

Министерство науки и высшего образования РФ  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
**«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Космических и информационных технологий**

институт

**Вычислительная техника**

кафедра

**УТВЕРЖДАЮ**

Заведующий кафедрой

О. В. Непомнящий

подпись инициалы, фамилия

« \_\_\_\_ » \_\_\_\_ 2020 г.

**БАКАЛАВРСКАЯ РАБОТА**

09.03.01 Информатика и вычислительная техника

код и наименование направления

Программно-аппаратный комплекс управления комфорностью среды  
тема

Пояснительная записка

Руководитель

\_\_\_\_\_

подпись, дата

профессор,

канд. техн. наук

должность, ученая степень

Б. И. Борде

инициалы, фамилия

Выпускник

\_\_\_\_\_

подпись, дата

А.Р. Исмагилов

инициалы, фамилия

Нормоконтролер

\_\_\_\_\_

подпись, дата

профессор,

канд. техн. наук

должность, ученая степень

Б. И. Борде

инициалы, фамилия

Красноярск 2020

Министерство науки и высшего образования РФ  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
**«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Космических и информационных технологий

институт

Вычислительная техника

кафедра

УТВЕРЖДАЮ

Заведующий кафедрой

О. В. Непомнящий

подпись инициалы, фамилия

« \_\_\_\_ » \_\_\_\_\_ 2020 г

**ЗАДАНИЕ**  
**НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ**  
**в форме бакалаврской работы**

Студенту Исмагилову Антону Руслановичу  
фамилия, имя, отчество

Группа КИ16-06Б Направление (специальность) 09.03.01

номер

код

Информатика и вычислительная техника

наименование-

Тема выпускной квалификационной работы Программно-аппаратный комплекс управления комфорtnостью среды

Утверждена приказом по университету № \_\_\_\_\_ от \_\_\_\_\_

Руководитель ВКР Борде Б. И., канд. техн. наук., профессор, кафедра ИВТ

инициалы, фамилия, должность, ученоe звание и место работы

Исходные данные для ВКР Провести анализ известных систем управления комфорtnостью среды, разработать оптимальную систему управления параметрами микроклимата помещения с поддержкой аппаратной составляющей

Перечень разделов ВКР Анализ предметной области и существующих решений систем управления комфорtnостью, проектирование системы программно-аппаратного комплекса, реализация программного обеспечения программно-аппаратного комплекса, написание инструкции пользователя разработанной системы

Перечень графического материала презентация в формате PowerPoint

Руководитель ВКР

подпись

Б. И. Борде

инициалы и фамилия

Задание принял к исполнению

А. Р. Исмагилов

подпись, инициалы и фамилия студента

« \_\_\_\_ » \_\_\_\_\_ 2020 г.

## **РЕФЕРАТ**

Выпускная квалификационная работа по теме «Программно-аппаратный комплекс управления комфорностью среды» содержит 65 страниц текстового документа, 5 таблиц, 31 иллюстрацию, 23 использованных источников.

АВТОМАТИЗАЦИЯ, УМНЫЙ ДОМ, ПРОГРАММА,  
МИКРОКОНТРОЛЛЕР, КОМФОРТ, СЦЕНАРИИ, ДАТЧИКИ,  
ИСПОЛНИТЕЛЬНЫЙ УСТРОЙСТВА.

Цели выпускной квалификационной работы:

- создание доступной системы управления комфорностью среды поддерживающей аппаратную составляющую;
- обеспечение удобной автоматизации выполнения задач основных систем управления комфорностью необходимых пользователю;

В результате выполнения выпускной квалификационной работы была определена структура веб-приложения управления комфорностью среды, установлены функциональные требования, выбраны средства реализации, произведена разработка клиентской и серверной части приложения, а также разработано приложение для отладки аппаратной части. Разработано веб-приложения поддерживающее аппаратную часть, регулирующее параметры комфорта необходимые пользователю по средствам создания сценариев различного типа.

## **СОДЕРЖАНИЕ**

Введение.....	4
1 Анализ предметной области .....	8
1.1    Постановка задачи .....	8
1.2    Функциональные возможности .....	9
1.3    Анализ существующих решений.....	10
1.3.1 Xiaomi.....	10
1.3.2 IKEA .....	11
1.3.3 Apple .....	12
1.3.4 Samsung .....	12
1.4    Определение параметров комфортности.....	13
1.5    Вывод по главе .....	14
2 Проектирование системы.....	15
2.1 Определение функциональных требований.....	15
2.1.1 Работа с пользователями .....	16
2.1.2 Работа с устройствами.....	16
2.1.3 Работа со сценариями .....	18
2.1.4 Работа с логом событий.....	19
2.1.5 Диаграмма прецедентов .....	19
2.2 Определение структуры приложения .....	21
2.2.1 Структура контроллеров .....	23
2.2.2 Структура представлений .....	24
2.2.3 Структура моделей .....	26
2.3 Выбор средства реализации .....	28
2.3.1 Выбор программной платформы для реализации web-клиента	28
2.3.2 Выбор аппаратной платформы для реализации .....	29

2.3.3 Выбор модели представления данных.....	30
2.3.4 Выбор протокола.....	32
2.3.5 Выбор средств разработки .....	33
2.4 Вывод по главе .....	33
3 Реализация приложения .....	34
3.1 Разработка серверного программного обеспечения.....	34
3.1.1 Разработка системы аутентификации .....	34
3.1.2 Разработка сервиса передачи данных между сервером и устройствами .....	35
3.1.3 Обработка сценариев .....	37
3.2 Разработка web-клиента .....	39
3.2.1 Разработка контроллеров .....	39
3.2.2 Разработка представлений .....	42
3.2.3 Определение моделей данных .....	48
3.3 Разработка приложения для аппаратной части.....	51
3.4 Вывод по главе .....	52
4 Инструкция пользователя .....	53
Заключение .....	63
Список использованных источников .....	65

## **ВВЕДЕНИЕ**

Наша жизнь полна рутины, одинаковых действий, выполняемых изо дня в день в один и тот же промежуток времени. Каждый из нас перед выходом из дома проверяет закрыты ли окна, выключен ли свет, перекрыта ли вода. Если в доме есть питомец – нельзя забывать о его питании. То же самое можно сказать и о растениях – если не позаботиться о них должным образом, то они могут засохнуть. Но повседневность сегодня такова, что по приходу домой единственное желание, которое у нас появляется – это лечь спать, чтобы побыстрее добраться до выходных. Но даже здесь появляются трудности – вы уже легли, но забыли выключить лампу, а выключатель слишком далеко, шторы не закрыты, а свет фонарного столба как на зло бьет вам прямо в глаза. Эти действия могут негативно сказываться как и на общей жизнедеятельности человека, так и на его психологическом состоянии, на его здоровье и уровне продуктивности. Время – один из главных ресурсов, оно расходуется куда быстрее, пока мы выполняем все эти действия день ото дня.

Развитие науки и техники, а также сетей Интернет, создало новое направление продуктов, тесно связанных с нашей повседневной жизнью. Речь идет о концепции «Умный дом» и «Интернета вещей». Эти концепции неразрывно связаны. Так как в доме могут быть находятся вещи, которые связаны между собой и непосредственно связаны с домом (помещением), в котором они находятся. Следующим шагом в развитии этого направления стало развитие решений задач автоматизации – процесс выполнения определенных повседневных задач без участия человека.

Автоматизация включает в себя решение таких задач, как:

- управление источниками света;
- коррекция работы отопительной системы;
- уведомление об угрозе жизни (вторжение, возгорание, протечка воды).

Современные системы дают пользователям возможность гибкой настройки под свои нужды и предлагают широкий спектр датчиков и исполнительных устройств. Несмотря на разных производителей подобных систем, существуют основные типы устройств, присутствующих в каждой системе и решают выше представленные задачи.

Основными устройствами системы автоматизации являются:

- **Головное устройство** – устройство, регулирующее входные данные и исполнительные устройства, связывая их в единую сеть и обеспечивая доступ в сеть интернет для доступа откуда угодно;
- **Датчики** – устройства, позволяющие системе считать текущее состояние внешней среды;
- **Исполнительные устройства** – устройства позволяющие выполнять действия на основе поступающей команды или изменения состояний внешней среды.

Связь может осуществляться как по проводным сетям Ethernet, так и по беспроводным сетям (Wi-Fi), при этом датчики и исполнительные устройства делятся на три основных типа: систему безопасности представленную в таблице 1, систему управления освещением представленную в таблице 2 и систему управления климатом представленную в таблице 3, которые решают три основные задачи. Данные задачи решаются системами автоматизации, и в таблицах можно увидеть различные предлагаемые алгоритмы работы.

Таблица 1 – Элементы системы безопасности

Датчики	Алгоритмы
Датчики движения, датчики присутствия, датчики вибрации, датчики разбития стекла, датчики открытия окна или двери	Регистрация нежелательного проникновения уведомление владельцев
Видеонаблюдение, Видеодомофоны и видео глазки	Запуск видеосъемки
Сирены	Включение сирены
Электронные замки и модули управления воротами	Запирание входных или межкомнатных дверей

Таблица 2 – Элементы системы управления освещением

Датчики	Алгоритмы
Умные выключатели и диммеры	Автоматически регулировать освещенность в зависимости от сезона и времени суток или по другим заранее заданным правилам
Датчики освещенности	Автоматически поддерживать освещенность на постоянном уровне, регулируя яркость светильников и положение жалюзи или штор
Модули управления шторами, жалюзи и рольставнями	Автоматически включать свет, когда люди входят в помещение, и выключать, когда выходят
Датчики движения и присутствия	Автоматически включать свет, когда люди входят в помещение, и выключать, когда выходят

Таблица 3 – Элементы системы управления климатом

Датчики	Алгоритмы
Датчики влажности	Автоматически поддерживать влажность, комфортную для людей и подходящую для помещения и предметов обстановки
Датчики температуры	Автоматически поддерживать комфортную температуру в помещениях, где находятся люди
Терmostаты для поддержания постоянной температуры или её автоматического регулирования	Автоматически снижать мощность батарей и кондиционеров в отсутствие людей и ночью
Гигростаты для поддержания постоянной влажности или её регулирования	Автоматически вентилировать помещения и очищать воздух, поддерживая комфортное качество воздуха

При разработке подобных систем нужно учитывать такие факторы, как безопасность соединения и передачи данных, а также стоимость комплектующих, используемых при проектировании системы.

Разработка данных систем является актуальным направлением в проектировании и разработке, так как рынок систем автоматизации постоянно растет. По данным опроса портала Hi-Tech Mail.ru [1]:

- 88% россиян знают, что такое система автоматизации;
- 27% пользуются такими технологиями;
- 58% используют систему управления освещением;
- 50% видеонаблюдение;

- 41% климат контроль;
- 39% пожарная и аварийная сигнализация.

Рассматривая вопрос цены, внедрение систем автоматизации для 40% респондентов обошлось от 5 тысяч до 20 тысяч рублей и более 20 тысяч рублей потратило 33% опрашиваемых. Заплатить за дополнительный комфорт от 5 тысяч до 20 тысяч готовы 57% опрошенных. Таким образом, разработка доступной системы автоматизации, которая смогла бы обеспечить решение актуальных задач, является перспективным направлением. Так же подобные системы могут использоваться не только при домашним использовании, но и на коммерческих предприятиях, что обеспечит безопасность и комфорт человека не только на работе, но и на рабочем месте.

# **1 Анализ предметной области**

## **1.1 Постановка задачи**

Целью текущей бакалаврской работы является разработка приложения, осуществляющего управление комфортностью жилого помещения. Разработанная система должна обеспечивать автоматический процесс решения задач таких систем как: безопасности, управления освещением и управлением климатом.

Система должна удовлетворять следующим требованиям:

- состав системы должен включать в себя головное устройство – сервер, обрабатывающий данные с датчиков. датчики, состоящие из самого датчика и системы передачи данных через выбранный интерфейс, исполнительное устройство, также включающее в себя исполнительное устройство и систему передачи данных;
- при передаче данных на удаленный сервер должна сохраняться их целостность;
- должно обеспечиваться устойчивое соединение клиентов (датчиков и исполнительных устройств) и удаленного сервера;
- система должна иметь более низкие показатели энергозатрат по сравнению с аналогичными системами;
- стоимость системы должна быть ниже, чем стоимость систем аналогов;
- точность данных должна варьироваться в пределах допустимых значений погрешности;
- вывод данных и управление исполнительными устройствами должно осуществляться через простой в использовании интерфейс.

Система должна выполнять следующие задачи:

- управление источниками света;

- коррекция работы отопительной системы;
- уведомление об угрозе жизни (вторжение, возгорание, протечка воды).

Целями создания данной системы являются:

- создание оптимальной системы управления комфортностью среды поддерживающей аппаратную составляющую;
- обеспечение удобной автоматизации выполнения задач основных систем.

Для решения поставленной цели необходимо решение следующих задач:

- провести анализ существующих систем управления комфортностью среды;
- спроектировать систему опираясь на установленные функциональные требования;
- реализовать систему благодаря выбранным средствам реализации удовлетворяющую функциональным требованиям.

## **1.2 Функциональные возможности**

- возможность включения/выключения подключенных к системе осветительных приборов, как по состоянию внешних условий, так и по активации пользователя;
- регулирование подачи тепла в зависимости от предпочтения пользователя или от внешних условий;
- возможность уведомления пользователя об изменения состояния среды, исполнительных устройств или получения определенного набора данных от датчиков.

## **1.3 Анализ существующих решений**

### **1.3.1 Xiaomi**

#### **Архитектура системы:**

В системе Xiaomi [2] используются устройства, взаимодействующие между собой по протоколу Zigbee с подключением к шлюзу. Шлюзов в системе может быть несколько. Помимо Zigbee есть и самостоятельные устройства, взаимодействующие по средствам Wi-Fi без возможности локального управления.

#### **Экосистема:**

Xiaomi производит огромное количество элементов умного дома под своим собственным брендом. Другие компании также выпускают продукцию, подходящую к подключению к умному дому.

#### **Кроссплатформенность, управление голосом:**

Mi Home поддерживает голосовые ассистенты Google, Amazon, Yandex. Однако управление ограничено и обеспечивает полноценную поддержку только через официальное приложение Mi Home.

Mi Home легко интегрируется в систему Apple Homekit, однако может использоваться один определенный шлюз с ограниченной поддержкой некоторых устройств.

#### **Подключение к интернету, сценарии:**

Основной отличительной особенностью Mi Home является взаимодействие не только элементов внутри сети, но и взаимодействие сценариев.

Управление удаленно реализовано лучше, чем в остальных системах, но даже находясь в одной сети нельзя управлять устройствами без интернета.

### **1.3.2 IKEA**

#### **Архитектура системы:**

В системе IKEA Home Smart [3] центром системы является шлюз, связывающий устройства по протоколу Zigbee, он подключается к роутеру соединяя устройства с домашней сетью. У этого подхода есть преимущества, он позволяет не нагружать канал Wi-Fi лишними устройствами, устройства, использующие протокол Zigbee, потребляют меньше энергии. Недостатком является зависимость системы и всех устройств от головного устройства.

#### **Экосистема:**

Экосистема IKEA не включает в себя огромного количества датчиков. Все они производятся самой компанией. А фирменный шлюз не поддерживает другие устройства, поддерживающие интерфейс Zigbee.

#### **Кроссплатформенность, управление голосом:**

Говоря о кроссплатформенности, линейка Tradfri куда более «дружелюбна». Шлюз полноценно поддерживает основные системы голосового управления: Amazon Alexa, Google Home, Apple Homekit. Точно так же, без помощи сторонних разработчиков, система от IKEA отлично внедряется в Samsung Smartthings и другие Zigbee-системы.

#### **Подключение к интернету, сценарии:**

В данный момент система Home Smart не может контролироваться извне без подключения внешних сервисов. Все возможности фирменного приложения ограничиваются управлением внутри локальной сети, что позволило разработчикам сделать полностью автономную систему.

Обеспечена поддержка сценариев – автоматических и пользовательских.

### **1.3.3 Apple**

#### **Архитектура системы:**

Apple Homekit [4]. Устройства входящие в экосистему Apple являются в большинстве случаев обособленными устройствами которые с легкостью можно использовать и без экосистемы. При этом устройства делятся на те, которым не нужно головное устройство. Такие устройства, как кнопки и различного рода датчики лучше использовать с головным устройством. Связь между устройствами может осуществляться как по средствам Bluetooth или Wi-Fi.

#### **Экосистема:**

Устройства Homekit не производятся самой компанией. Однако проходят строгую сертификацию, что обеспечивает безопасность и устойчивость к внешним воздействиям. Сертификация также увеличивает цену устройства по сравнению с конкурентами. Количество устройств с момента запуска платформы составляет более 500 устройств от разных производителей.

#### **Кроссплатформенность, управление голосом:**

Кроссплатформенность является одним из недостатков системы Homekit, так как управление устройствами может осуществляться с другим устройством экосистемы Apple. То же самое касается и управления голосом.

#### **Подключение к интернету, сценарии:**

Взаимодействие между устройствами осуществляется по Wi-Fi и Bluetooth внутри одной сети.

### **1.3.4 Samsung**

#### **Архитектура системы:**

Система Samsung [5] базируется на собственном шлюзе, с которого начинается вся система, однако и другие устройства могут выполнять эту роль. В качестве шлюза соответственно может использоваться несколько устройств. Связь также осуществляется по средствам Wi-Fi и Bluetooth.

### **Экосистема:**

Все свои устройства SmartThings производятся самой компанией Samsung. К шлюзу могут подключаться и устройства от сторонних производителей. Это позволяет расширить аудиторию и спектр используемых устройств. Так на данный момент экосистема насчитывает более 5000 устройств.

### **Кроссплатформенность, управление голосом:**

Кроссплатформенность является одним из очередных преимуществ так как приложения SmartThings выпущены под такие операционные системы как Android, IOS, Tizen, watchOS.

Со стороны голосового управления обеспечена поддержка таких популярных голосовых ассистентов как Google Assistant, Amazon Alexa, Samsung Bixby, Yandex.Alisa.

### **Подключение к интернету, сценарии:**

Предусмотрена возможность использования только локальных сценариев без интернета. Однако если устройства подключены к шлюзу через протоколы Zigbee/Z-Wave сценарий их работы будет работать. Но если сценарий включает в себя устройство подключающееся через Wi-Fi, то оно уже не запустится.

## **1.4 Определение параметров комфорtnости**

Для определения параметров комфорtnости среды необходимо обратится к ГОСТ определяющих параметры микроклимата в жилых помещениях. А именно к ГОСТ 30494-2011 [6] «Здания жилые и общественные. Параметры микроклимата в помещениях» который определяет какие параметры определяют комфорtnость жилого помещения и в каких пределах они должны находятся.

Так такими параметрами комфорtnости являются являются:

- температура воздуха;
- скорость движения воздуха;
- относительная влажность воздуха;
- результирующая температура помещения;

- локальная асимметрия результирующей температуры.

Данные параметры повторяют параметры систем реализуемых в системах Умного дома, что еще раз подчеркивает важность реализации управления данными параметрами.

## **1.5 Вывод по главе**

В данной главе была поставлена задача по разработке веб-приложения, позволяющего решать задачи основных систем автоматизации и «умного дома», направленных на комфорт пользователей. Такими системами являются: системы управления источниками света, коррекции работы отопительной системы, уведомлений об изменении состояния среды. Обзор аналогов позволил получить представление о том, какой должна быть система и какой функционал она должна реализовывать. ГОСТ устанавливающий параметры микроклимата соответствующих комфорtnому состоянию среды жилого помещения еще раз подчеркнул важность реализации решения задач вышеописанных систем.

## 2 Проектирование системы

### 2.1 Определение функциональных требований

После анализа приложений и систем, реализующих аналогичные задачи, а также при более детальном рассмотрении технического задания были сформированы следующие требования к функционалу:



Рисунок 1 – Наглядное отображение функционала доступного пользователю

В таком случае функционал должен разделится на следующие модули:

- a) работа с пользователями:
  - 1) авторизация пользователя;
  - 2) регистрация пользователя.
- b) работа с устройствами:
  - 1) добавление устройства;
  - 2) получение данных от устройств;
  - 3) обработка данных от устройств;
  - 4) сохранение данных от устройств.
- c) работа со сценариями:
  - 1) создание сценариев;
  - 2) редактирование сценариев;
  - 3) удаление сценариев;
  - 4) выполнение сценариев.

- г) работа с логом событий:
  - 1) создание лога;
  - 2) отображение лога.

### **2.1.1 Работа с пользователями**

Для использования приложения необходима аутентификация пользователя в системе. Для этого должны быть предусмотрены системы авторизации и регистрации пользователей:

- а) система авторизации должна использовать следующие данные, с помощью которых должен осуществляться вход в систему:
  - 1) адрес электронной почты;
  - 2) пароль;
- б) система регистрации должна использовать следующие данные, с помощью которых должна осуществляться регистрация в системе:
  - 1) адрес электронной почты;
  - 2) имя и фамилия пользователя;
  - 3) логин;
  - 4) пароль.

### **2.1.2 Работа с устройствами**

При разработке приложения необходимо связать устройства и приложение на сервере. Между ними должен осуществляться обмен сообщения для их последующей обработки как на устройстве, так и в приложении на сервере. Обработка включает в себя следующие функции:

- а) добавление устройства для последующего обмена данными между устройством и приложением на сервере, должно осуществляться по средствам отправки устройством приветственного сообщения в формате json которое может содержать следующую информацию:

1) тип устройства;

2) способ передачи данных в обратную сторону.

к этим данным могут быть добавлены дополнительные данные, которые изменяет сам пользователь:

3) имя;

4) описание;

5) комната, в которой находится устройство.

б) при подключении устройства к приложению должно осуществляться добавление информации об устройстве в соответствующую таблицу базы данных;

в) при регистрации нового пользователя необходимо осуществлять получение информации от устройств, которые принадлежат авторизированному пользователю благодаря генерируемому токену. данный токен будет использоваться для подключения пользовательских устройств и дальнейшего получения от них данных;

г) при получении данных от устройств пользователей, данные должны добавляться в лог, представляющий собой таблицу базы данных для последующего вывода для пользователя в читаемом и наглядном формате. лог может содержать следующую информацию:

1) идентификатор устройства, от которого получены данные;

2) данные от подключенных устройств;

3) временная метка (когда были получены данные).

д) серверное приложение должно осуществлять обработку входящих данных от устройств, подключенных к приложению. Данные должны обрабатываться в соответствии с устройством, и данными, которые были получены. так приложение должно обрабатывать следующие события:

1) подключение к системе нового устройства для последующего получения данных с датчиков или для управления исполнительным устройством;

- 2) при получении данных от устройств в формате json необходимо обработать полученные данные, добавить их в соответствующую таблицу базы данных и проверить наличие сценариев, относящихся к устройству, от которого пришло сообщение.
- е) пользователь должен получать информацию о подключенных устройствах к системе;
- ж) пользователь должен иметь возможность получить более подробную информацию об устройстве, изменить доступную информацию и удалить устройство из системы при необходимости.

### **2.1.3 Работа со сценариями**

Одной из основных функций приложения является создание сценариев, которые должны связывать датчики и исполнительные устройства пользователей между собой.

- а) пользователь должен иметь возможность создавать, изменять, удалять и просматривать сценарии взаимодействия устройств;
- б) сценарии подразумевают их выполнение при определенных условиях, проверка которых осуществляется после получения данных от устройств;
- в) сценарии могут делиться на различные категории, которые позволяют их структурировать, по-разному обрабатывать и выполнять;
- г) сценарии, которые создают пользователи могут содержать в себе следующие данные, в зависимости от которых сценарии будут выполняться по разным алгоритмам:

- 1) текстовое описание сценария, для более удобного поиска и отображения;
- 2) время и дата, при каких значениях должен выполняться скрипт;
- 3) условие, при котором сценарий начинает свою работу (например, данные, полученные от сенсора);

- 4) значение, которое должно сравниваться со значением, полученным от выбранного устройства;
- 5) исполнительное устройство, которое должно использоваться сценарием;
- 6) значение, которое должно отправляться на исполнительное устройство для его включения или отключения;
- 7) тип скрипта, который повторяет условные категории, на которые должны делиться сценарии, для сортировки в таблице базы данных и соответствующей обработки приложением;

#### **2.1.4 Работа с логом событий**

Приложение должно включать в себя лог устройств – сохранение и отображение данных от устройств, подключенных к приложению. Информация должна выводиться пользователю в наглядном виде, сортированном по устройствам.

- приложение должно получать данные пришедшие от устройств и записывать в соответствующую таблицу базы данных для дальнейшего отображения пользователю;
- пользователю должен предоставляться выбор устройства, для которого необходимо более подробное отображение информации;
- отображение подробной информации, полученной сервером от устройства, которая записана в соответствующую таблицу базы данных.

#### **2.1.5 Диаграмма прецедентов**

Учитывая весь вышеописанный функционал, можно построить следующую диаграмму прецедентов, отображенную на рисунке 2, которая отображает какие действия сможет выполнить пользователь при использовании данного приложения.

Действия пользователя, в которых можно получить конечный результат делятся на следующие группы:

- a) управление учетными записями:
  - 1) регистрация пользователя;
  - 2) авторизация пользователя.
- б) управление сценариями:
  - 1) создание сценария взаимодействия между устройствами;
  - 2) редактирование сценария взаимодействия между устройствами;
  - 3) удаление сценария взаимодействия между устройствами.
- в) управление устройствами:
  - 1) редактирование доступной информации об устройствах;
  - 2) удаление устройств из системы.
- г) управление логом:
  - 1) получение информации о данных устройствах.

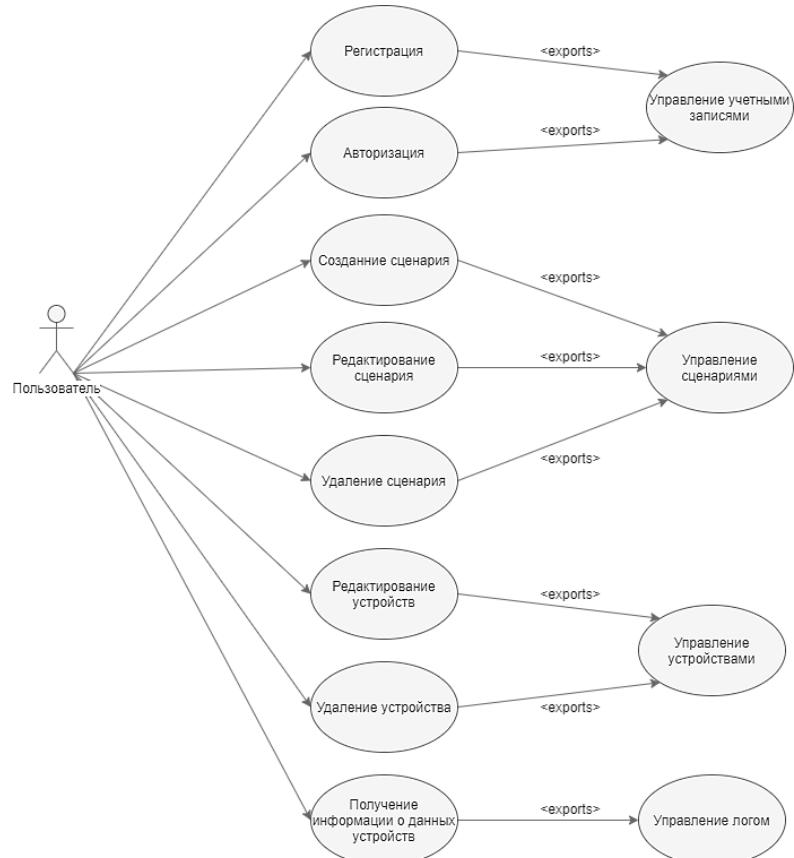


Рисунок 2 – Диаграмма прецедентов

## **2.2 Определение структуры приложения**

Структурная схема, отображенная на рисунке 3, включает в себя следующие элементы:

- а) персональное устройство – устройство, на котором будет запускаться клиент, представляющий пользовательский интерфейс, с которым будет взаимодействовать пользователь;
- б) сервер – клиент на персональном устройстве связан с сервером через http запросы. сервер позволяет выполнять заявленные действия в интерфейсе пользователя. на сервере происходит:
  - 1) формирование интерфейса пользователя, адаптированного под персональное устройство пользователя;
  - 2) отправка данных на страницы для их отображения и использования пользователем;
  - 3) обработка запросов на получение данных, обработка и последующая отправка пользователю;
  - 4) хранение данных, приходящих от устройств на сервер, в базе данных, которая расположена на сервере для удобного доступа и обработки.
- в) микроконтроллер – устройство, которое может быть подключено к приложению посредством токена, который генерируется при регистрации пользователя. для связи с сервером подразумевается, что микроконтроллер имеет возможность подключения к сети интернет. к микроконтроллеру непосредственно подключаются датчики, от которых сервер получает входные данные и исполнительные устройства, которыми необходимо управлять.

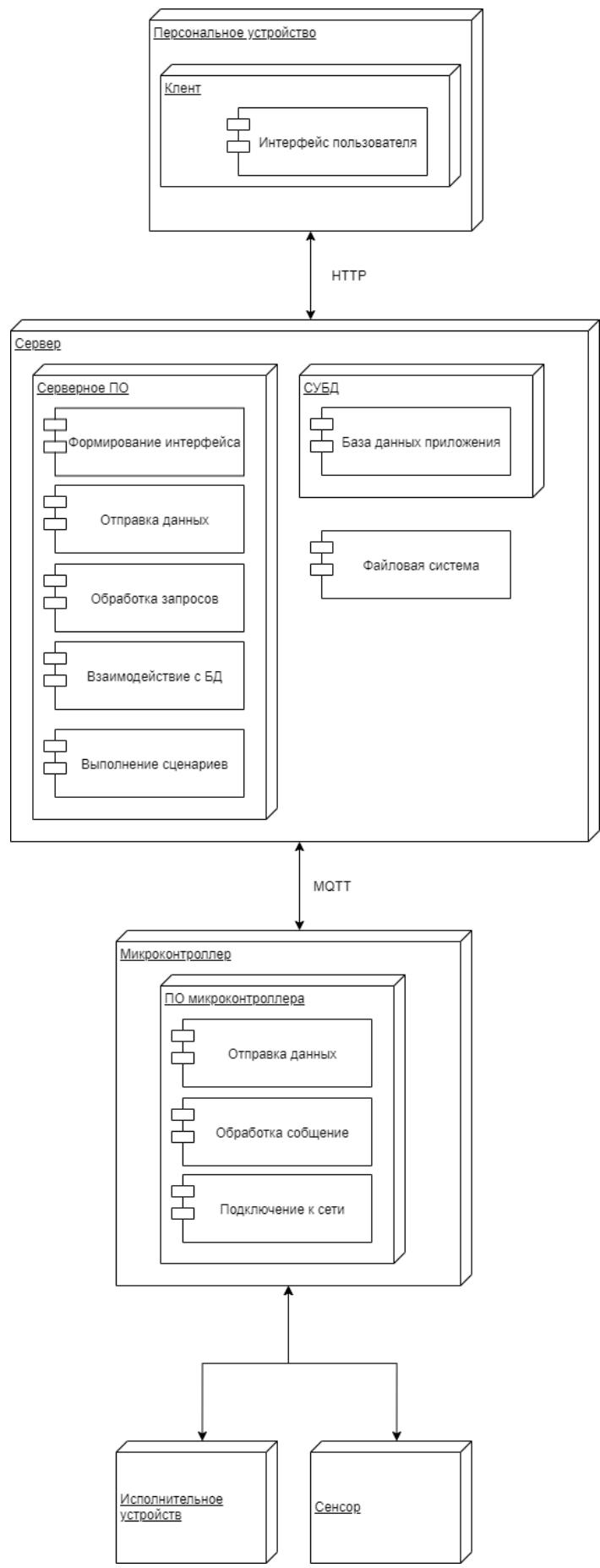


Рисунок 3 – Структурная схема

Для разработки серверного программного обеспечения необходимо разбить веб-приложение на логические части. Эта задача может решаться с помощью шаблонов архитектуры системы. Наиболее распространенными шаблонами являются:

- Model-View-Controller;
- Model-View-Presenter;
- Model-View-ViewModel.

Так как это веб-приложение и обновление части клиента подразумевается чаще, чем серверной части, исходя из этого и был выбран шаблон проектирования MVC представленный на рисунке 4.

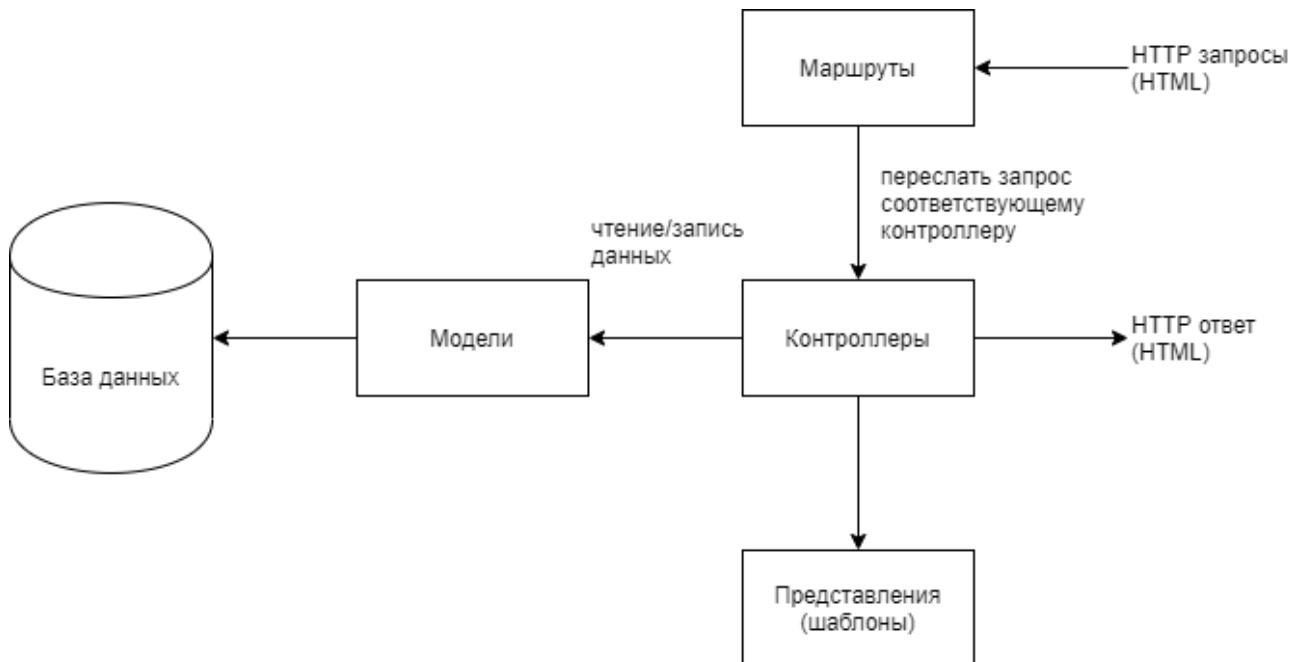


Рисунок 4 – Структура MVC

### 2.2.1 Структура контроллеров

Контроллер – это основной элемент системы серверной части. Данный элемент реализовывает функции, которые может использовать пользователь и делится на функциональные требования, необходимые к реализации в веб

приложении. Так в приложении должны быть реализованы следующие контроллеры:

- AuthController – контроллер, осуществляющий обработку логики для аутентификации новых пользователей в веб-приложении;
- ScriptController – контроллер, осуществляющий работу со скриптами, их создание, редактирование и удаление;
- LogController – контроллер, осуществляющий работу отображения данных, полученных от устройств авторизированного пользователя;
- DeviceController – контроллер, осуществляющий работу отображения данных об устройствах, которые были добавлены в систему пользователями;
- HomeController – контроллер, осуществляющий работу отображения технических представлений (например, инструкции по использованию веб-приложением).

### **2.2.2 Структура представлений**

Представления являются клиентской частью веб-приложения. Пользователь взаимодействует только с ними, добавляя или изменяя базу данных через данные, передаваемые через представления контроллеру.

Рассматривая требуемый функционал можно сделать вывод, что необходимы следующие представления представленные на рисунке 5:

a) представления аутентификации:

1) AuthIndexView – главное представление при входе в веб-приложение предоставляющее выбор между авторизацией пользователя или регистрацией нового;

2) AuthSignUpView – представление для регистрации пользователей, принимающее данные о пользователе;

3) AuthSignInView – представление для авторизации пользователей. При успешной регистрации представление принимает данные для авторизации.

б) представление сценариев пользователя:

1) ScriptListView – представление, отображающее все сценарии, который пользователь добавил в систему. Через данное представление должен предоставляться выбор для удаления или изменения сценария, что вызовет переход на соответствующие представления;

2) ScriptAddView – представление для добавление сценария в систему, представляющее из себя форму, которая предлагает на выбор параметры для создания сценария пользователя;

3) ScriptEditView – представление для изменения сценария в системе, представляющее из себя форму, которая предлагает на выбор параметры для редактирования пользователем.

в) представление лога событий:

1) LogListView – представление, предоставляющее на выбор устройство, для которого должна быть выведена более подробная информация – данные, которые были получены от выбранного устройства;

2) LogDataView – представление, отображающее информацию, которая была получена от выбранного устройства.

г) представление устройств:

1) DeviceListView – представление, отображающее все устройства, которые пользователь добавил в системе. Через данное представление должен предоставляться выбор для удаления или изменения информации об устройстве, что вызовет переход на соответствующие представления;

2) DeviceEditView - представление для изменения доступной информации устройства пользователя в системе, представляющее из себя форму, которая предлагает на выбор параметры для редактирования пользователем.

д) представления технические:

- 1) HomeProfileView – представление, которые отображает информацию о пользователе, авторизированного в системе, а также информацию об устройствах и сценариях, которые были добавлены пользователем;
- 2) HomeMainView – представление, отображаемое при успешной авторизации пользователя в системе, предлагающее пользователю ознакомиться с инструкцией по использованию веб-приложением.

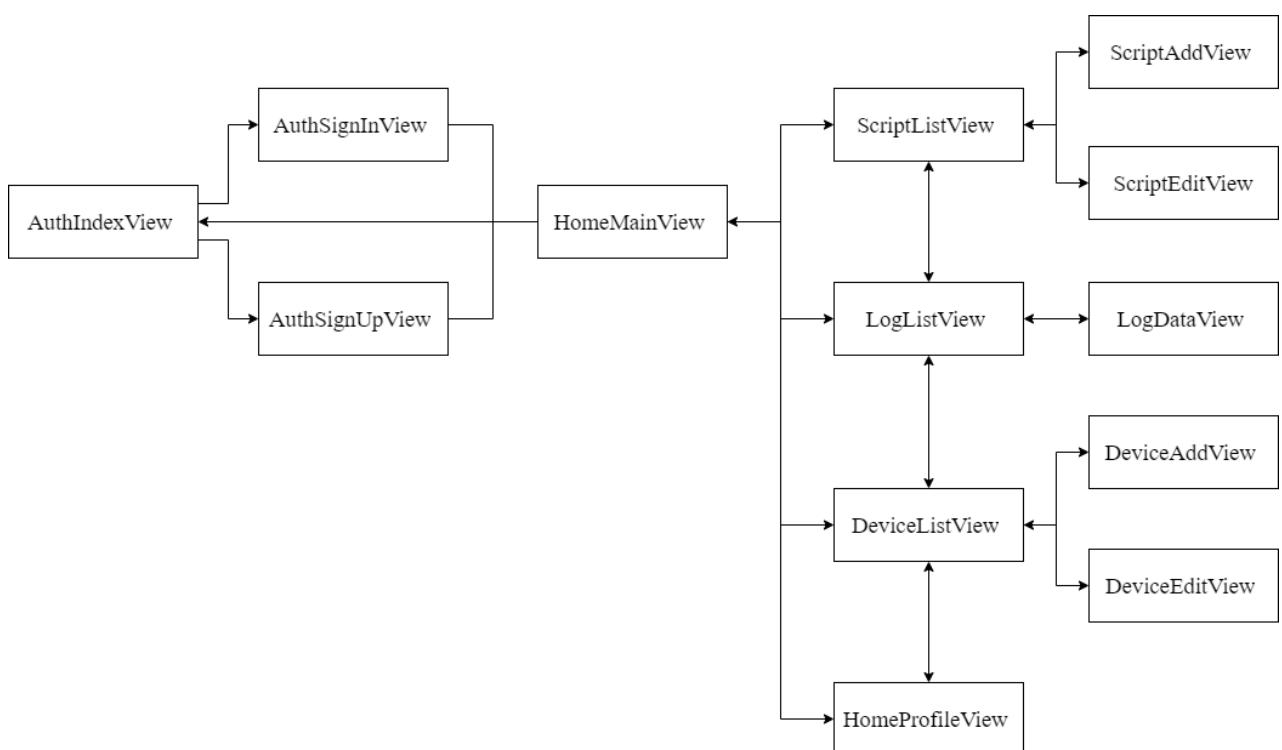


Рисунок 5 – Иерархия представлений

### 2.2.3 Структура моделей

Шаблон проектирования MVC [7] включает в себя (Model-View-Controller) использование моделей. Модель, в данном контексте, представляет собой данные и методы работы с ними, при этом не зависит от представления (View) и контроллера (Controller). Структура моделей будет повторять структуру модели представления данных. Таким образом шаблон проектирования будет

использовать следующие модели, которые будут содержать нижеперечисленные поля:

а) DeviceModel – модель представления данных об устройствах подключенных к системе:

- 1) уникальный идентификатор устройства;
- 2) имя устройства;
- 3) описание устройства;
- 4) комната, в которой находится устройство;
- 5) топик для связи с устройством;
- 6) тип устройства;
- 7) уникальный идентификатор пользователя, которому принадлежит устройством.

б) LogsModel – модель представления данных, полученных от устройств, подключенных к веб-приложению:

- 1) уникальный идентификатор элемента лога;
- 2) данные, которые были получены от устройства;
- 3) время получения данных;
- 4) уникальный идентификатор устройства, от которого получены данные;
- 5) временная метка, когда эти данные были получены.

в) ScriptsModel – модель представления сценариев взаимодействия устройств между собой:

- 1) уникальный идентификатор сценария;
- 2) текстовое описание сценария;
- 3) тип сценария;
- 4) время и дата, в которые должен выполняться сценарий;
- 5) условие, при котором сценарий должен начать выполняться;
- 6) параметры исполнительного устройства, которые должны использоваться;

- 7) уникальный идентификатор пользователя, который создал сценарий.
- г) UsersModel – модель представления данных о пользователе, который авторизирован в системе:
- 1) уникальный идентификатор пользователя;
  - 2) имя пользователя;
  - 3) фамилия пользователя;
  - 4) логин пользователя;
  - 5) адрес электронной почты пользователя;
  - 6) личный токен для связи устройств пользователя в веб-приложении.

## 2.3 Выбор средства реализации

### 2.3.1 Выбор программной платформы для реализации web-клиента

При выборе программной платформы необходимо учитывать решения, подходящие для разработки веб-приложений, с помощью которых можно будет реализовать заявленный функционал.

Так рассматривая все возможные решения для данной задачи, подходят следующие платформы представленные в таблице 4:

Таблица 4 – Серверные платформы

Наименование платформы	Лицензия	Веб-сервер
ASP.NET	свободная	специализированный
C/C++	свободная	практически любой
Java	свободная	практически любой
Perl	свободная	практически любой
PHP	свободная	практически любой
Python	свободная	практически любой
Ruby	свободная	практически любой
Node.js	MIT License	собственный
ASP.NET	Apache 2	практически любой

Для разрабатываемого приложения нежелательно наличие проприетарной лицензии, что является причиной использования специализированных веб-серверов для их использования, которые накладывают определенные ограничения на веб-приложение и его использование. Для текущего проекта необходимо установить связь между сервером и устройствами, поэтому наиболее подходящей платформой для реализации веб-приложения является Node.js [8]

### **2.3.2 Выбор аппаратной платформы для реализации**

Одной из задач выпускной квалификационной работы является реализация программно-аппаратного комплекса, поэтому для отладки работы веб-приложения и управления устройствами нам необходима микроконтроллерная система, которая будет выступать в качестве аппаратной платформы. Микроконтроллерная система должна обладать следующей функциональностью:

- а) Возможностью подключения внешних устройств;
- б) Осуществлять связь с сервером и передачу/принятие данных по средствам сети Интернет;
- в) Платформа должна быть адаптирована как устройство iot:
  - 1) использовать соответствующие протоколы;
  - 2) обладать возможностью упаковать соответствующее данные в универсальный формат.

Микроконтроллерные системы, подходящие под требования отображены в таблице 5.

Таблица 5 – Микроконтроллерные системы

Наименование	Базовая тактовая частота процессора	Диапазон напряжения	ROM	RAM	I/O	Цена
ESP32	16 MHz	3,3 В	4Mb	80Kb	11	540руб/250руб*
Arduino Mega 2560	16 MHz	5В	4Mb	80Kb	54	5800р/2100руб*
WeMos D1 mini	16 MHz	3,3 В	8Mb	80Kb	11	290руб/133руб*
* - доставка из других регионов						

Исходя из вышеперечисленных требований и представленных микроконтроллерных систем, можно сделать вывод, что наиболее подходящей является ESP, а именно ESP8266 [9] изображенная на рисунке 6.



Рисунок 6 – Микроконтроллерная система ESP8266

### 2.3.3 Выбор модели представления данных

При выборе модели представления данных подразумевается то, как данные будут храниться в базе данных. Так модели делятся на два типа:

- Реляционные базы данных [10];
- Не реляционные базы данных. [11].

Реляционные базы данных хранят структурированные данные, которые обычно представляют объекты реального мира. Например, это могут быть сведения о человеке или же данные о содержимом корзины для товаров в магазине, сгруппированные в таблицах, формат которых задан на этапе проектирования хранилища.

Не реляционные базы данных устроены иначе. Например, документо-ориентированные базы хранят информацию в виде иерархических структур данных. Речь может идти об объектах с произвольным набором атрибутов. То, что в реляционной БД будет разбито на несколько взаимосвязанных таблиц, в не реляционной может храниться в виде целостной сущности.

Для разработки веб-приложения нам необходима реляционная база данных, которая позволит разбить хранимые данные на таблицы и связать их между собой с помощью первичного ключа.

Рассматривая выбор систем управления базами данных и учитывая выбор платформы, можно сделать вывод что для разработки данного веб-приложения подходят две наиболее распространенные системы:

- MySQL [12];
- MongoDB [13].

MySQL это свободная и реляционная система управления базами данных, постоянно обновляемая и официально поддерживаемая производителем. Основными преимуществами является наличие документации, поддерживает большое количество пользовательских интерфейсов. Из недостатков – необходима настройка объектно-реляционного отображения системы обеспечивающая более простое взаимодействие с базой данных на уровне кода.

Mongo DB – это документно-ориентированная система управления базами данных с открытым исходным кодом, не требующая описания схемы таблиц, использующая json-подобные документы и схему базы данных. Из преимуществ можно выделить скорость и простоту использования. В качестве недостатки можно выделить: SQL не используется в качестве языка запросов.

Подытоживая выбор систем управления базами данных и учитывая все преимущества и недостатки, которые были рассмотрены, можно сделать вывод, что наиболее подходящей является MySQL, несмотря на дополнительные на манипуляции для работы с ней.

### **2.3.4 Выбор протокола**

Для выбора протокола для связи устройств, добавляемых пользователями при использовании веб-приложения, необходимо рассмотреть существующие протоколы, которые распространены в устройствах интернета вещей.

В рамках интернета вещей протоколы делятся на следующие участки:

- сенсорный узел – сенсорный узел (самый распространенный протокол dds);
- сенсорный узел – сервер (coap, mqtt, xmpp, stomp);
- сервер – сервер (amqp).

DDS (Data Distribution Service) [14] – реализует шаблон публикации-подписки для отправки и приема данных, событий и команд среди конечных узлов. Узлы-издатели создают информацию, «topic» (темы, разделы: температура, местоположение, давление) и публикуют шаблоны. Узлам, заинтересовавшимся в данных разделах, DDS прозрачно доставляет созданные шаблоны. В качестве транспорта – UDP. Также DDS позволяет управлять параметрами QoS (качество обслуживания).

MQTT (Message Queue Telemetry Transport) [15] – осуществляет сбор данных от множества узлов и передачу на сервер. Этот протокол основывается на модели издаатель-подписчик с использованием промежуточного сервера – брокера (приоритизация сообщений, формирование очередей и др.). В качестве транспорта – TCP. На основе MQTT был сформирован специализированный протокол MQTT-SN для сенсорных сетей.

Деление протоколов на участки работы позволяет исключить протоколы, относящиеся к работе по связи «сервер-сервер», так как в разрабатываемом веб-приложении подразумевается использование только одного сервера. А задача построение программно-аппаратного комплекса управления комфортностью среды подразумевает использование большого количества датчиков, от которых

необходимо получать данные. Поэтому MQTT удовлетворяет всем требованиям для протокола.

### **2.3.5 Выбор средств разработки**

В результате анализа средств разработки компонентов веб-приложения были выбраны следующие программные продукты:

- редактор кода Microsoft Visual Studio Code [16] – для взаимодействия с программной платформой с целью реализации веб-приложения Node.js;
- интегрированная среда разработки Arduino IDE [17] – для взаимодействия с аппаратной платформой;
- система управления базами данных MySQL [18] – для работы с моделью представления данных;
- программно обеспечение, реализующее MQTT клиент MQTTBox [19] – для работы с протоколом взаимодействия устройств, добавленных пользователем и веб-приложением.

## **2.4 Вывод по главе**

В ходе проектирования системы были определены функциональные требования, благодаря которым было получено представление, какой функциональностью должно обладать разрабатываемое веб-приложение, определена структура приложения, а также выбраны средства реализации для каждого элемента системы.

### **3 Реализация приложения**

#### **3.1 Разработка серверного программного обеспечения**

##### **3.1.1 Разработка системы аутентификации**

Реализация систем аутентификации состоит из двух основных функций – регистрация и авторизация пользователей. Для каждой из этих функций необходимо получение данных от пользователя, сохранение данных пользователей, сравнение данных с данными, хранящимися в таблице пользователей базы данных и последующее сравнение с данными, введенными пользователями. Это можно реализовать стандартными методами. Но есть и существующие решения позволяющие настроить систему авторизации. Одним из таких решений является Passport.js [20].

Passport.js является библиотекой для построения системы аутентификации на Node.js. Отличительной особенностью является использование стратегий – от обычного логина и пароля, до стратегий, поддерживающих авторизацию через социальные сети такие как Google, Вконтакте и другие. Защита маршрутов, которые могут использовать пользователи защищены через стратегию, а при успешной авторизации данные сохраняются с помощью сессий.

Для использования Passport.js должен находиться конфигурационный файл, реализующий следующие стратегии и функции:

- LocalSignUp – локальная стратегия регистрации нового пользователя в системе, получающая необходимые данные и проверяющая их наличие в таблице пользователей базы данных;
- LocalSignIn - локальная стратегия авторизации нового пользователя в системе, получающая необходимые данные и проверяющая их наличие в таблице пользователей базы данных;
- SerializeUser – для создания сессии для нового пользователя;

- DeserializeUser – для удаления сессии при выходе пользователя.

Также необходимо установить промежуточную проверку маршрутов веб-приложения, которая необходима для проверки на авторизованность пользователя для перехода по данному маршруту.

### **3.1.2 Разработка сервиса передачи данных между сервером и устройствами**

Для подключения устройств к серверу был выбран протокол MQTT – упрощенный сетевой протокол, работающий поверх TCP/IP, ориентированный для обмена сообщениями между устройствами по принципу издатель-подписчик. Протокол ориентируется на простоту использования, невысокую нагрузку на каналы связи, работу в условиях постоянной потери связи, легкую встраиваемость в любую систему. Основная задача данного протокола – работа с телеметрией от различных датчиков и устройств. Использование шаблона подписчика обеспечивает возможность устройствам выходить на связь и публиковать сообщения, которые не были заранее известны или предопределены, в частности, протокол не вводит ограничений на формат передаваемых данных.

Для этого реализовывается файл сервис MQTT который обеспечивает следующие методы, которые необходимы при работе других модулей:

- subscribeTopic – метод,зывающий подписку на передаваемый топик. Данный метод используется при регистрации нового пользователя и своевременной подписки на личный топик пользователя, после чего может быть добавлено новое устройство;
- unsubscribeTopic – метод,зывающий отписку от передаваемого топика. Данный метод используется при удалении пользователем устройства и своевременной отписки от топика устройства, после чего приложение больше не сможет получать от него данные;

- publishToTopic – метод, публикующий переданное сообщение в выбранный топик. Данный метод используется при выполнении любого типа сценария, в котором задействовано исполнительное устройства для отправки команды.

Также реализуются стандартные методы, такие как:

а) Connect – стандартный метод, вызывающийся при подключении приложения к MQTT брокеру. Данный метод вызывает методы модуля onConnectStartup, такие как User для подписки на все личные топики пользователей, и Device для подписки на все устройства всех пользователей, которые были добавлены в приложение;

б) Message – стандартный метод, прослушивающий сообщения, которые были получены в топиках, на которые было подписано приложение. Данный метод вызывает методы модуля onMessageHandle, которые позволяют обработать сообщения в зависимости от топика, на который они пришли. Так реализуются следующие методы:

1) Connect – обработка сообщений, пришедших на личный топик пользователя. Сообщение содержит в себе информацию об устройстве, которое добавляет пользователь, после чего оно добавляется в базу данных и вызывается метод subscribeTopic для дальнейшего получения информации от устройства или управления;

2) Event/Command – обработка сообщений, полученных от датчиков и исполнительных устройств соответственно. Данный метод вызывает функцию создания лога, добавляя полученное сообщение в соответствующую таблицу, а также функцию проверки наличия сценариев, связанных с устройством, от которого получено сообщение.

### **3.1.3 Обработка сценариев**

Для обработки и выполнения сценариев разработан алгоритм представленный на рисунке 7, а также реализован модуль HandleScript, методы которого вызываются из модуля onMessageHandle функции checkScripts.

Так реализовываются следующие методы, соответствующие типам поддерживаемых типов сценариев:

- Guard – метод,зывающий функцию verifyCondition, проверяющий выполнение условия сценария. При выполнении условия, информация о сценарии и полученных данных отправляется на электронную почту пользователя, который добавил данный сценарий;
- Classic – метод,зывающий функцию verifyCondition, проверяющий выполнение условия сценария. При выполнении условия, на исполнительное устройство с помощью метода publishToTopic в топик исполнительного устройства, выбранного в сценарии, отправляется выбранная команда.

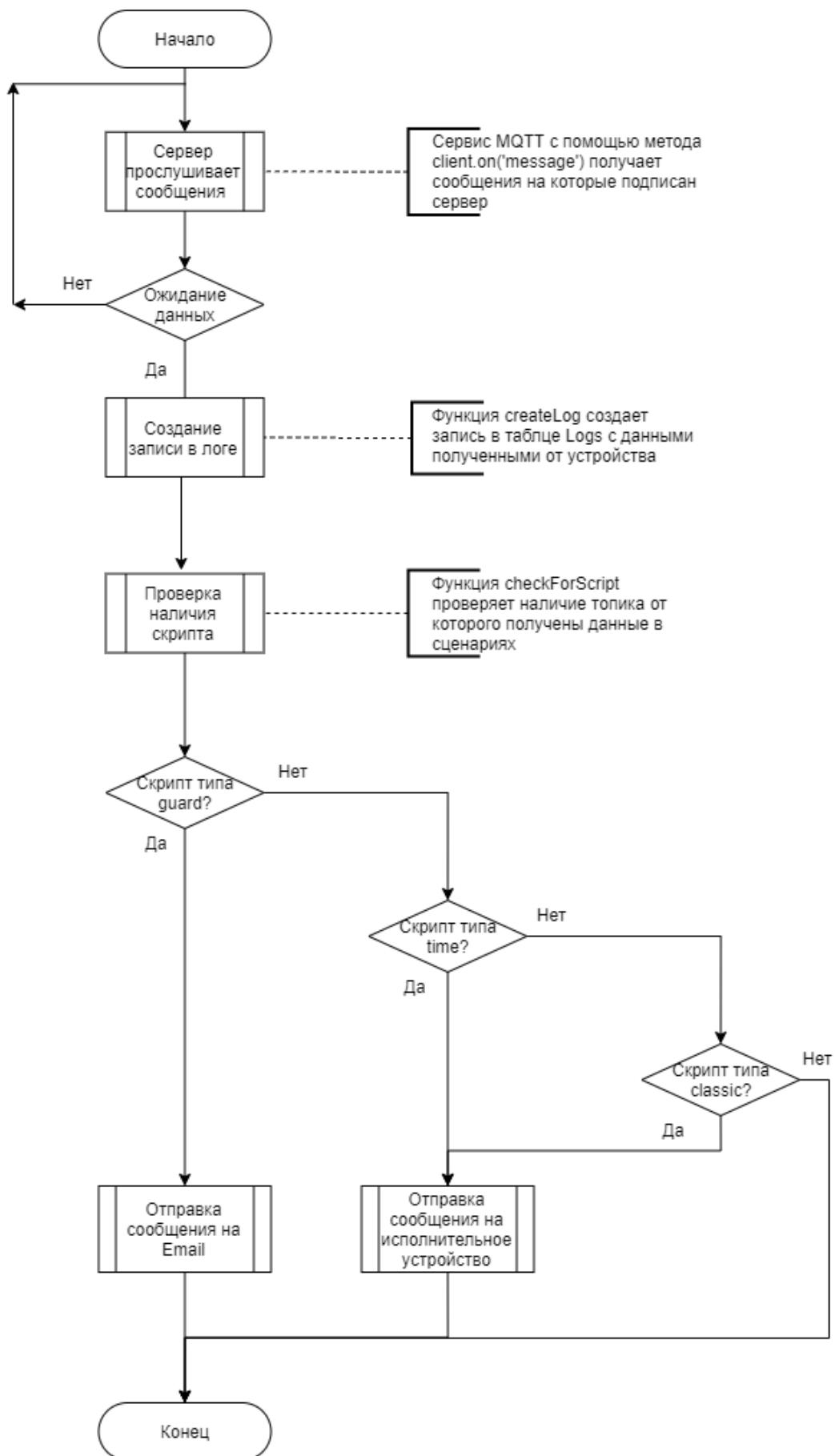


Рисунок 7 - Алгоритм обработки данных

Для выполнения сценариев, использующих временные метки, реализуется модуль Cron. Методы данного модуля вызываются из методов контроллера scriptController. Cron реализует следующие методы:

- createCronPost – метод, вызываемый из метода контроллера createScriptPost. Если создаваемый сценарий имеет временную метку, то сценарий передается в данный метод и создается задача Cron, которая при достижении временной метки отправляет на выбранное исполнительное устройство выбранную команду с помощью метода publishToTopic;
- deleteCronId – метод, вызываемый из метода контроллера deleteScriptPost. Если создаваемый сценарий имеет временную метку, то сценарий передается в данный метод и удаляется задача Cron, которая соответствует выбранному сценарию;
- editCron – метод, вызываемый из метода контроллера editScript. Если изменяемый сценарий имеет временную метку, то сценарий передается в данный метод, тогда изменяется задача, которая соответствует выбранному сценарию.

Для инициализации строки, передаваемой в метод создания задачи Cron, реализуется функция initStringCron, в которую передаётся информацию о временных метках сценария, после чего создается сама задача.

## **3.2 Разработка web-клиента**

### **3.2.1 Разработка контроллеров**

При разработке структуры приложения был сделан вывод, что в системе должны использоваться следующие контроллеры, реализующие следующие функции, которые обеспечивают наличие требуемого функционала приложения:

- a) AuthController – контроллер, осуществляющий обработку маршрутов для аутентификации новых пользователей в веб-приложении:

- 1) Signup – отображение представления AuthSignupView, отображающее форму для регистрации нового пользователя;
  - 2) Signin – отображение представления AuthSigninView, отображающее форму для авторизации нового пользователя;
  - 3) Index – отображение представления AuthIndexView, позволяющее выбрать новому пользователю на выбор регистрацию или авторизацию.
- б) ScriptController – контроллер, осуществляющий работу со скриптами, их создание, редактирование и удаление:
- 1) GetScript – получает все сценарии, которые были добавлены авторизированным пользователем и отображает их в наглядном виде на представлении ScriptListView;
  - 2) CreateScript – функция, получающая данные, введенные пользователем на представлении ScriptAddView, содержащие информацию о добавляемом сценарии для последующего сохранения. При успешном добавлении сценария происходит перенаправление на представление ScriptListView;
  - 3) DeleteScript – удаление сценария происходит после выбора сценария для удаления на представлении ScriptListView. Удаление происходит по его уникальному идентификатору;
  - 4) EditScript – изменение скрипта также осуществляется после выбора сценария на представлении ScriptListView, после чего полученная информация о сценарии по его уникальному идентификатору отображается для изменения на представлении ScriptEditView.
- в) LogController – контроллер для отображения данных полученных, от устройств авторизованного пользователя:
- 1) getLog – отображение устройств, от которых были получены и могут быть отображены более подробные данные. Функция возвращает список этих устройств на представление LogListView;

2) getLogID – функция получает параметры устройства (уникальный идентификатор) для отображения данных, полученных от него. Представление LogDataView получает информацию об устройстве и отображает в наглядном виде эти данные.

г) DeviceController – контроллер для отображения данных об устройствах, которые были добавлены в систему пользователями:

1) GetDevice – получает все устройства, которые были добавлены авторизированным пользователем и отображаются в наглядном виде на представлении DeviceListView;

2) CreateDevice – функция, получающая данные, введенные пользователем на представлении DeviceAddView, содержащие информацию о добавляемом устройстве для последующего сохранения. При успешном добавлении сценария происходит перенаправление на представление DeviceListView;

3) DeleteDevice – удаление сценария происходит после выбора устройства для удаления на представлении DeviceListView. Удаление происходит по его уникальному идентификатору;

4) EditDevice – изменение устройства также осуществляется после выбора устройства на представлении DeviceListView, после чего полученная информация о сценарии по его уникальному идентификатору отображается для изменения на представлении DeviceEditView.

д) HomeController – контроллер для отображения технических представлений (например, инструкции по использованию веб приложением):

1) Profile – функция отображения личной страницы авторизованного пользователя и информации, которая к нему относится (например, информации об аккаунте, количестве добавленных скриптов и устройств).

2) Main – функция отображения главной страницы, которая предполагает отображение различной технической информации (например, инструкции по использованию веб-приложением).

### 3.2.2 Разработка представлений

Как правило веб-приложения использующие визуальный интерфейс используют не стандартные файлы html, а специальные сущности – представления, из которых создаются html-файлы. Преимуществом данного метода является, что можно определять некоторые шаблоны, вместо которых затем вставляется какое-то динамическое содержимое с помощью javascript кода. Управление представлениями, которые используют шаблоны осуществляется через view engine. Наиболее распространенными являются: Pug, Jade, Dust, Nunjucks, EJS, Handlebars. Все они предоставляют схожую функциональность. Поэтому были выбраны шаблоны Handlebars [21].

Каждый из шаблонов, использующихся в веб приложении, был разбит на следующие логические части - макеты, которые используются в каждом основном шаблоне:

Sidenav – меню, через которое осуществляется навигация по веб-приложению.

Каждое из представлений, необходимых в веб-приложении должно содержать определенный компонент, который необходим для корректной работы приложения.

a) представления аутентификации:

1) AuthIndexView – представление на рисунке 8, осуществляющий переход по маршруту для авторизации и регистрации пользователя.

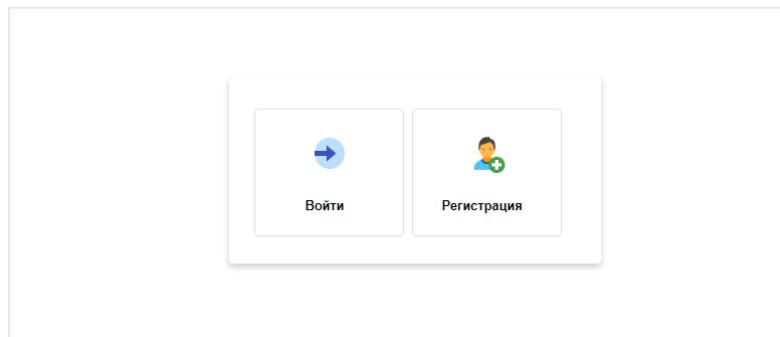


Рисунок 8 – Главная страница системы аутентификации

2) AuthSignUpView – представление на рисунке 9 формы получения данных для регистрации пользователя, после успешной регистрации происходит переход на представление HomeMain.

The screenshot shows a registration form titled 'Адрес электронной почты' (Email address). It contains five input fields: 'Enter Email', 'Фамилия' (Last name), 'Имя' (First name), 'Имя пользователя' (Username), and 'Пароль' (Password). Below these fields is a button labeled 'Зарегистрироваться' (Register). At the bottom of the form, there is a link 'Уже есть аккаунт? Войти' (Already have an account? Log in).

Рисунок 9 – Страница регистрации пользователя

3) AuthSignInView – представление на рисунке 10 формы получения данных для авторизации пользователя.

The screenshot shows a login form titled 'Адрес электронной почты' (Email address). It contains two input fields: one with the value 'oldvertu@gmail.com' and another with the placeholder '.....'. Below these fields is a button labeled 'Войти' (Log in). At the bottom of the form, there is a link 'Еще нет аккаунта? Зарегистрироваться' (No account yet? Register).

Рисунок 10 – Страница авторизации пользователя

6) представление сценариев пользователя:

1) ScriptListView – представление на рисунке 11 для отображения всех сценариев, добавленных пользователем.

The screenshot shows a user interface titled "Сценарии" (Scenarios). At the top right is a button labeled "Добавить сценарии" (Add scenario). Below the title is a section titled "Список сценариев" (List of scenarios). Three scenarios are listed in separate boxes:

- EXAMPLE4**  
guard  
devices/[sensor\_temperature]/event  
equally 76  
Buttons: Изменить (Edit), Удалить (Delete)
- EXAMPLE**  
datetime  
devices/[actuator\_siren]/commands  
1  
Buttons: Изменить (Edit), Удалить (Delete)
- EXAMPLE7**  
guard  
devices/[sensor\_temperature]/event  
more 456  
devices/[actuator\_siren]/commands  
1  
Buttons: Изменить (Edit), Удалить (Delete)

At the bottom of the list are two additional items: "test script 2" and "test comfort script".

Рисунок 11 – Страница всех сценариев

2) ScriptAddView – представление формы получения данных для добавления нового сценария пользователя представленное на рисунке 12.

3) ScriptEditView – представление формы получения данных для изменения уже добавленного сценария пользователя.

Создание сценария

Описание Классический скрипт

Тип Классический

Время дд.мм.гггг -- : --

Датчик Предел

Исполнительное устройство

Команда Включить

Отправить

К списку сценариев

Рисунок 12 – Страница добавления и изменения сценария

в) представление лога событий:

1) LogListView – представление на рисунке 13, отображающий все устройства авторизированного пользователя, от которых были получены данные для последующего просмотра более подробных данных.

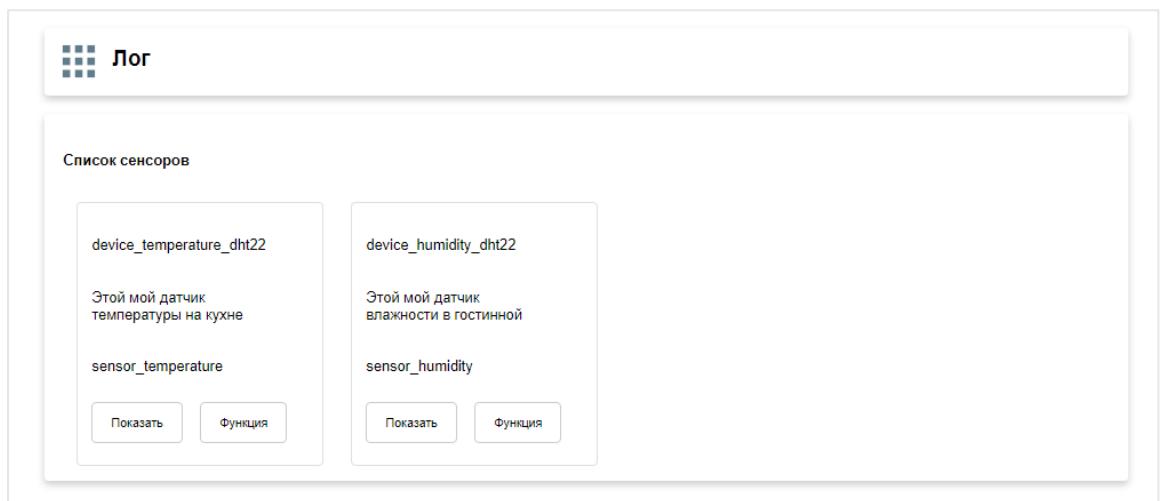


Рисунок 13 – Страница со списком устройств лога

2) Log DataView – представление на рисунке 14 для отображения более подробных данных, полученных от выбранного устройства авторизированного пользователя.



Рисунок 14 – Страница с графиком полученных данных

г) представление устройств:

1) DeviceListView – представление на рисунке 15 для отображения данных всех устройств пользователя.

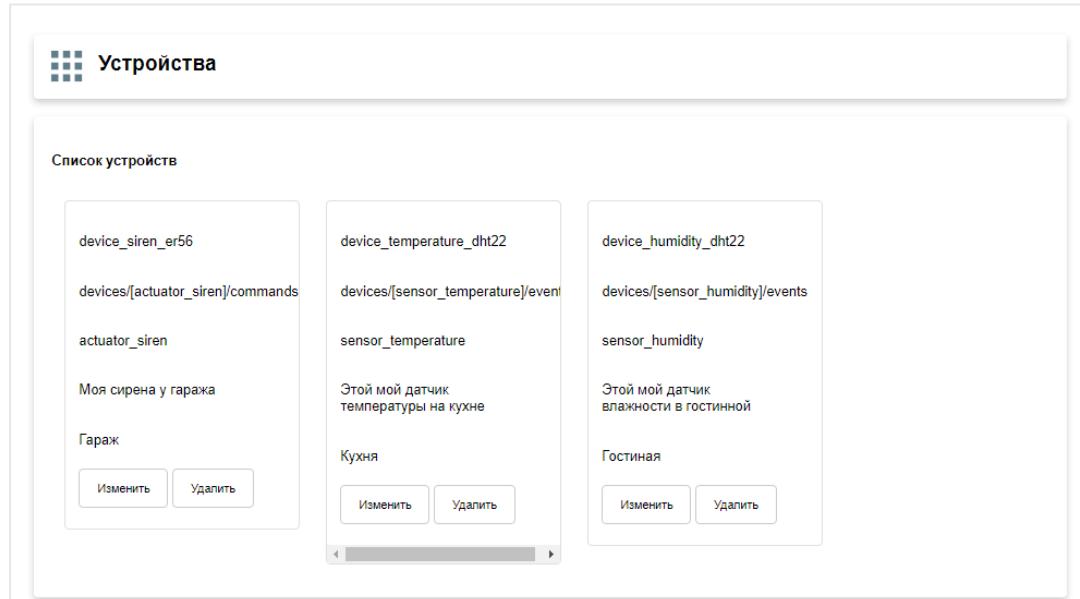


Рисунок 15 – Страница со списком подключенных устройств

2) DeviceEditView – представление формы на рисунке 16 для получения данных для изменения уже добавленного устройства пользователя.

Изменить устройство

Название  
device\_temperature\_dht22

Описание  
Этот мой датчик температуры на кухне

Комната

Отправить

Рисунок 16 – Страница редактирования информации об устройстве

д) представления технические:

1) HomeProfileView – представление на рисунке 17 для отображения информации, связанной с авторизированным пользователем.

Профиль

Исмагилов Антон  
mr\_tonik  
Электронная почта:  
oldvertu@gmail.com  
Уникальный топик:  
devices/3f45ba9/connected

Devices: 3      Script: 5

Show devices      Show scripts

Рисунок 17 – Страница профиля пользователя

2) HomeMainView – представление на рисунке 18 для отображения технических данных для авторизированного пользователя.

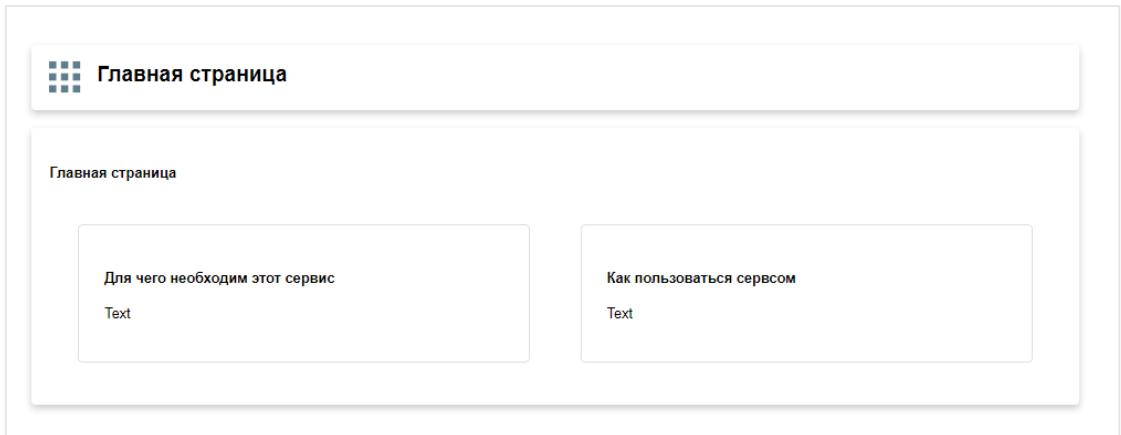


Рисунок 18 – Главная страница приложения

### 3.2.3 Определение моделей данных

Подключение к системе управления базами данных и работа с таблицами MySQL на платформе Node.js может осуществляться различными способами. Одним из способов является использование стандартных драйверов, которые осуществляют связь веб-приложения с базой данных. Главным недостатком данного метода является то, что для запросов к базе данных в контроллере сервера необходимо использовать SQL-запросы, что усложняет работу. Другой способ – использование технологии программирования объектно-реляционного тображения, (Object-Relational Mapping) которая связывает базу данных с концепциями объектно-ориентированных языков. Одной из таких ORM-библиотек является Sequelize [22], которая осуществляет сопоставление таблиц в базе данных и отношений между ними с классами. При использовании Sequelize можно не писать запросы SQL, а работать с данными как с обычными объектами.

Для работы с Sequelize необходим дополнительный файл, который связывает все используемые модели и создает соответствующие таблицы в базе данных, модели все также остаются доступны к использованию в других частях программы. Так необходимы следующие модели представленные на рисунке 19:

а) DeviceModel – модель представления данных об устройствах, подключенных к системе:

- 1) device\_id – уникальный идентификатор устройства;
- 2) device\_name – имя устройства;
- 3) device\_description – описание устройства;
- 4) device\_room – комната, в которой находится устройство;
- 5) device\_topic – топик для связи с устройством;
- 6) device\_type – тип устройства;
- 7) user\_id – уникальный идентификатор пользователя, которому принадлежит устройством.

б) LogsModel – модель представления данных, полученных от устройств, подключенных к веб-приложению:

- 1) log\_id – уникальный идентификатор элемента лога;
- 2) log\_value – данные, которые были получены от устройства;
- 3) device\_id – уникальный идентификатор устройства, от которого получены данные;
- 4) log\_date – временная метка, когда эти данные были получены.

в) ScriptsModel – модель представления сценариев взаимодействия устройств между собой:

- 1) script\_id – уникальный идентификатор сценария;
- 2) script\_description – текстовое описание сценария;
- 3) script\_type – тип сценария;
- 4) script\_datetime – время и дата, в которые должен выполняться сценарий;
- 5) script\_comparison – символ сравнения полученных данных и необходимых для выполнения сценария;
- 6) script\_if – источник полученных данных;
- 7) script\_if\_that – значение необходимое для выполнения условий запуска сценария;

- 8) script\_than – исполнительное устройство котороедолжно быть задействовано при выполнении сценария;
- 9) script\_that – команда отправляемая на исполнительное устройство при выполнении сценария;
- 10) user\_id – уникальный идентификатор пользователя, который создал сценарий.
- г) UsersModel – модель представления данных о пользователе, который авторизирован в системе:
- 1) id – уникальный идентификатор пользователя;
  - 2) firstname – имя пользователя;
  - 3) lastname – фамилия пользователя;
  - 4) username – логин пользователя;
  - 5) email – адрес электронной почты пользователя;
  - 6) password – пароль учетной записи;
  - 7) private\_topic – личный топик для добавления устройств.

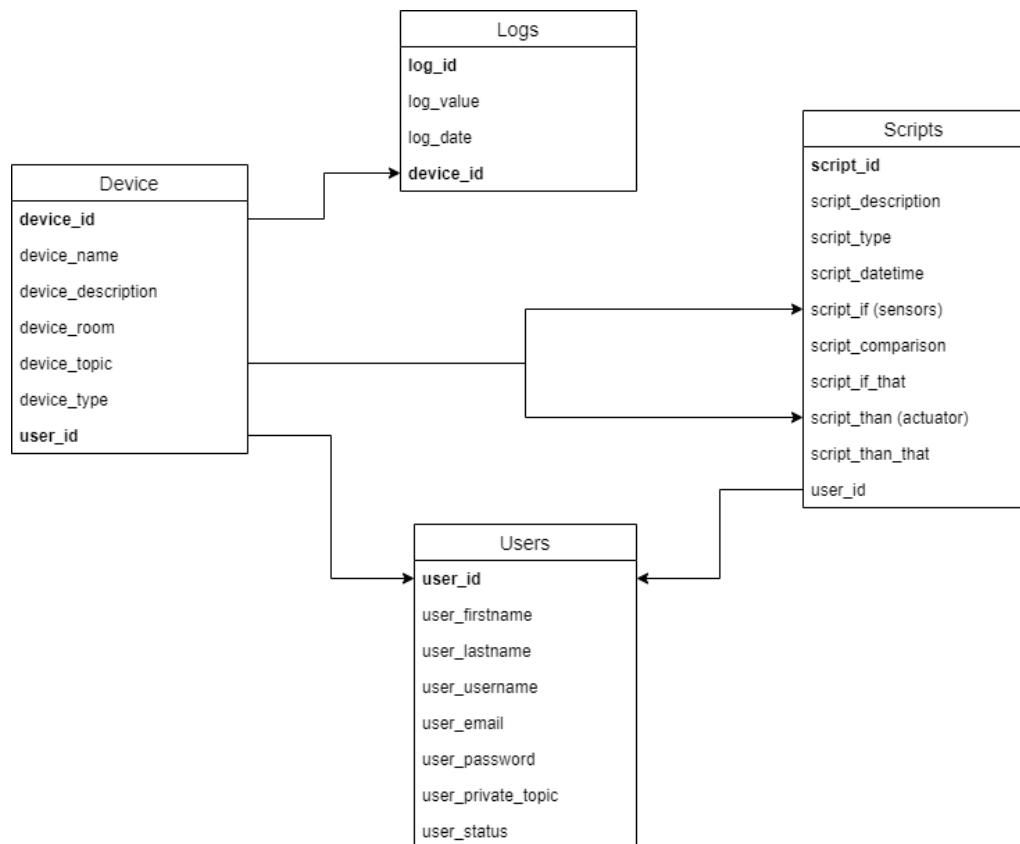


Рисунок 19 – Структура базы данных

### **3.3 Разработка приложения для аппаратной части**

Приложение для аппаратной части представляет собой скетч с Arduino совместимой программой для микроконтроллерной системы ESP8266 в которой реализуются следующие стандартные методы:

- Setup – метод, в котором устанавливается необходимое подключение к датчикам и исполнительным устройствам, осуществляется подключение к беспроводной сети, вызывая функцию wifi\_setup и последующее подключение к MQTT брокеру.
- Loop – основной цикл программы, в котором вызываются такие функции, как: wifi\_reconnect при отсутствии подключения к сети, установка callback для обработки получаемых микроконтроллерной системой сообщений, а также отправка данных, полученных от сенсора по протоколу MQTT с помощью функции mqtt\_senddata.

Подключение к беспроводным сетям для обмена данными осуществляется с помощью стандартной библиотеки функциями:

- wifi\_setup – функция, принимающая SSID и пароль выбранной беспроводной сети и выводящая сообщение об успешном подключении.
- wifi\_reconnect – функция, которая в случае разрыва соединения с беспроводной сетью выполнит повторное подключение к выбранной сети.

Для работы с MQTT реализуются следующие функции:

- mqtt\_callback – функция, прослушивающая сообщения, на которые подписана микроконтроллерная система для последующей обработки сообщения и выполнения команд.
- mqtt\_sendStateSensor – функция,читывающая и формирующая текст в формате json для последующей отправки через MQTT на топик устройства.

- mqtt\_sendHello – функция, отправляющая приветственное сообщение на топик пользователя, содержащее информацию о типе устройства, и топик для обратной связи.

### **3.4 Вывод по главе**

Была разработана структура приложения, соответствующая сформированным требованиям и функционалу, а также соответствующая программной платформе реализации, а также выбраны методы и средства разработки требуемого функционала. В результате было разработано веб-приложение, соответствующее необходимому функционалу и реализованы необходимые модули.

## **4 Инструкция пользователя**

При запуске приложения пользователю предоставляется на выбор два действия представленные на рисунке 20:

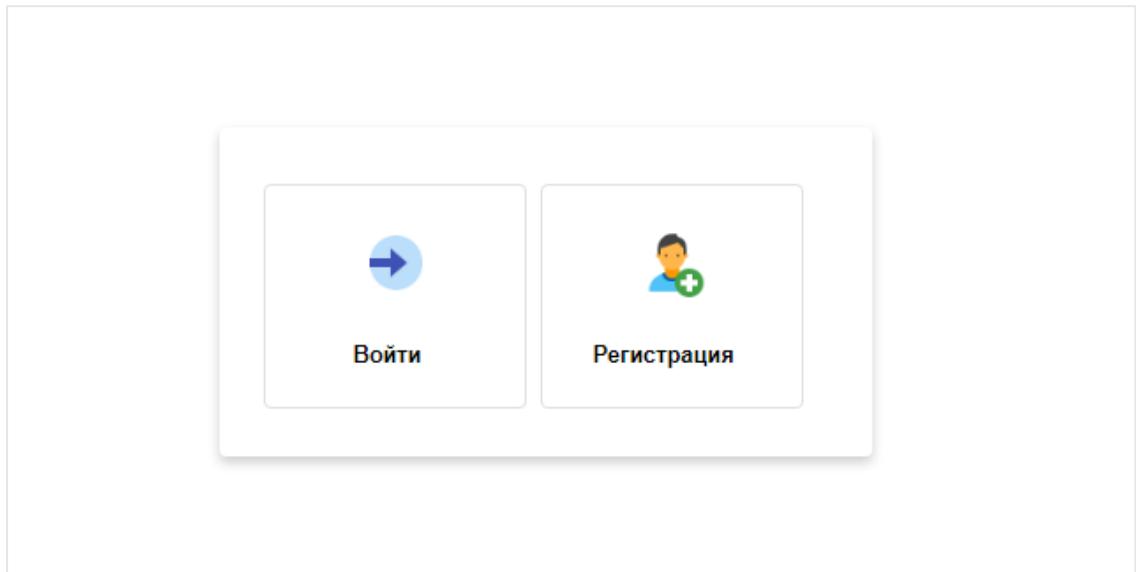


Рисунок 20 – Представление вход или регистрации пользователя

«Войти» – для пользователя, уже зарегистрировавшегося в приложении. При нажатии на данную кнопку пользователь переходит на страницу Входа в приложение.

При выборе функции «Войти» пользователь должен ввести параметры аутентификации представленные на рисунке 21 (адрес электронной почты, и пароль) для входа в приложение. При вводе неправильных параметров пользователь будет перенаправлен на данную страницу, где увидит сообщение об ошибке. При вводе правильных параметров пользователь будет

перенаправлен на главную страницу приложения.

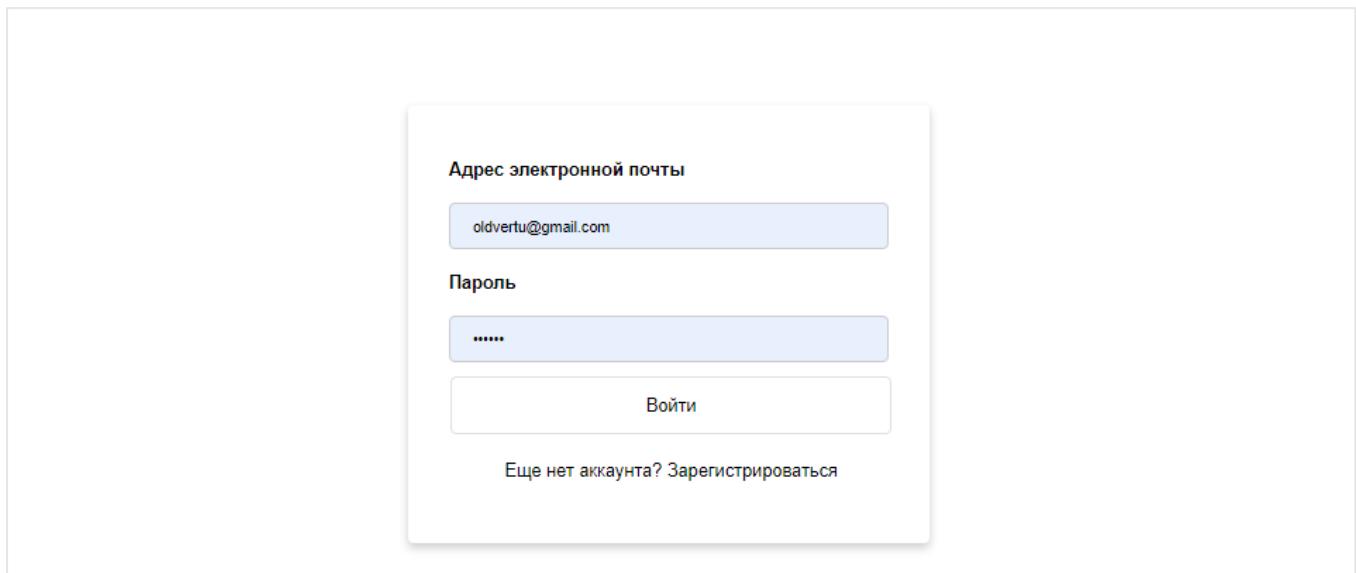
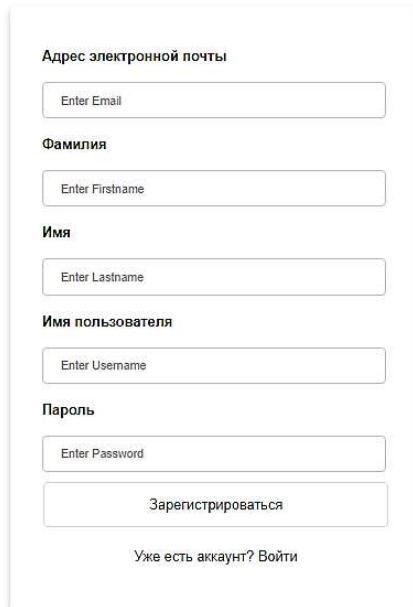


Рисунок 21 – Страница входа в приложение

«Регистрация» – регистрация нового пользователя, который хочет воспользоваться приложением впервые. При нажатии на данную кнопку пользователь переходит на страницу Регистрации в приложении представленную на рисунке 22.

При выборе функции «Регистрация» пользователь должен ввести параметры регистрации (адрес электронной почты, имя и фамилию, имя пользователя в приложении, пароль) для регистрации в приложении. При вводе всех параметров пользователь будет перенаправлен на главную страницу приложения.



The registration form consists of several input fields and labels:

- Адрес электронной почты (Email address) - Enter Email
- Фамилия (Last name) - Enter Fristname
- Имя (First name) - Enter Lastname
- Имя пользователя (User name) - Enter Username
- Пароль (Password) - Enter Password
- A registration button labeled "Зарегистрироваться" (Register).

At the bottom right of the form area, there is a link: "Уже есть аккаунт? Войти" (Already have an account? Log in).

Рисунок 22 – Страница регистрации

После успешной регистрации или авторизации пользователь перенаправляется на главную страницу приложения представленную на рисунке 23. На нем находится основная область, верхний бар, содержащий кнопку открытия меню, заголовок (опционально, кнопки дополнительных функций), и меню навигации для перехода на соответствующие страницы приложения.

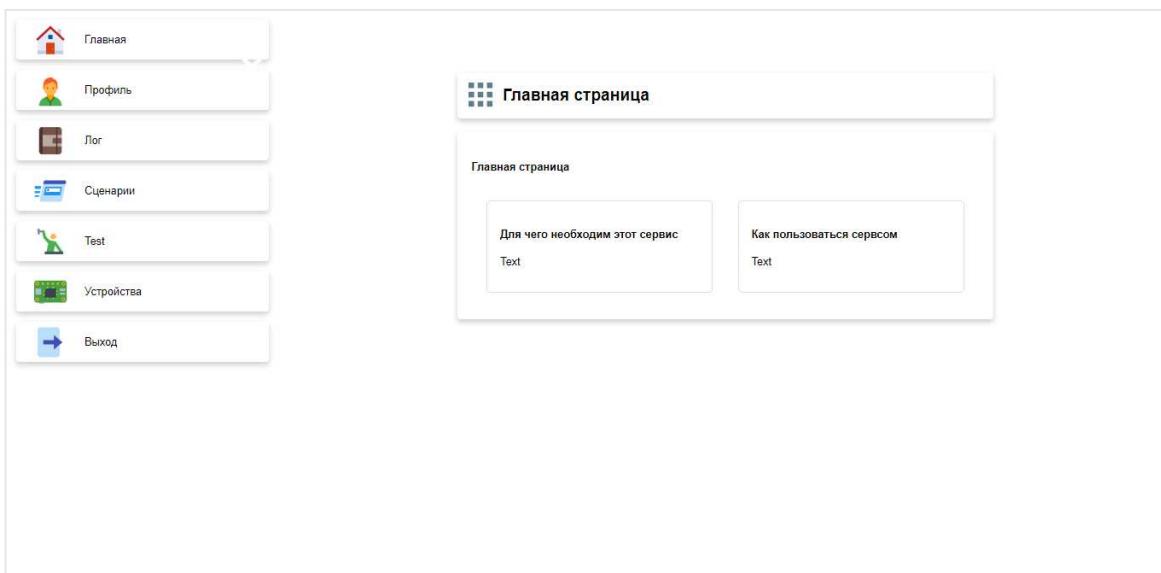


Рисунок 23 – Главная страница приложения

Пользователь может перейти на следующие страницы, несущие соответствующие функции:

а) «Профиль» – на данной странице представленной на рисунке 24 пользователь может просмотреть информацию о своем аккаунте и получить доступ к уникальному топику, который позволит подключать устройства к приложению для авторизированного пользователя. Для добавления устройства необходимо ввести данный топик в устройстве. После успешного добавления устройство отобразится на представлении «Устройства»

При нажатии на кнопку «Показать устройства», пользователь сразу может перейти на страницу «Устройства», либо при нажатии на кнопку «Показать сценарии» пользователь сразу может перейти на страницу «Сценарии»

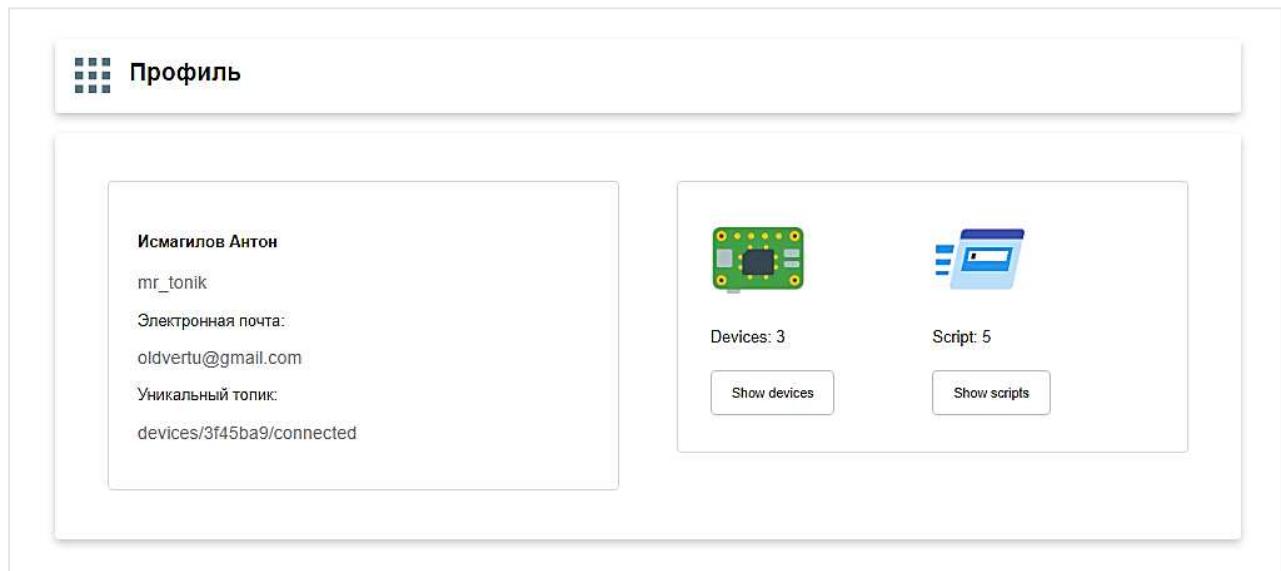


Рисунок 24 – Страница профиля пользователя

б) «Лог» – страница представленная на рисунке 25, отображающее все устройства типа сенсор и дополнительная информация для их идентификации пользователем.

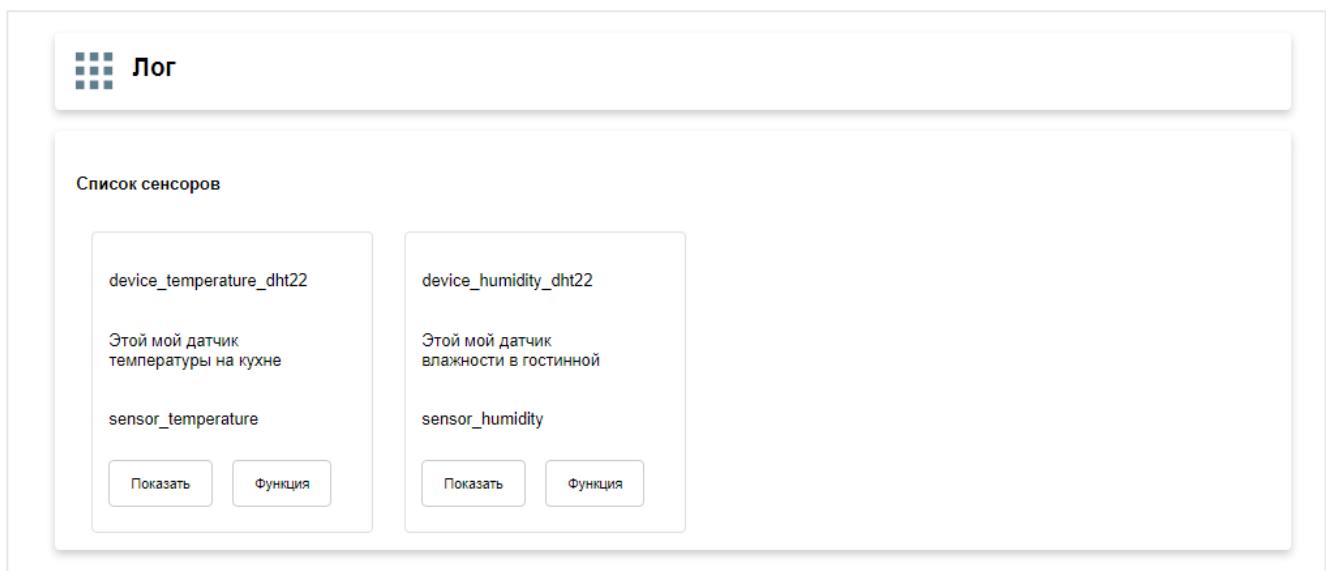


Рисунок 25 – Страница выбора лога

Каждое устройство содержит кнопку «Показать», которая позволит перейти на страницу «Данные лога» представленную на рисунке 26 для отображения информации, которая была получена от устройства в виде графика.

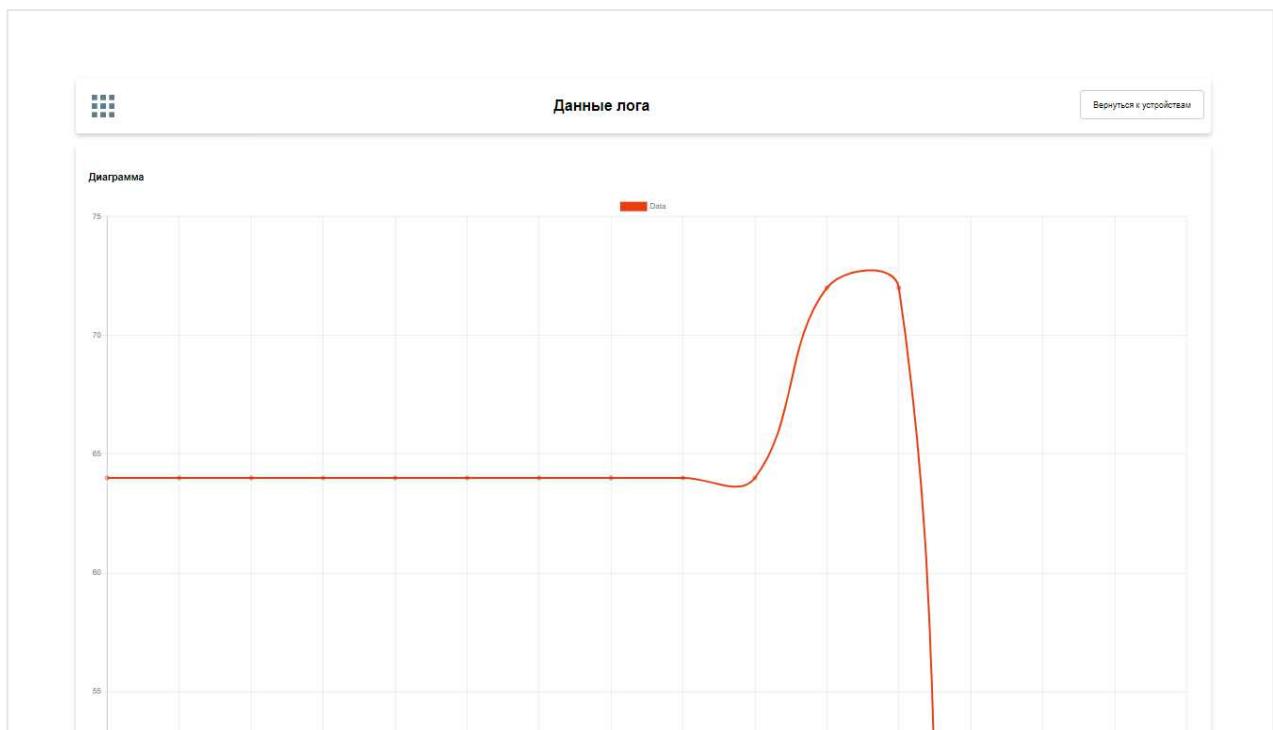


Рисунок 26 – Страница отображающая данные устройства

На данной странице пользователь может увидеть данные, полученные от устройства на графике. При нажатии на кнопку «Вернуться к устройствам» пользователь будет перенаправлен на предыдущую страницу.

в) «Сценарии» – страница приложения представленная на рисунке 27, отображающая все сценарии, которые были добавлены пользователем.

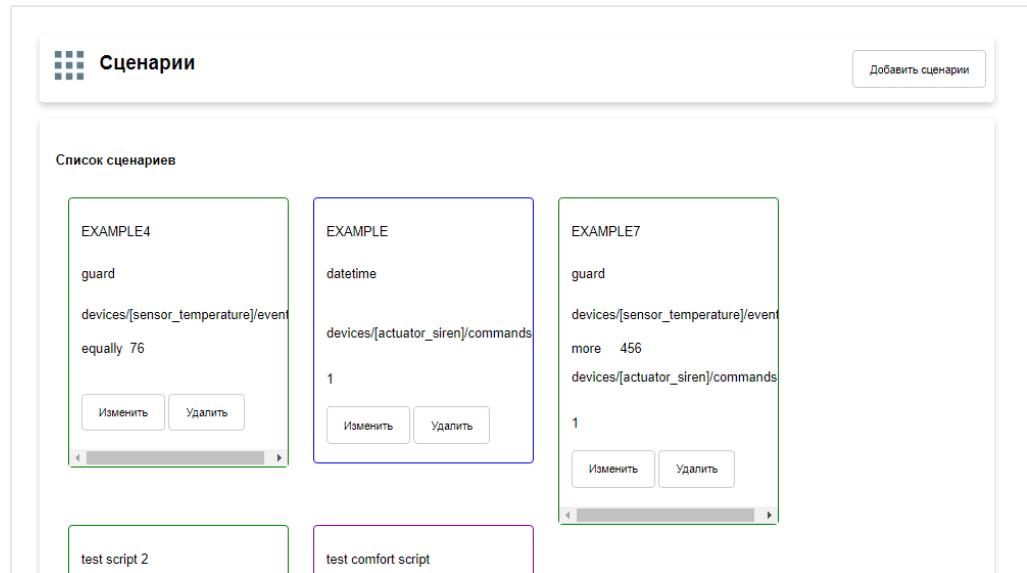


Рисунок 27 – Страница отображения сценариев

Каждый сценарий содержит в себе следующие кнопки:

1) «Изменить» – при нажатии на данную кнопку пользователь будет перенаправлен на страницу изменения сценария представленное на рисунке 28, которое будет содержать информацию о выбранном сценарии, которую можно будет изменить и при нажатии на кнопку «Отправить» сохранить.

Создание сценария

Описание Классический скрипт

Тип Классический

Время dd.MM.yyyy -- ::--

Датчик Предел

Исполнительное устройство

Команда Включить

Отправить

Рисунок 28 – Страница изменения сценария

2) «Удалить» – при нажатии на данную кнопку пользователь удаляет выбранный сценарий.

Верхний бар страницы «Сценарии» содержит кнопку «Добавить сценарий», при нажатии на данную кнопку пользователь переходит на страницу создания сценария представленную на рисунке 29, которая представляет собой форму, содержащую параметры сценария.

Создание сценария

Описание Классический скрипт

Тип Классический

Время dd.MM.yyyy -- ::--

Датчик Предел

Исполнительное устройство

Команда Включить

Отправить

Рисунок 29 – Страница создания сценария

Здесь пользователю предоставляется выбор параметров, которые может содержать сценарий.

- а) Описание – текстовое описание сценария, понятное пользователю для более легкой идентификации.
- б) Тип – в зависимости от типа сценария будет предоставлен выбор различных параметров, которые можно добавить в сценарий, а также определен алгоритм его выполнения. Так пользователь может добавить следующие типы сценариев:
  - 1) Классический – сценарий, которому необходимы такие параметры как устройство, от которого должны быть получены определенные данные, которые устанавливает пользователь, устройство, которое должно быть задействовано и какие данные должны быть отправлены на него.
  - 2) Временной – сценарий, которому необходимы такие параметры, как время и дата в которые будет отправлены установленные данные на выбранное исполнительное устройство.
  - 3) Безопасность – сценарий, которому необходимы такие параметры, как устройство, от которого должны быть получены определенные данные для последующего оповещения по электронной почте.
- в) Время – параметр, устанавливающий временную отметку, когда должен выполниться сценарий, включает в себя время и точную дату.
- г) Датчик – параметр, который устанавливает от какого устройства могут быть получены данные для запуска выполнения сценария.
- д) Предел – в каких числовых пределах должно находиться число, полученное с датчика для запуска выполнения сценария.
- е) Исполнительное устройство – параметр, устанавливающий какое исполнительное устройство должно быть использовано сценарием при выполнении условия.

ж) Команда – параметр, устанавливающий какое значение отправить на исполнительное устройство при выполнении условия сценария. После успешного добавления сценария, пользователь будет перенаправлен на страницу «Сценарии», где сможет увидеть добавленный сценарий.

г) «Устройства» – страница приложения представленная на рисунке 30, в котором пользователь может получить информацию об устройствах, которые были добавлены для использования.

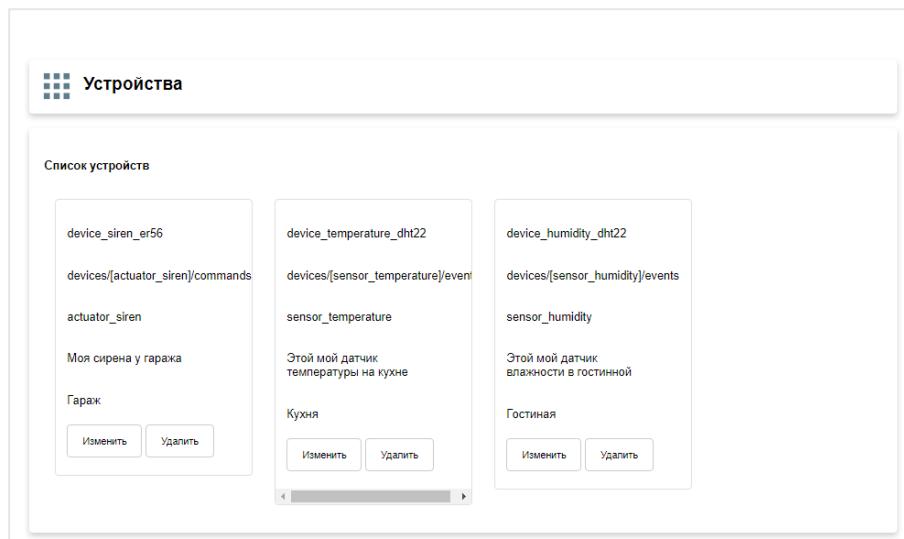


Рисунок 30 – Страница устройств

Каждое устройство содержит в себе следующие кнопки:

1) «Изменить» – при нажатии на кнопку пользователь перенаправляется на страницу «Изменить устройство» представленную на рисунке 31. Страница представляет из себя форму, содержащую в себе параметры устройства. Так пользователь может изменить название, текстовое описание устройства, а также комнату, в которой предполагается использование устройства.

Изменить устройство

Название: device\_temperature\_dht22

Описание: Этой мой датчик температуры на кухне

Комната:

Отправить

Рисунок 31 – Страница изменения устройства

- 2) «Удалить» – при нажатии на кнопку удалить, устройство удаляется из системы и приложение больше не может получать или отправлять данные на устройство.
- д) «Выход» – выход из приложения и пользовательского аккаунта, пользователь перенаправляется на главную страницу приложения.

## **ЗАКЛЮЧЕНИЕ**

В процессе выполнения выпускной квалификационной работы целью которой является создание доступной системы управления комфортностью были выполнены следующие задачи:

- был проведен обзор аналогов, благодаря которому был определен список функциональных требований и получено представление о создаваемой системе;
- было проведено проектирование системы;
  - а) определены функциональные требования для следующих модулей:
    - 1) передача данных между устройством и сервером;
    - 2) работа с пользователями;
    - 3) работа со сценариями;
    - 4) работа с логом событий;
    - 5) работа с устройствами;
  - б) определена структура приложения, при которой был проведен выбор средств разработки и реализации для:
    - 1) программной платформы для реализации web-приложения;
    - 2) аппаратной платформы для реализации;
    - 3) модели представления данных;
    - 4) протокола для связи сервера и подключаемых устройств;
  - в) реализовано приложение:
    - 1) с использованием шаблона mvc;
    - 2) разработана серверная часть;
    - 3) разработан web-клиент;
    - 4) разработано приложение для отладки аппаратной части;

- система полностью удовлетворяет полученным требованиям, а именно:

- a) состав системы состоит из сервера, веб-приложения, устройства с подключенными к нему датчиками и исполнительными устройствами;
- б) передача данных осуществляется с учетом возможной потери данных и в корректном виде через протокол mqtt;
- в) вывод данных и управление устройствами осуществляется через простой в использовании интерфейс;
- г) система решает следующие задачи:
  - 1) управление освещением;
  - 2) управление другими системами (безопасности, климата);

Если рассмотреть требуемые функциональные возможности, то было реализовано:

- возможность включения/выключения подключенных к системе осветительных приборов, как по состоянию внешних условий, так и по активации пользователя;
- управление другими системами (безопасности, климата);
- пользовательский интерфейс в виде веб-приложения;
- серверная часть, реализующая логику обработки поступающих данных.

Данная бакалаврская работа написана в соответствии с нормами СТО 4.2–07–2014 Система менеджмента качества [24].

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Системы умного дома [Электронный ресурс]: Tadviser.ru – Режим доступа: [https://tadviser.ru/index.php/Статья:Системы\\_умного\\_дома](https://tadviser.ru/index.php/Статья:Системы_умного_дома)
2. Глупый умный дом. Часть 2. Xiaomi [Электронный ресурс]: mobile-review.ru – Режим доступа: <https://mobile-review.com/articles/2019/smart-home-3.shtml>
3. Глупый умный дом. Часть 3. IKEA [Электронный ресурс]: mobile-review.ru – Режим доступа: <https://mobile-review.com/articles/2019/smart-home-3.shtml>
4. Глупый умный дом. Часть 2. Apple [Электронный ресурс]: mobile-review.ru – Режим доступа: <https://mobile-review.com/articles/2019/smart-home-2.shtml>
5. Глупый умный дом. Часть 2. Samsung [Электронный ресурс]: mobile-review.ru – Режим доступа: <https://mobile-review.com/articles/2019/smart-home-2.shtml>
6. ГОСТ 30494-2011 Здания жилые и общественные. Параметры микроклимата в помещениях. – Введ. 01.01.2013. – Москва : Стандартинформ, 2014. – 4 с.
7. Построение полноценного MVC веб-сайта на ExpressJS [Электронный ресурс]: Habrahabr – Режим доступа: <https://habr.com/ru/post/192256/>
8. Официальный сайт Node.js [Электронный ресурс]: Node.js – Режим доступа: <https://nodejs.org/en/>
9. Официальный сайт NodeMCUv.3 [Электронный ресурс]: NodeMCU – Режим доступа: [http://www.nodemcu.com/index\\_en.html](http://www.nodemcu.com/index_en.html)
10. Как работает реляционная БД [Электронный ресурс]: Habrahabr – Режим доступа: <https://habr.com/ru/company/mailru/blog/266811/>

11. SQL или NoSQL — вот в чём вопрос [Электронный ресурс]: Habrahabr – Режим доступа: <https://habr.com/ru/company/ruvds/blog/324936/>
12. Официальный сайт MySQL [Электронный ресурс]: MySQL – Режим доступа: <https://www.mysql.com/>
13. Система управления базами данных MongoDB [Электронный ресурс]: MongoDB – Режим доступа: <https://www.mongodb.com/>
14. Data Distribution Service (DDS) [Электронный ресурс]: DDS-Foundation – Режим доступа: <https://www.dds-foundation.org/what-is-dds-3/>
15. Habrahabr Протокол MQTT: концептуальное погружение [Электронный ресурс]: Habrahabr. – Режим доступа: <https://habr.com/ru/post/463669/>
16. Редактор кода Microsoft Visual Studio Code [Электронный ресурс]: Visual Studio Code – Режим доступа: <https://code.visualstudio.com/>
17. Интегрированная среда разработки Arduino IDE [Электронный ресурс]: Arduino IDE – Режим доступа: <https://www.arduino.cc/>
18. Система управления базами данных MySQL [Электронный ресурс]: MySQL – Режим доступа: <http://www.mysql.ru/docs/man/>
19. Программно обеспечение, реализующее MQTT клиент MQTTBox [Электронный ресурс]: Workswithweb – Режим доступа: <http://workswithweb.com/mqttbox.html>
20. Система аутентификации Passport.js [Электронный ресурс]: Passport.js – Режим доступа: <http://www.passportjs.org/>
21. Шаблонизатор Handlebars [Электронный ресурс]: Handlebars – Режим доступа: <http://handlebarsjs.com/>
22. ORM-библиотека Sequelize [Электронный ресурс]: Sequelize – Режим доступа: <https://sequelize.org/>
23. СТО 4.2–07–2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 09.01.2014. – Красноярск : СФУ, 2014. – 60 с.

Министерство науки и высшего образования РФ  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
**«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Космических и информационных технологий  
институт  
Вычислительная техника  
кафедра

УТВЕРЖДАЮ  
Заведующий кафедрой  
О. В. Непомнящий  
подпись инициалы, фамилия  
«        » 2020 г.

### БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника  
код и наименование направления

Программно-аппаратный комплекс управления комфорностью среды  
тема

#### Пояснительная записка

Руководитель



подпись, дата

профессор,  
канд. техн. наук  
должность, ученая степень

Б. И. Борис  
инициалы, фамилия

Выпускник



подпись, дата

профессор,  
канд. техн. наук  
должность, ученая степень

А.Р. Исмагилов  
инициалы, фамилия

Нормоконтролер



подпись, дата

профессор,  
канд. техн. наук  
должность, ученая степень

Б. И. Борис  
инициалы, фамилия

Красноярск 2020