор и обоснование инструментов разработки

Написание кода осуществляется в среде разработки PhpStorm.

PhpStorm – это среда разработки, позволяющая разрабатывать приложения на основе серверного языка PHP.

HTML – это язык разметки веб - страницы, который используется для создания веб - сайтов.

CSS – каскадные таблицы стилей, для оформления содержимого страницы в соответствии с описанными правилами.

PHP – это популярный язык программирования, который создан для веб-разработки. Код PHP может внедряться непосредственно в HTML. Данный язык отличается от JavaScript тем, что PHP-скрипты выполняются на сервере и создают HTML, который передаётся клиенту. PHP крайне прост для освоения, но вместе с тем способен удовлетворить запросы профессиональных программистов.

Symfony – бесплатный PHP – фреймворк с открытым исходным кодом, предназначенный для разработки веб – сайтов с использованием архитектурной модели MVC.

MySQL –самая популярная в мире база данных с открытым исходным кодом. По данным DB-Engines, MySQL занимает второе место [1] по популярности после Oracle Database. На MySQL работают многие наиболее посещаемые приложения, включая Facebook, Twitter, Netflix, Uber, Airbnb, Shopify и Booking.com.

JavaScript – Этот язык позволяет использовать сложные элементы на веб-странице, которые обеспечивают динамическое взаимодействие с пользователем при каждом изменении страницы, а не только её статическое отображение.

Webpack – сборщик модулей JS, построенный на модульной архитектуре, который позволяет приложению быть удобным как в production-режиме, так и в dev-режиме.

Knockout — свободный JavaScript каркас веб-приложений, реализующий Model-View-View-Model шаблон c образцами.

Redis –нереляционная резидентная СУБД, хранящая данные в виде пар «ключ-значение».

Arduino – комплекс аппаратно-программных средств построения и прототипирования простых систем, моделей и экспериментов в области электроники, автоматики, автоматизации процессов и робототехники.

# 2.2 Структура базы данных

На рисунке 2.2.1 представлена схема базы данных проекта. Все локации хранятся в таблице location. К каждой локации привязан издатель (сенсор/датчик или же девайс), хранящийся в таблице publisher. Таблица publisher\_description хранит в себе значение настройки (из таблицы publisher\_setting) конкретного издателя. Таблица publisher имеет связь с таблицей publisher\_value\_archieve, в которой хранятся архивные значения, присланные издателем, на основе которых выстраиваются графики. У таблицы location имеется связь с таблицей notice, что хранит уведомления по конкретному помещению. Таблица user имеет связь с таблицей user\_api, которая хранит в себе данные пользовательских плат, которые подключаются к нашей системе для получения и отправки данных и информации об издателях, которые подключены к этой плате.

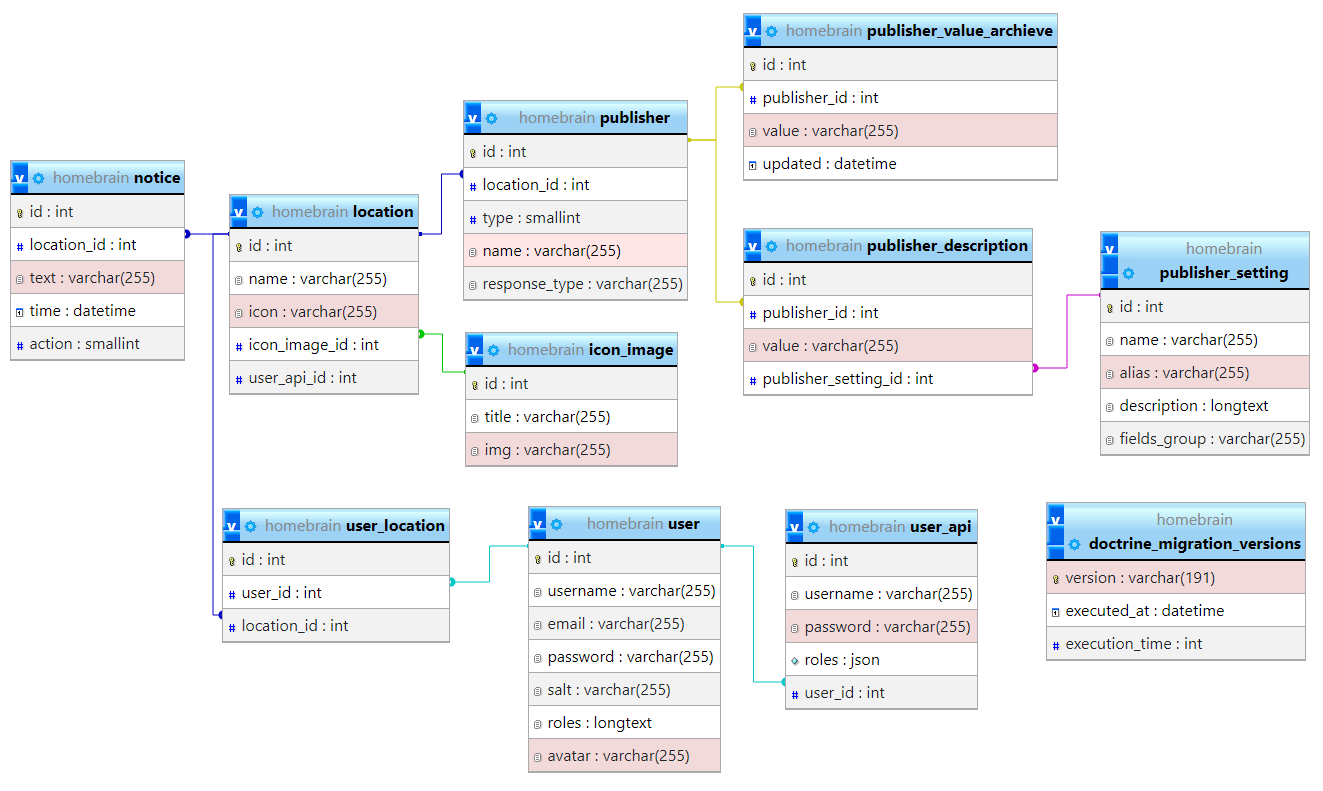


Рис. 2.2.1. – Структура базы данных приложения «HomeBrain»

# 2.3 Структура веб-ресурса

Веб-ресурс включает в себя такие разделы, как:

Страницы авторизации и регистрации (рис. 2.3.1, 2.3.2):

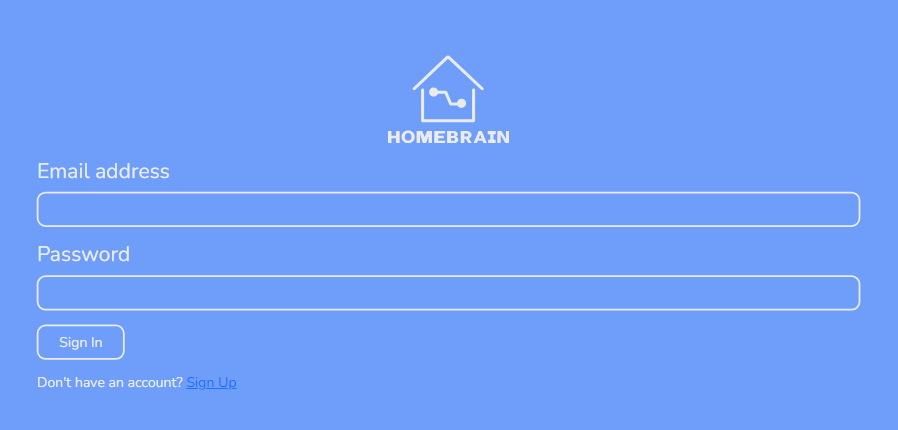


Рис. 2.3.1. – Страница авторизации «HomeBrain»



Рис. 2.3.2. – Страница регистрации «HomeBrain»

Главная (домашняя) страница (рис. 2.3.3):

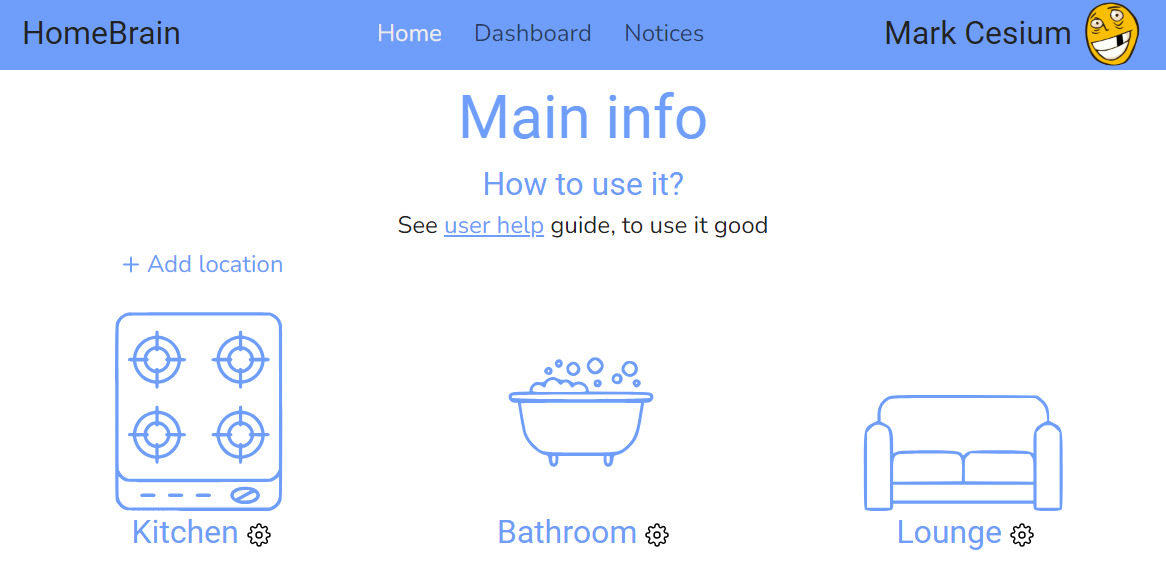


Рис. 2.3.3. – Главная (домашняя) страница «HomeBrain»

На главной странице пользователь видит все доступные ему локации, по нажатию на иконку или на название – попадает в просмотр получаемых значений от издателей (в данном случае сенсоров/датчиков) или управление издателями (девайсами). По нажатию на шестерёнку, пользователь попадает на страницу редактирования комнаты.

Страница локации (рис. 2.3.4, 2.3.5):

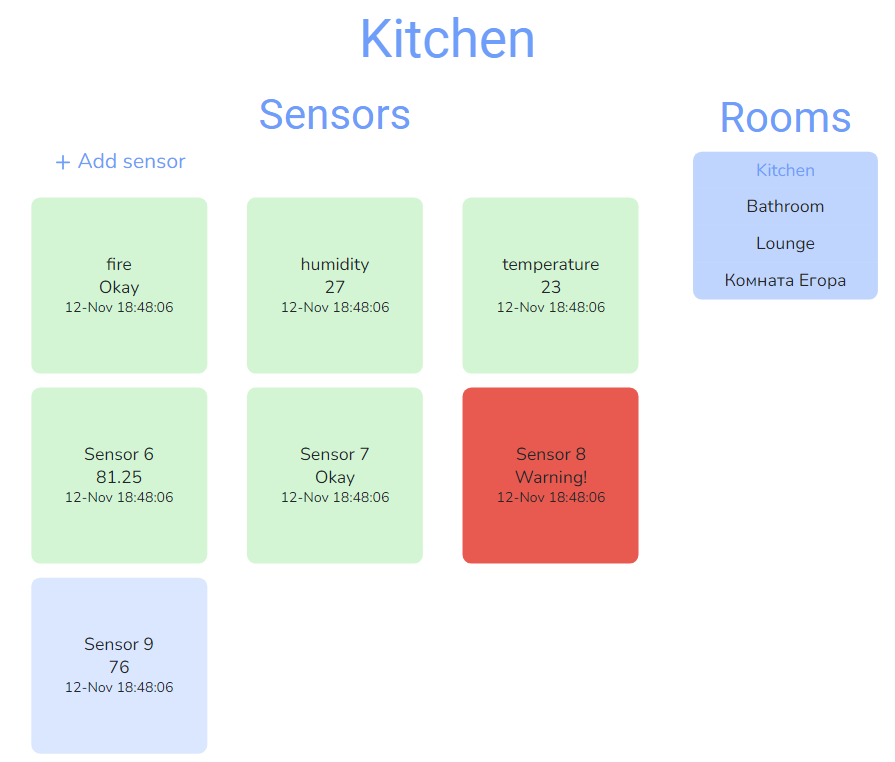


Рис. 2.3.4. – Страница локации «HomeBrain». Раздел «Sensors»

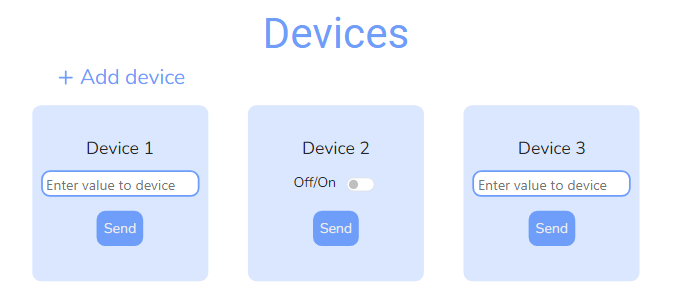


Рис. 2.3.5. – Страница локации «HomeBrain». Раздел «Devices»

На странице локации у пользователя два раздела: сенсоры и девайсы. Карточки сенсоров содержат в себе название, значение, выводимое в зависимости от настроек, дату последнего обновления значения. Также карточка подсвечивается определённым цветом, в зависимости от прохождения условия валидации значений, выставляемых пользователем. Карточки девайсов содержат в себе название девайса, поле ввода значения либо переключатель, в зависимости от настройки, кнопку отправить.

Теперь, попробуем добавить нового издателя, например, датчик температуры духовки (рис. 2.3.6).

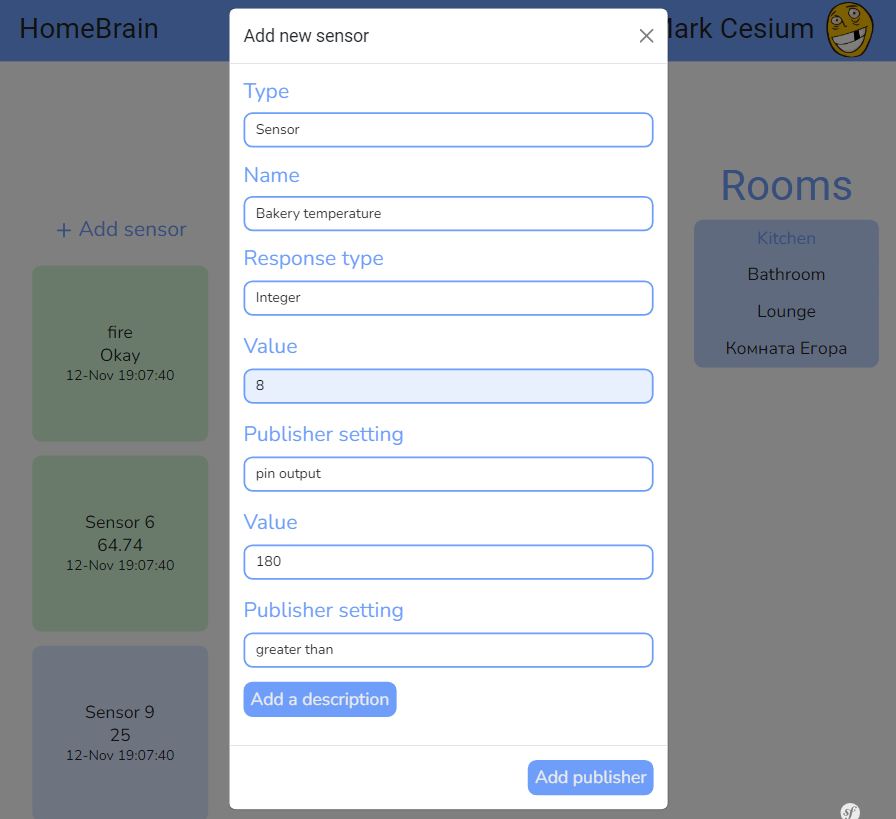


Рис. 2.3.6. – Модальное окно создания издателя

Мы выбираем тип (Sensor/Device, при нажатии на соответствующие кнопки система сама выбирает тип), указываем имя, выбираем тип данных, который мы хотим получать от данного сенсора (Integer/Float/Boolean), указываем свои настройки, в данном случае мы указали pin на выход и валидацию значения температуры (greater than – 180, карточка будет подсвечена зелёным при прохождения этого условия). Вот так гибко и просто происходит создание и первоначальная настройка.

Страница графиков:

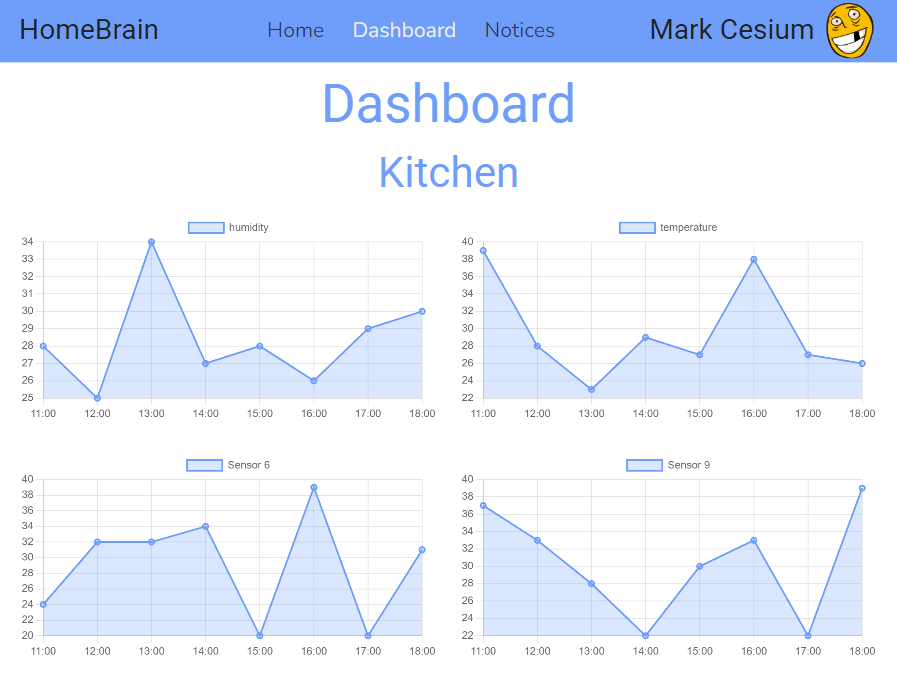


Рис. 2.3.7. – Страница графиков «HomeBrain»

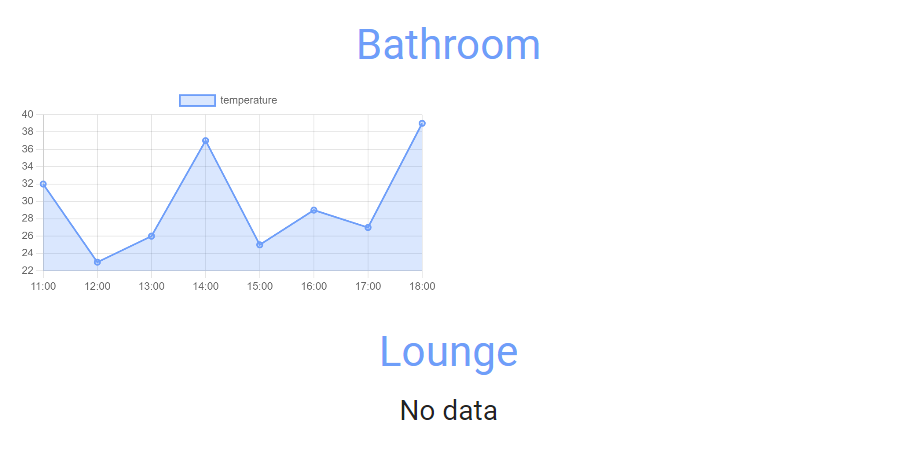


Рис. 2.3.8. – Страница графиков «HomeBrain»

На этой странице пользователь может просматривать данные, присылаемые издателями-сенсорами, с типом получаемых данных Integer или Float, в графическом виде за промежуток времени. Если же в комнате нет таких издателей, то будет выведена надпись «No data».

Страница уведомлений (рис. 2.3.9):



Рис. 2.3.9. – Страница уведомлений «HomeBrain»

На данной странице отображаются уведомления о истории изменений настроек (создание и удаление издателей, добавления локаций, и др). Уведомления содержат информативную иконку, текст уведомления, дату получения уведомления.

Страница личного кабинета (рис. 2.3.10, 2.3.11):



Рис. 2.3.10. – Страница личного кабинета «HomeBrain»

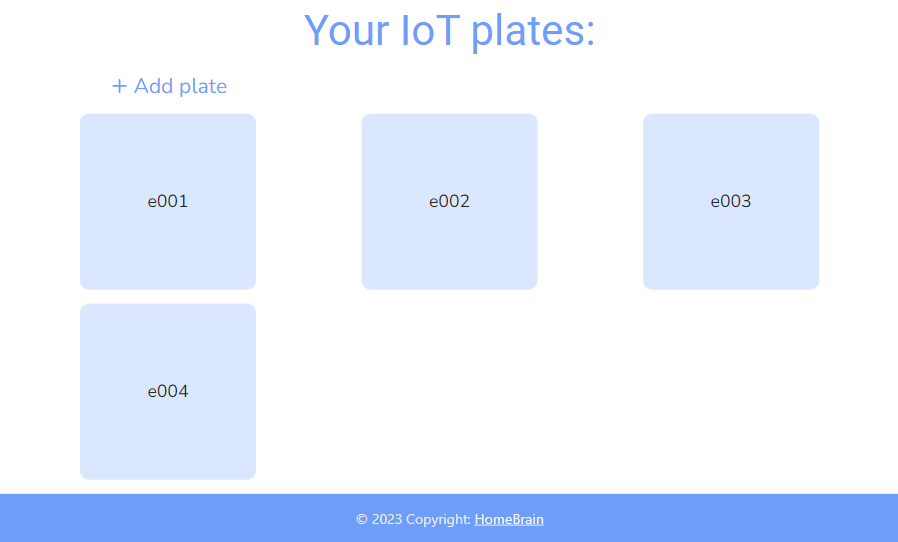


Рис. 2.3.11. – Страница личного кабинета «HomeBrain»

В личном кабинете пользователь может менять свои данные и добавлять новые платы (таблица user\_api).

На рисунке 2.3.12 представлен вид получаемых данных на микроконтроллере ArduinoMEGA

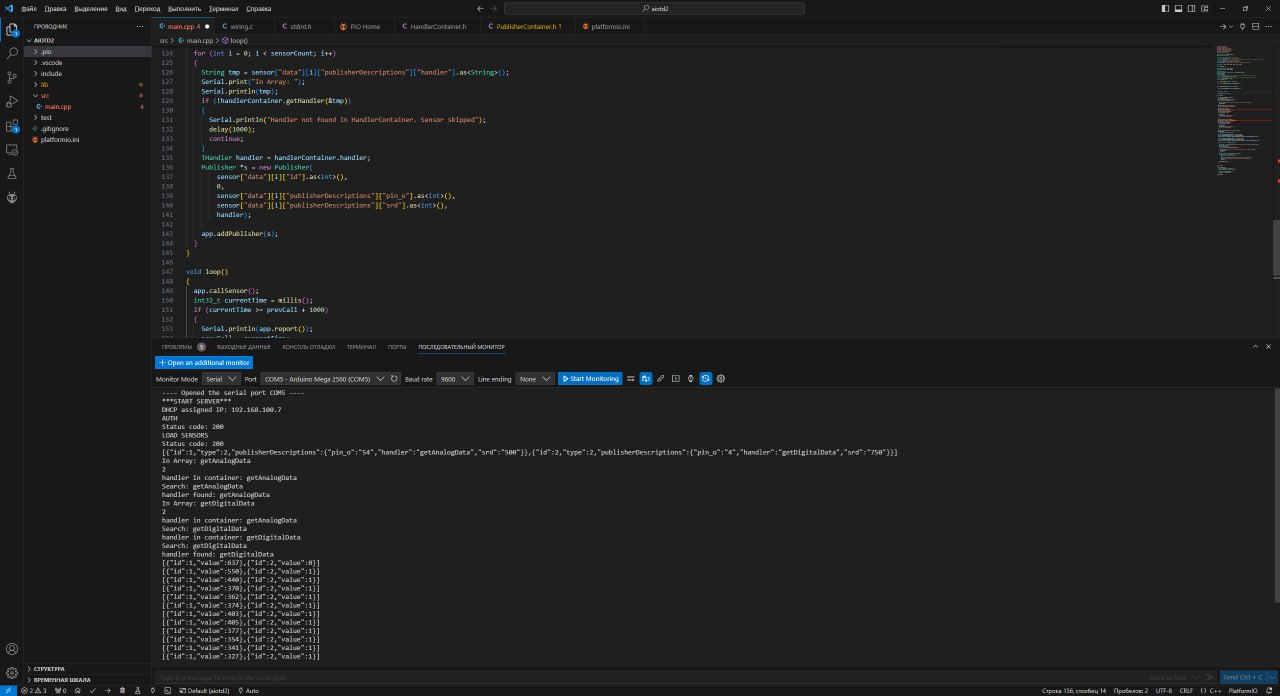


Рис. 2.3.12. - Вид получаемых данных на микроконтроллере Arduino MEGA

# 3. Тестирование

Тестирование программного обеспечения — процесс анализа программного обеспечения и его документации с целью обнаружения дефектов и повышения качества продукта.

В ходе тестирования были воспроизведены следующие случаи для проверки пользовательского интерфейса. Результаты тестирования приведены в таблице 3.1

Таблица 3.1 – Тестирование всех страниц сайта

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Шаги | Ожидаемый результат | Выполнение |
| 1 | 2 | 3 | 4 |
| Запуск всех страниц сайта | Запуск исполнимого файла | Успешный запуск файла | Пройден |
| Оптимизировать размеры контейнеров всех страниц сайта | Все компоненты находятся на своём месте | Все компоненты находятся на своём месте | Пройден |
| Переход на любую страницу сайта при помощи навигации | Нажать по очереди кнопки навигации | Переход на указанную страницу | Пройден |
| Правильное расположение футера на каждой странице | Запустить каждую страницу интернет-музея | Футер расположен под основной частью | Пройден |
| Отображение карточек, расположенных упорядоченно на каждом из столбцов | Запуск главной страницы | Карточки расположены упорядоченно | Пройден |

Все функциональные требования были выполнены и протестированы в соответствии с поставленным заданием.

# 4. Инструкция по установке

Для работы с веб-сайтом, персональный компьютер пользователя должен соответствовать минимальным системным требованиям перечисленных ниже программных средств.

Процесс установки:

* установка LAMP, Linux, Apache 2, или аналог, PHP 8+, MySQL 8+;
* установка PHP зависимостей (composer install);
* создание базы данных с помощью Doctrine ORM (php bin/console doctrine:migrations:diff, php bin/console doctrine:migrations:migrate);
* установка Redis;
* установка NodeJS, npm;
* установка js-зависимостей (npm install);
* компиляция js и scss скриптов, сборка frontend исходников проекта (npm run build);
* импорт данных в базу данных;
* прошить ваш микроконтроллер.

# 5. Руководство

Первым этапом является регистрация. Проходим регистрацию (рис. 5.1):



Рис. 5.1. – Этап регистрации

После прохождения регистрации ресурс просит нас авторизоваться. Авторизуемся (рис. 5.2):



Рис. 5.2. – Этап авторизации

После этого мы попадаем на главную (домашнюю) страницу, на которой должны располагаться локации, к которым мы имеем доступ. Но сейчас на странице мы увидим следующее (см. рис. 5.3):

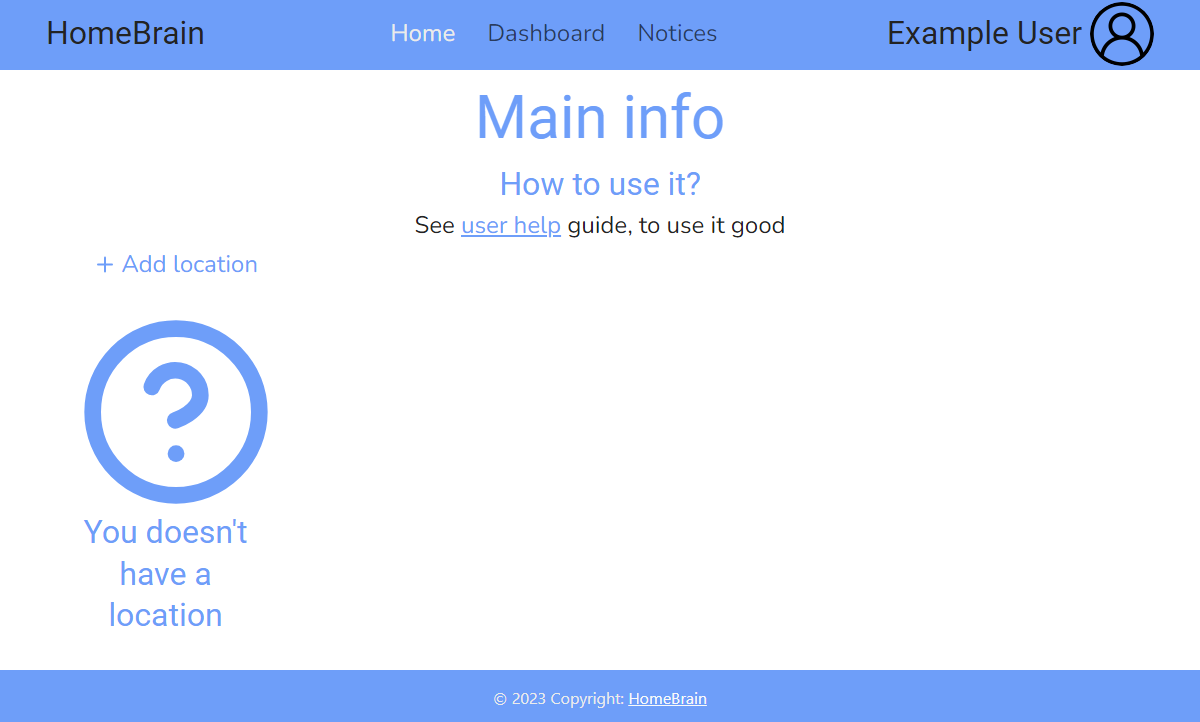


Рис 5.3. – Главная страница сразу после регистрации

Нам предлагают перейти на страницу с руководством по использованию, а вместо карточек с локациями мы видим карточку, что говорит нам о том, что мы пока не создали ни одной локации.

Для создания локации нам необходимо дать ей название, картинку (формата png, svg или jpg, размерностью до 512kB) из локального хранилища или же выбрать её из предложенных вариантов, а также плату. Плата создаётся в личном кабинете пользователя. Она необходима для подключения вашей IoT-системы к нашему сервису, чтобы иметь возможность получать и отправлять данные. Давайте создадим плату, перейдя в личный кабинет пользователя, нажав на имя пользователя, и опустимся в самый низ страницы (см. рис. 5.4):

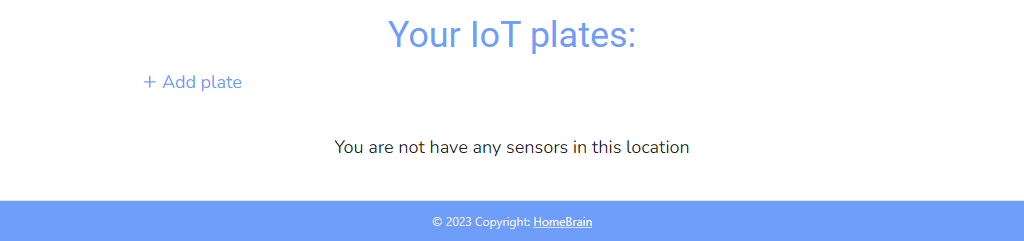


Рис 5.4. – Раздел с платами пользователя

Так как у нас новый аккаунт, то никаких плат у нас пока-ещё нет. Перейдём к непосредственному созданию платы, нажав на кнопку «Add plate» (см. рис. 5.5):

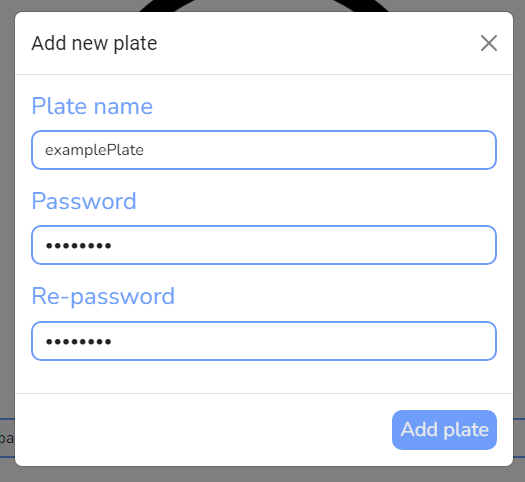


Рис. 5.5. – Модальное окно создание платы

Нас встречает модальное окно, в котором нас просят заполнить поля имени и пароля. Нужно обязательно запомнить пароль, ведь именно по валидным имени и паролю плата будет получать JWT, по которому она будет иметь доступ к получению и отправке данных. Нажмём на кнопку «Add plate» для создания платы и получим следующий результат (см. рис. 5.6):

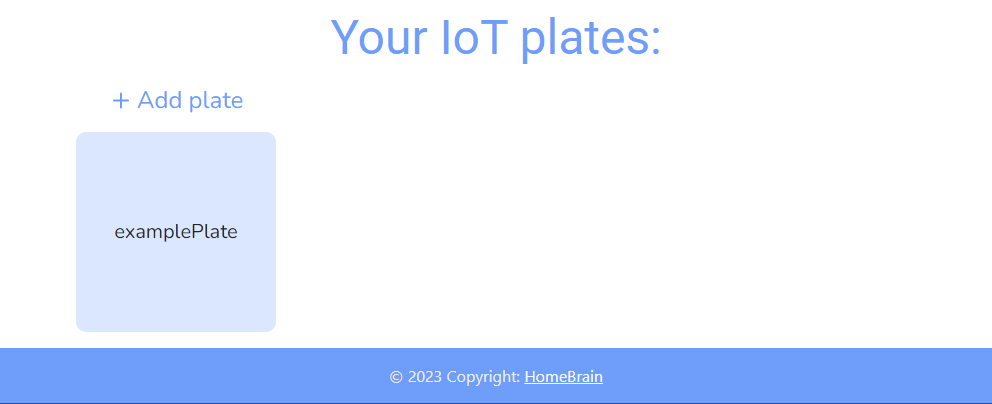


Рис. 5.6. – Результат добавления платы

Как мы можем видеть во всё том же разделе, мы успешно добавили плату. Теперь мы имеем возможность создать локацию. Вернёмся на главную страницу и нажмём на кнопку «Add location», для добавления новой локации, заполним некоторые поля тестовыми данными (см. рис. 5.7):

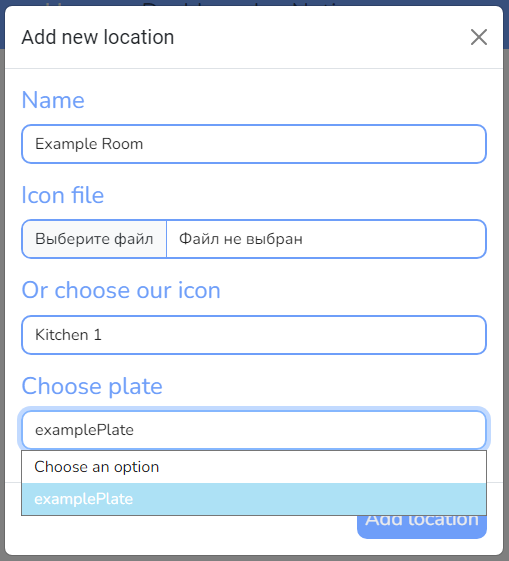


Рис. 5.7. – Модальное окно создания комнаты

Заполняем поле ввода имени, выбираем картинку комнаты (в данном случае я выбрал заготовленную сервисом иконку «Kitchen 1»), выбираем плату (это поле также является обязательным, именно поэтому до создания платы мы не могли создать комнату). Нажмём на кнопку «Add location» и увидим следующий результат (см. рис. 5.8):

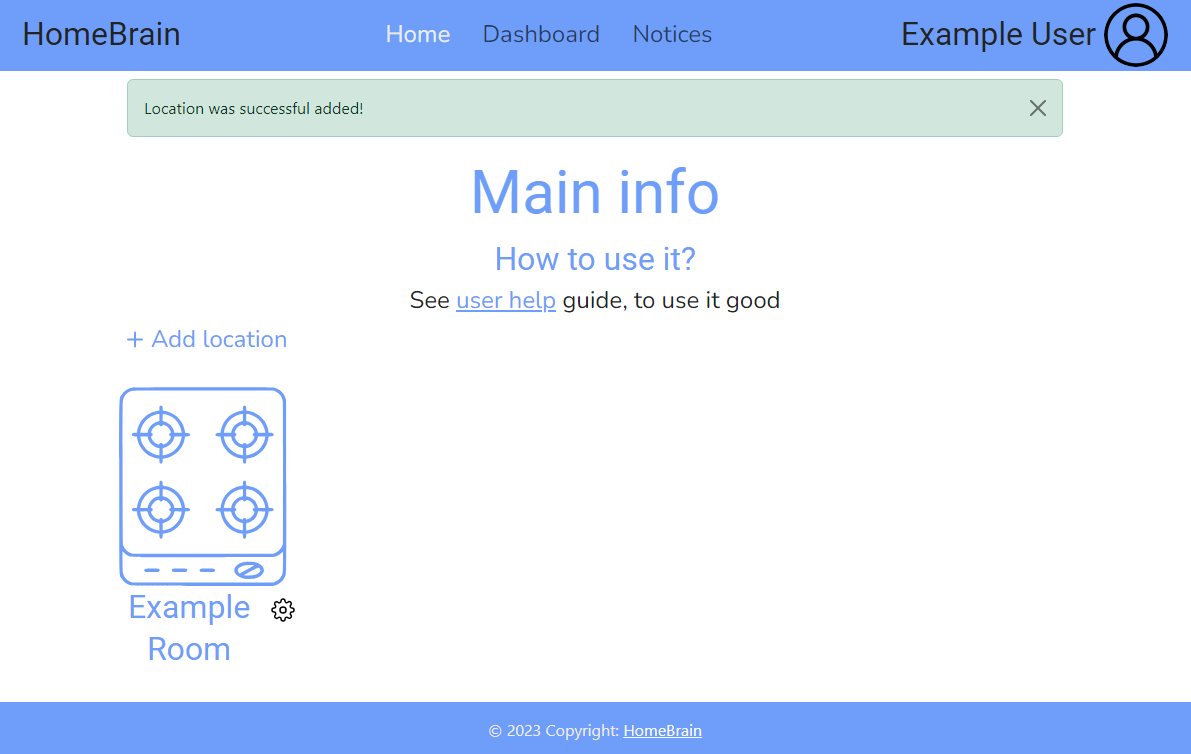


Рис. 5.8. – Результат создания комнаты

Мы можем видеть всплывающее уведомление о том, что локация успешно создана (в противном случае, при заполнении не валидных данных, мы бы получили всплывающее уведомление с содержанием ошибки).

Теперь перейдём в саму локацию, нажав на её название или иконку (не шестерёнку, это важно, об этом далее в руководстве) (см. рис. 5.9):

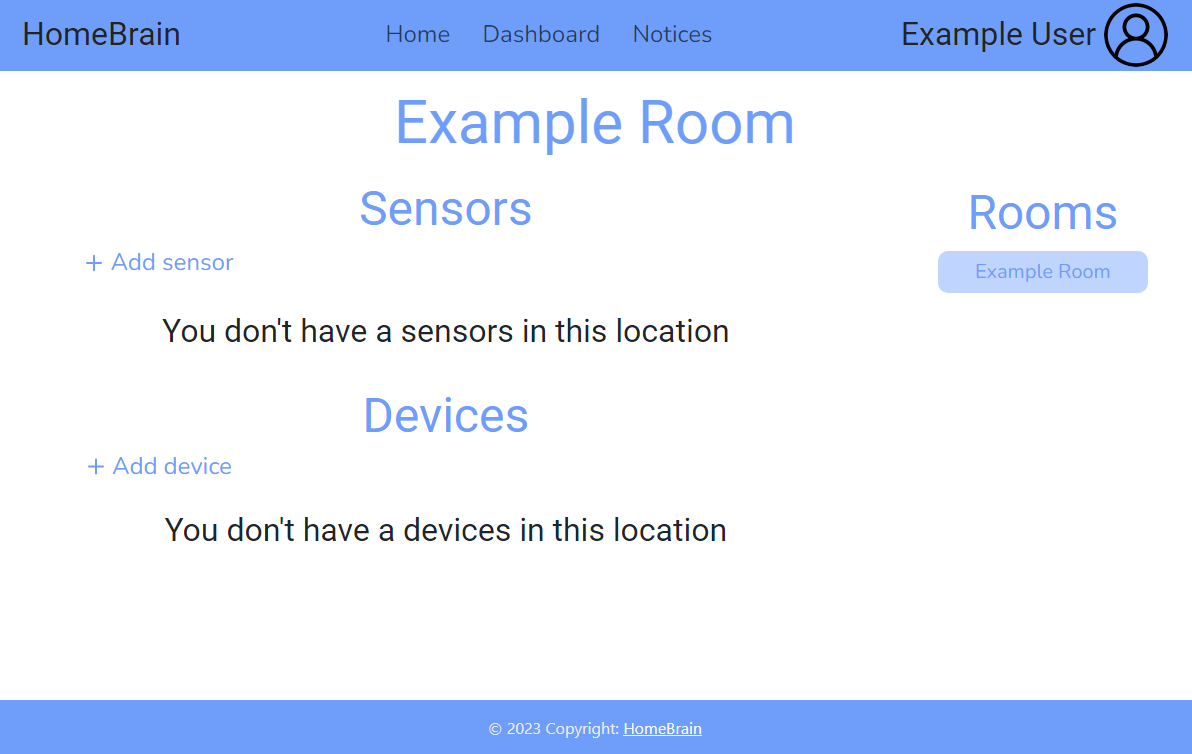


Рис. 5.9. – Обзор созданной комнаты

Здесь три основных раздела: сенсоры, девайсы, комнаты. В первых двух в будущем будут находиться карточки сенсоров или девайсов соответственно, в последнем у нас расположен виджет для быстрого переключения между локациями, которые есть у пользователя (в данном случае у нас имеется только одна локация). Сенсоры и девайсы для нашей системы – издатели. Логика их работы частично схожа. Давайте рассмотрим пример создания издателя на примере создания сенсора. Для создания сенсора нажмём кнопку «Add sensor» (см. рис. 5.10):

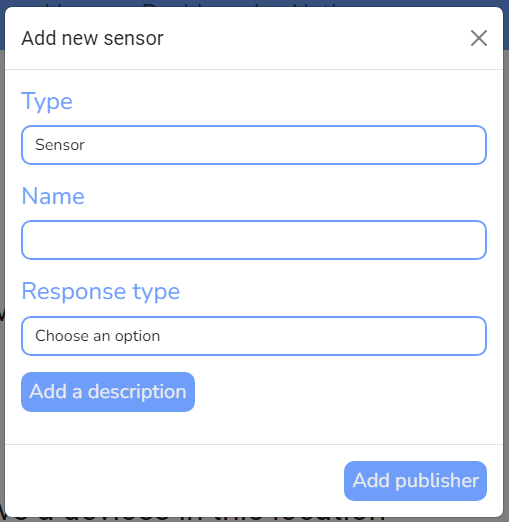


Рис. 5.10. – Модальное окно создания издателя (сенсора)

При создании нового издателя система автоматически присваивает указанный тип: датчик или устройство.

Для создания сенсора необходимо указать его название, возвращаемый response typeб т.е. тип данных (Integer – целое число, Float – дробное число, Boolean – да или нет), а также добавить «description», то есть описание (необязательно).

Настройки издателя подразделяются на несколько типов: activation и validation.

Первая группа настроек необходима для настройки работы издателя:

* пин на вход – номер пина, с которого необходимо считывать значения (в системе - pin\_i, в HomeBrain – pin input);
* пин на выход – номер пина, на который необходимо подать напряжение (в системе – pin\_o, в HomeBrain – pin output);
* хэндлер - функция, обрабатывающая приходящие значения с платы перед отправкой (в системе – handler, в HomeBrain – handler);
* время работы девайса – время, в течение которого будет работать девайс (в системе – d\_rt, в HomeBrain – device runtime);
* пин девайса на выход – пин, на который необходимо подать напряжение при срабатывании «опасного» хэндлера (в системе d\_pin\_o, в HomeBrain – device pin out);
* период опроса сенсора – задержка между запросами к сенсору за получением данных (в системе – srd, минимальное значение – 100 мс, в HomeBrain – sensor request delay).

Вторая группа настроек необходима для валидации значений, отправляемых издателем:

* больше чем (в HomeBrain – greater than) – для прохождения валидации значение датчика должно быть больше, чем указанное в сценарии;
* меньше чем (в HomeBrain – less than) – для прохождения валидации значение датчика должно быть меньше, чем указанное в сценарии;
* равно (в HomeBrain – equals) – для прохождения валидации значение датчика должно быть равно указанному в сценарии;
* не равно (в HomeBrain – not equals) – для прохождения валидации значение датчика должно быть не равно указанному в сценарии;
* в промежутке (в HomeBrain – values range) – для прохождения валидации значение датчика должно быть между начальным и конечным числами, которые указаны в сценарии;
* danger reaction (реакция на опасность) функция, которая будет активирована при не прохождении валидации (в HomeBrain – danger reaction).

Валидация работает по следующему принципу: если значение, присылаемое датчиком, не удовлетворяет сценарию – сервер определяет показания как не валидное, интерфейс подсвечивает блок датчика красным цветом, а логический модуль получает команду на активацию реакции на данное событие. При прохождении валидации блок будет подсвечен зелёным. В случае отсутствия настроек валидации блок отображается синим. Если, к примеру будет указан Response type со значением «Boolean», но не указана ни одна настройка из группы validation, то будет получена следующая ситуация на странице локации (см. рис. 5.11):



Рис 5.11. – Пример последствия указания неправильных настроек

Будьте внимательны при настройке!

Валидация проходит на стороне сервера при поступлении новых значений от издателя. Если присылаемое значение не проходит валидацию, то сервер отправляет обратно название (то есть указатель) хэндлера на случай не прохождения валидации (если такой имеется).

На данном этапе заполним поле имени со значением «Example Sensor» и выберем Response type со значением «Integer». Нажмём на кнопку Add publisher и увидим следующий результат (см. рис. 5.12):

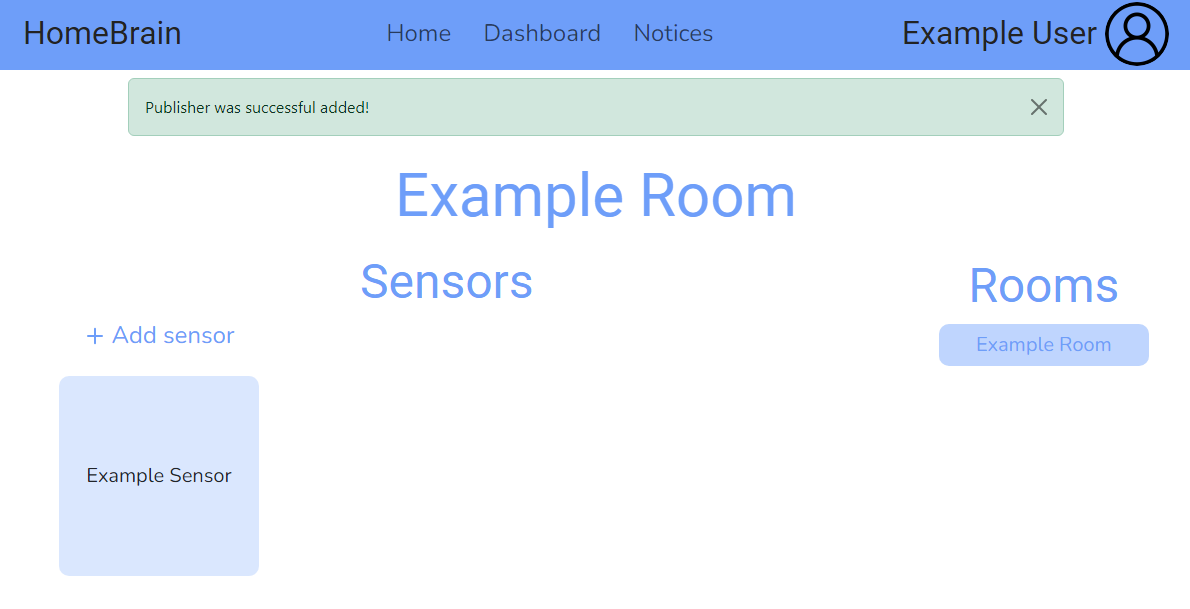


Рис. 5.12. – Результат добавления сенсора

После данной операции карточка сенсора содержит только его название. Чтобы получать какие-либо данные от этого сенсора нужно перезагрузить плату для обновления настроек издателей. Плата автоматически загрузит необходимые настройки.

# Раздел настройки логического блока

Программа для логического блока требует минимальную конфигурацию перед применением. Для базовой настройки необходимо указать логин и пароль логического блока, зарегистрированного в приложении. Общий процесс инициализации логического блока выглядит следующим образом:

1. Логический блок отправляет POST-запрос на авторизацию серверу загрузки данных издателей. В качестве параметров запрос отправляет username и password, зарегистрированного логического блока.
2. После успешной авторизации логический блок принимает JSON Web Token (JWT). Данный токен будет необходим для всех взаимодействий с сервером обработки данных.
3. Блок обращается к серверу загрузки данных для получения информации об издателях всех видов.
4. После получения данных об издателях происходит инициализация сенсоров и девайсов.
5. Начинается цикличный опрос датчиков согласно переданным настройкам. Одновременно блок ожидает команд для устройств.

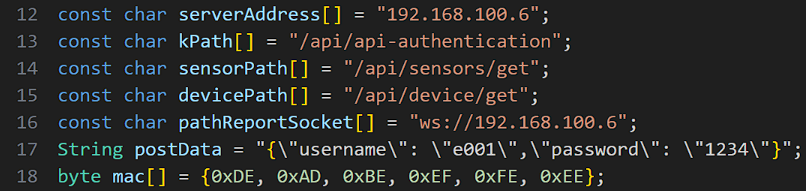


Рис ХХХ – Конфигурация логического блока

* serverAddress – адрес сервера загрузки данных издателей (необходимо изменить только при локальной установке сервера);
* kPath – адрес аутентификации логического блока;
* sensorPath – путь для загрузки списка издателей-сенсоров;
* devicePath – путь для загрузки списка издателей-девайсов;
* pathReportSocket – адрес сервера обработки данных (необходимо изменить только при локальной установке сервера);
* postData – логин и пароль логического блока;
* mac – mac-адрес устройства.

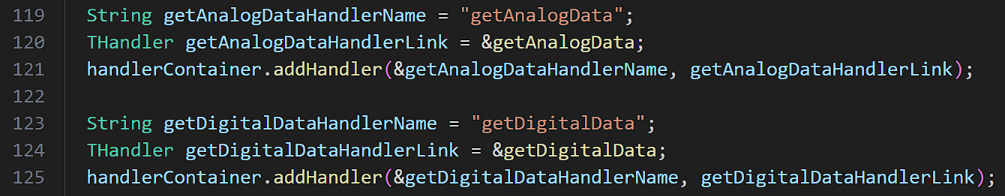


Рис. Ччч – Добавление функции обработчика издателя – сенсора в сервис – контейнер.

В случае необходимости добавления своего обработчика издателя (сенсора) это можно сделать следующим образом:

1. Создайте строковое название вашей функции.
2. Создайте указатель на вашу функцию обработчик.
3. Воспользуйтесь методом addHandler сервис-контейнера и в качестве параметра передайте указатель на функцию и строковое название.

Теперь данный обработчик доступен в качестве обработчика данных для любого выбранного вами датчика.

# ЗАКЛЮЧЕНИЕ

В результате выполнения проекта было разработан программно-аппаратный комплекс «HomeBrain» для управления и мониторинга IoT-системы.

В ходе реализации комплекса были выполнены следующие задачи:

* разработано программное обеспечение для логического блока;
* разработаны серверные части комплекса для обработки входных данных и выходных данных логического модуля и веб – приложения;
* разработан интерфейс веб – приложения, учитывающий особенности дизайна;
* проведено тестирование комплекса.
* описана инструкция по настройке и использования проекта.

В данном проекте были описаны такие разделы, как постановка задачи, проектирование, программная реализации, тестирование, применение, руководство пользователя.

Цели разработки были достигнуты, программно-аппаратный комплекс соответствует всем требованиям.

# Список использованных источников

1. Информационный веб-сайт DB-Engines [Электронный ресурс]. ­­­– Режим доступа: https://db-engines.com/en/ranking/ Дата доступа: 03.11.2023
2. Информационный веб-сайт MySQL [Электронный ресурс]. – Режим доступа: https://www.mysql.com.
3. Информационный веб-сайт PHP [Электронный ресурс]: Режим доступа: https://www.php.net.
4. Информационный веб-сайт фреймворка Symfony [Электронный ресурс]: Режим доступа: https://symfony.com.
5. Современный учебник JavaScript [Электронный ресурс]. - Режим доступа: // https://learn.javascript.ru.
6. Справочник веб технологий [Электронный ресурс]: Режим доступа: https://html5book.ru.
7. Информационный веб-сайт Arduino [Электронный ресурс]: Режим доступа: <https://arduino.ru>.
8. https://jwt.io/