Техническая документация к программному обеспечению

**Структура и возможности приложения**

* Отображение получаемой температуры и состояний защитных датчиков
* Отправка Max\_temp на брокер
* Запись данных за сессию в базу данных
* Просмотр графиков сессий

**Структура базы данных**

* Таблица temp с показателями датчиков

**Принцип работы приложения**

На тестовое устройство поступают показания с 7 температурных и защитных датчиков и 1 контрольного

Устройство отправляет эти данные + Max\_temp (установленную температуру) на mqtt брокер под топиком test/Temp, добавляя в конце hash

Брокер отправляет данные приложению в формате ‘Sample, Max\_temp, Temp\_1, Trigger1 … Trigger\_7, hash’

Приложение проверяет hash

В случае необходимости из приложения можно задать температуру нагрева Max\_temp. Число отправляется на брокер под топиком test/Max, затем на тестовое устройство

При начале сессии (нажатии start в приложении) приложение начинается запись данных в базу данных MySql (добавляя номер сессии, время получения в формате ISO и убирая hash)

При открытии Graph из приложения можно вывести график определенной сессии

**Принцип работы серверной части приложения**

Приложение получает и обрабатывает сообщения с температурой датчиков с MQTT брокера. Оно подписывается на топики test/Max и test/Temp. Получаемое сообщение оно обрабатывает и выводит на экран в виде значений отдельных датчиков. А также выводит графики за определенные сессии. В этом участвует несколько файлов:

* mqtt.js

В этом файле хранятся настройки для подключения к mqtt брокеру и сама функция подключения

* utlis.js

В этом файле хранится асинхронная функция для подключения к базе данных (connect), библиотека для проверки хэша (crcTable) и сама функция (crc16(s)), функция для добавления данных в базу данных (insertMessageIntoDatabase) и функция для вывода из базы данных номера последней сессии (initializeDatabase).

* confirmSessionRoute.js

Данный хэндлер отвечает за показ графика определенной сессии (число которой было введено) и находится по адресу /confirmSession. Серверная часть выводит из базы данных все строки, где session = input, затем данные отображаются в canvas, где x = timestamp (колонка с временем)

* getAllRoute.js

Этот хэндлер выводит на экран всю таблицу из базы данных и находится по адресу /getAllSessions.

* getDataBySessionRoute.js

Этот хэндлер запускается при открытии страницы с графиком и выводит график последней сессии. Находится по адресу /getDataBySession. Он находит все записи в базе данных, где session = lastSession и посылает их в mainPage.js.

* getSessionRoute.js

Этот хэндлер ищет в базе данных все данные за заданную дату и выводит какие значения были в колонке session без повторений. Находится по адресу /getSessions.

* graphRoute.js

Этот хэндлер производит переход с главной страницы на страницу с графиком и находится по адресу /graph.

* mainRoute.js

Этот хэндлер возвращает пользователя со страницы с графиком на главную страницу и находится по адресу /.

* sendMessageRoute.js

Этот хэндлер получает числи из html и отправляет на брокер в топик test/Max, затем добавляет в базу данных в колонку Max\_temp. Находится по адресу /send-message.

**Основной серверный файл index.js**

В файле index.js находятся адреса, на которые отправляются запросы с клиентской части для получения данных с брокера или из базы данных. Реализовано это с помощью вспомогательных функций, описанных выше.

* /start

Этот запрос принимает значения min и sec, затем меняет значение переменной start на true, добавляет 1 к номеру последней сессии и ставит таймер на min минут и sec секунд, после чего возвращает значение start на false. Если в течении этого времени с брокера поступят новые данные, то они будут записаны в базе данных.

* /stop

Этот запрос останавливает таймер start и возвращает переменной start значение false.

* /checkUpdates

Html отправляет этот запрос раз в секунду и проверяет пришли ли новые данные с mqtt брокера. Если да – выводит их на экран.

* /getlastSession

Этот запрос выводит число последней сессии из базы данных и записывает в переменную lastSession. Используется при нажатии start и при открытии страницы с графиком.

**Принцип работы клиентской части приложения**

Функционал на клиентской части отвечает за отображение данных с брокера, начало сессий и отображение графика. В этом участвует 2 файла – drawGarph.js и mainPage.js

mainPage.js отвечает за главную страницу, а drawGraph.js – за страницу с графиком.

* mainPage.js

При нажатии на кнопку Start приложение принимает значения min и sec, а также checkbox (если checkbox выключен его данные не будут записываться в базу данных). Затем спрашивает пользователя уверен ли он, что хочет начать сессию с таким временем и при нажатии на да отправляет запрос на адрес /start. Спустя min минут и sec секунд выводит в консоль ‘session is over’.

При нажатии на кнопку Stop просто отправляется запрос на адрес /stop.

При нажатии на кнопку Set приложение принимает значение из Set temp, спрашивает уверен ли пользователь, затем отправляет значение на адрес /send-message и выводит его в поле Max Temp.

Раз в секунду приложение проверяет есть ли новые сообщения от брокера и, если есть обновляет данные на странице.

При нажатии на See graph происходит переход на страницу graph.html

* drawGraph.js

При открытии страницы с графиками приложение отправляет запрос на адрес /getDataBySession и выводит на экран график последней сессии.

При нажатии на кнопку Last session приложение отправляет запрос на адрес /getlastSession и выводит в поле session number номер последней сессии.

При нажатии на кнопку See session приложение принимает дату из поля Show data from и отправляет запрос с датой на адрес /getSessions. Затем создает в поле session number выпадающий список с номерами сессий за эту дату.

При нажатии на кнопку confirm приложение принимает число из поля session number, спрашивает уверен ли пользователь и отправляет запрос с номером на адрес /confirmSession. Затем из полученных данных строит график.