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
« ____» _____ 2023 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

«Система удаленного доступа к лабораторному оборудованию»
тема
09.04.01 *«Информатика и вычислительная техника»*
код и наименование направления
09.04.01.11 «Вычислительные системы и сети»
код и наименование магистерской программы

Научный руководитель	_____ подпись, дата	доцент каф.ВТ, канд. тех. наук должность, ученая степень	Титовский С.Н. инициалы,фамилия
Выпускник	_____ подпись, дата		Кулаев С.Ю. инициалы,фамилия
Рецензент	_____ подпись, дата	доцент каф. СИИ ИКИТ СФУ, канд. физ.-мат. наук должность, ученая степень	Коченов Д.А. инициалы,фамилия
Нормоконтролер	_____ подпись, дата	доцент каф.ВТ, канд. тех. наук должность, ученая степень	Титовский С.Н. инициалы,фамилия

Красноярск 2023

Содержание

Введение

В текущих реалиях на фоне быстрого развития технического прогресса уже давно поддерживается тренд внедрения современных технических устройств почти во все сферы деятельности человека. Современное общество стремится не только упростить, но и тратить меньше времени на свои обязанности, будь они повседневными или рабочими. При этом технический прогресс непрерывно предлагает средства для достижения данных целей, которые также касаются и сферы обучения.

Произошедшая недавно пандемия только многократно ускорила и без того устойчивое внедрение в учебный процесс различных информационных технологий. Благодаря им обучающиеся теперь имеют возможность проходить курсы в дистанционном формате.

Но все же остаются практические навыки и знания, которые могут быть получены только при очном посещении учебного заведения. Среди них важное место занимает практическая подготовка учащихся при проведении лабораторных и практических работ, а также курсового и дипломного проектирования. Практические навыки, несомненно, являются одним из наиболее значимых компонентов современного обучения.

И хотя в настоящий момент очный формат обучения снова закрепился в качестве основного, потребность в дистанционном обучении все еще остается, особенно это касается лабораторного практикума. Это обуславливается многими причинами.

Разберем основные:

1) высокая стоимость современного высокотехнологичного оборудования, из-за чего встает проблема обеспечения требуемого количества лабораторных мест;

2) невозможность работать с оборудованием из дома;

3) потребность сдачи лабораторных работ обучающимися заочного отделения. Заочно обучающимся было бы гораздо удобнее выполнять и сдавать работу онлайн;

4) при очном обучении преподаватель вынужден следить за сохранностью лабораторного оборудования и за соблюдением техники безопасности студентами.

Таким образом, **актуальность** данной работы вытекает из потребности образовательных организаций проводить дистанционно не только обычные занятия, но и лабораторные, в частности такой потребностью обладает ИКИТ СФУ.

На основании вышеизложенного определена основная **цель диссертационной работы:** исследование, моделирование и реализация решений необходимых для разработки программного комплекса способного обеспечить возможность дистанционной работы с различным лабораторным оборудованием в процессе обучения.

Предполагаемая научная новизна исследования заключается в методе объединения разностороннего оборудования в единый сетевой комплекс удаленного доступа и алгоритмах организации сетевого взаимодействия элементов и систем комплекса, основанных на микросервисной архитектуре позволяющих организовать многопользовательский режим доступа к лабораторному оборудованию, а также организовать процесс обучения в режиме дистанционного доступа с выполнением задач на реальном оборудовании.

Для достижения поставленной целей необходимо решить ряд задач:

- изучить предметную область, проанализировать аналоги разрабатываемой системы;
- сформулировать функциональные требования к системе;
- спроектировать архитектуру системы, определить ее компоненты и принципы взаимодействия между ними;

- провести моделирование взаимодействия компонентов системы по выбранным принципам при работе с оборудованием;
- разработать систему планирования сессий с оборудованием;
- спроектировать и осуществить программную реализацию компоненты системы;
- провести развертку разработанной системы на сервере ИКИТ СФУ;
- провести тестирование всех компонентов системы.

1 Обзор предметной области, анализ и имеющиеся проблемы

1.1 Общий обзор возможных вариантов проведения лабораторных работ

Выполнение лабораторных работ с различным оборудованием при его отсутствии под рукой может осуществляться большим количеством разнообразных способов [4, 8]. Первый вариант – использовать виртуальную лабораторию, и для этого существует множество программ, которые могут имитировать работу большого количества плат, например, proteus и orcad [9]. Виртуальный способ также может быть представлен в виде удаленного смоделированного ресурса. Однако хоть симуляторы и сделаны весьма качественно, но не всегда код, что работает в симуляторе, правильно работает на реальном оборудовании.

Помимо виртуальных лабораторий существуют и удаленные лаборатории с реальным оборудованием. Каждая из лабораторий предлагает свои способы взаимодействия с оборудованием. Существуют даже такие лаборатории, где для воздействия на оборудование в качестве «робота» задействуют преподавателя. В таких лабораториях студенты дают инструкции «роботу», а он их выполняет, иногда указывая на ошибки в инструкциях [30].

Однако существуют более автоматизированные варианты взаимодействия с реальным удалённым оборудованием. Примером могут выступать очень распространенные лаборатории, основанные на продукции компании National Instruments(NI), использующие среду LabVIEW. Такие лаборатории часто предполагают подключение по удаленному рабочему столу к компьютеру, к которому подключены различные устройства от NI, и на котором должна быть установлена среда LabVIEW. После подключения к удаленному рабочему столу студенту нужно только открыть LabVIEW, и там

он сможет свободно взаимодействовать с оборудованием и программировать его [6, 28, 29].

Некоторые лаборатории используют платы, такие как arduino или подобные им [3, 17], программируя их таким образом, чтобы они могли, например, эмулировать различные воздействия на целевом микроконтроллере или выполнять некоторые действия в экспериментальной среде. Этот тип взаимодействия является более гибким, поскольку подача команд в arduino может осуществляться не только с помощью удаленного рабочего стола, но и с помощью веб-сервера. Для лаборатории, где работа ведется с целевым микроконтроллером, важно также придумать способ для его прошивки [5].

1.2 Анализ требований к удаленной лаборатории в рамках ИКИТ СФУ

Необходимо создать сервис, который можно будет применять в курсе «микропроцессоры» в ИКИТ СФУ. В рамках этого курса основными платами, с которыми работают студенты, являются STK-500, STM32, Altera de 1 soc. Следовательно, необходимо, чтобы лабораторный стенд поддерживал эти платы, и была возможность смены платы на лабораторном стенде без изменения исходных кодов программ. Помимо этого, есть еще множество требований для проведения лабораторных работ с дистанционным оборудованием:

- прежде всего должна быть возможность программирования выбранного набора микроконтроллеров;
- большинство лабораторных работ в курсе микропроцессоров завязаны на работе со светодиодами, поэтому важно обеспечить стабильный видеопоток с изображением оборудования. Он должен приходить в режиме

реального времени, следовательно, необходимо разобраться со способом передачи видеопотока до конечного пользователя с минимальной задержкой;

- выбранный набор плат обладает встроенными элементами взаимодействия, такими как кнопки и свитчи, поэтому необходимо реализовать возможность дистанционного воздействия на них пользователем. При этом воздействия должны иметь минимальную задержку между нажатием в программе и реальным откликом;
- так как лабораторные стенды ограничены в количестве и одновременная работа сразу нескольких пользователей за одной платой затруднительна, необходимо разработать принцип распределения времени работы с оборудованием между пользователями. Также нужно обеспечить многопользовательский режим, чтобы выполнение работы мог контролировать преподаватель;
- кроссплатформенный доступ к оборудованию;
- разрабатываемая система должна быть легко масштабируема. Добавление или изолирование стенда должно сопровождаться минимальной настройкой.

1.3 Аналоги разрабатываемой системы

1.3.1 Система дистанционного управления Казанского государственного технического университета им. А. Н. Туполева

Данная разработка датируется 2009 годом. Она содержит интересные технические и программные решения. Система дистанционного управления Казанского университета (СДУКУ) предназначена для измерительных лабораторных работ по предметам «Электроника», «Основы теории цепей» и «Радиотехнические цепи и сигналы». СДУКУ состоит из следующих компонентов: ПК удаленных пользователей, главный сервер, измерительные

лаборатории. Под измерительной лабораторией может пониматься как обычный ПК, в котором заранее выполнены все эксперименты и присутствуют все данные, так и ПК с подключенным NI ElVIS с заранее собранной электронной схемой для исследования.

Общая схема работы СДУКУ следующая: отправленный удаленным пользователем запрос на измерение передается по сети и принимается главным сервером. После предварительной обработки (идентификация удаленного пользователя, преобразование формата запроса, проверка корректности и т.д.) запрос помещается в очередь, откуда он будет извлечен после освобождения требующегося для его обработки измерительного средства соответствующей лаборатории. Полученные результаты измерений возвращаются удаленному пользователю [1].

Из особенностей стоит отметить то, что одна и та же измерительная лаборатория может использоваться несколькими пользователями в режиме разделения времени. Обработка пользовательских запросов может происходить на основе реальных измерений на физических объектах.

Данная система имеет сходные проблемы, но связана с измерительными задачами.

1.3.2 Реализация недорогой системы дистанционного управления оборудованием

Еще одним аналогом разрабатываемого сервиса можно считать разработку новозеландских студентов. Ими был представлен подход к удаленному доступу и управлению средствами автоматизации для инженерно-практической деятельности. В частности, он решает проблему доступа к машинам и управления ими для задач программирования ПЛК. Они предложили использовать для удаленной работы на оборудовании сочетание планировщика, удаленного доступа к рабочему столу,

графического пользовательского интерфейса и микроконтроллера. В качестве планировщика выступает обычный google документ, в который студенты записываются выбирая время сеанса. Удаленный доступ к рабочему столу подразумевает то, что студент в выбранное им в планировщике время с помощью ПО TeamViewer подключается к ПК, к которому подсоединен один модуль ПЛК (Schneider Compact PLC TM221CE24T), который в свою очередь связан с двумя аппаратными средствами: системой реверсивного конвейера и сортировочной машиной Festo. ПЛК связан с аппаратными средствами промежуточным микроконтроллером, который управляет переключением между аппаратными средствами.

На ПЛК и аппаратные средства направлена веб-камера, которая напрямую подключена к ПК с оборудованием.

Основным авторским ПО является клиентское приложение, в котором отображено видео с камеры и из которого можно переключать оборудование.

Непосредственно программирование ПЛК происходит через специализированое ПО для программирования ПЛК SoMachine Basic [13].

Данная разработка имеет следующие плюсы:

- минимальная задержка видео с камеры;
- возможность программирования ПЛК;
- выбор оборудования для управления через клиентское приложение;
- однако система не лишена недостатков:
- отсутствие контроля доступа;
- отсутствие нормального планирования;
- сложность в масштабировании;
- невозможность дополнительного воздействия на ПЛК, например, посредством кнопок (есть возможность только программировать);
- проблемы с безопасностью.

1.3.3 Дистанционная образовательная лаборатория IoT для микроконтроллеров на базе микросхем STM32

Дистанционная образовательная лаборатория IoT для микроконтроллеров на базе микросхем STM32 была разработана для курса микропроцессорной техники, где студенты учатся работать с основными периферийными устройствами микроконтроллеров STM32F446RE. В этой разработке подразумевается, что у ученика на руках будет все оборудование, а именно:

- два микроконтроллера STM32F446RE, где первый микроконтроллер STM32F446RE выполняет функции мониторинга, а второй такой же микроконтроллер, но который ученики используют уже для своих задач;
- модуль esp8266, используемый для связи с сервером, который создает сетевое соединение между учителем и учениками.

Программная часть состоит из трех компонентов: управляющего приложения учителя, главного сервера, программы устройства мониторинга IoT.

Благодаря этому программно-аппаратному комплексу учитель во время урока или экзамена может подключаться к микроконтроллеру ученика. Функции удаленной системы помогают учителю в образовательном процессе, где он имеет полный контроль над микроконтроллером ученика. Это дает возможность контролировать и наблюдать за активностью учащихся во время занятий. Устройство мониторинга IoT также отправляет информацию о программе микроконтроллера учащегося, над которой он работает.

Система включает в себя функции для исследований, таких как определение напряжения, генерируемого приложением учителя через устройство мониторинга IoT, или определение периода и рабочего цикла ШИМ-сигнала.

Из достоинств данной системы можно отметить то, что она действительно позволяет проводить дистанционное обучение, при этом имеет многофункциональное управляющее приложение учителя, благодаря которому он может оказывать различное воздействие на контроллер ученика [16].

В целом это интересная разработка, но ее главные минусы в том, что ученику необходимо иметь целых 2 микроконтроллера stm32 и iot модуль, которые необходимо соответствующим образом соединить, после чего прошить и настроить iot модуль, что может быть затруднительно.

1.4 Резюме о необходимости проведения своих исследований

В ходе изучения предметной области были выявлены основные способы взаимодействия с удаленным оборудованием и типы удаленных лабораторий. Был составлен ряд общих требований, которые преподаватели хотят видеть в разрабатываемой системе.

Обзор аналогов показал, что известные решения удаленного доступа, как правило узконаправленные, поэтому требуется создать с нуля свой собственный сервис. В результате разработки необходимо добиться масштабируемой архитектуры программного комплекса. Проработать алгоритмы организации сетевого взаимодействия элементов и систем комплекса, основанные на микросервисной архитектуре и позволяющие организовать многопользовательский режим доступа к лабораторному оборудованию с минимальным откликом, а также организовать процесс обучения в режиме дистанционного доступа с выполнением задач на реальном оборудовании.

2 Разработка архитектуры системы и принципов взаимодействия пользователей с системой

2.1 Разработка функциональных требования к разрабатываемой системе

Функциональные требования выражены в виде диаграммы прецедентов (рисунок 1).

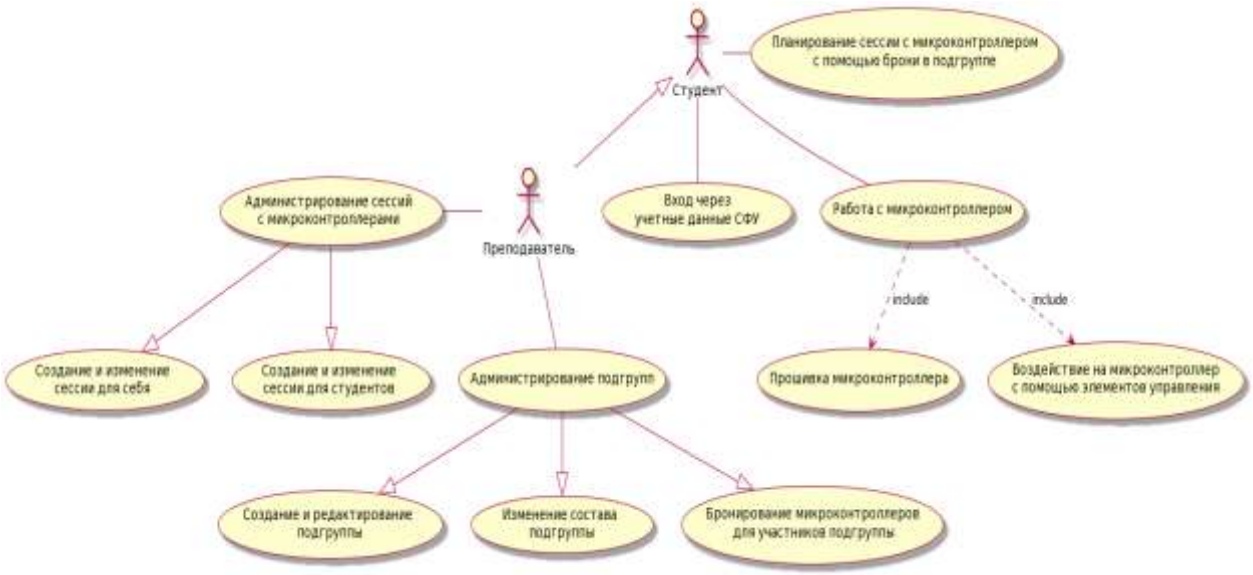

Рисунок 1 - Диаграмма прецедентов

Прежде всего стоит отметить, что в системе необходимо обеспечить наличие двух ролей: преподавателя и студента. Каждая из ролей помимо общего функционала имеет свой набор дополнительных функций.

Описание прецедентов:

1) Прецедент «Вход через учетные данные СФУ». Доступ к функции имеют все роли;

Так как система разрабатывается под конкретный университет, то для упрощения получения доступа пользователей к системе стоит добавить

возможность осуществлять аутентификацию с использованием БД СФУ. Это позволит осуществлять вход в систему с помощью учетной записи СФУ, которая в свою очередь имеется у любого студента и преподавателя, избавив пользователей от необходимости дополнительной регистрации.

2) Прецедент «Администрирование сессий с микроконтроллерами». Доступ к функции должны иметь только пользователи с ролью преподаватель;

Для того чтобы пользователь мог работать с каким-то микроконтроллером ему необходимо иметь созданную именно для него сессию. Возможность создания и изменения данных всех сессий обязательно должна присутствовать только у преподавателя. Он должен быть способен создавать и изменять сессии как для себя, так и для других.

Теперь, если студент хочет поработать с каким-то микроконтроллером, ему необходимо напрямую обращаться к преподавателю и просить создать сессию. Создание всех сессий для студентов таким подходом очень неудобно как для преподавателя, так и для студентов, из-за этого вытекает необходимость в предоставлении пользователю самому создавать сессии.

3) Прецедент «Администрирование подгрупп». Доступ к функции должны иметь только пользователи с ролью преподаватель;

Для того, чтобы у преподавателя не было необходимости создавать все сессии самолично, предлагается внести в систему возможность объединять студентов в подгруппы и создавать в рамках подгруппы бронь на оборудование. Под бронью подразумевается выделение микроконтроллера на некоторый промежуток времени, в котором участники подгруппы смогут самолично создавать себе сессии. Каждая бронь должна иметь фиксированную длительность одной сессии и максимальное количество сессий в день для одного студента.

В рамках администрирования подгруппы подразумевается функционал по созданию подгруппы, изменение ее состава и создание броней в ней.

4) Прецедент «Планирование сессии с микроконтроллером с помощью брони в подгруппе». Доступ к функции должен иметь только пользователь с ролью студент;

Данная функция подразумевает, что у любого студента состоящего в подгруппе, у которой есть бронь на микроконтроллер, должна быть возможность создать себе сессию для работы с этим микроконтроллером. Созданная сессия должна быть доступна для подключения только для преподавателей и студента создавшего эту сессию.

5) Прецедент «Работа с микроконтроллером».

В рамках сессии с микроконтроллером у пользователя должна быть возможность прошивать микроконтроллер файлом с кодом предварительно скомпилированной программы. Помимо прошивки платы должна быть возможность оказывать физическое воздействие на плату, например, нажатия на кнопки, переключение свича, изменение значения переменного резистора. В визуальном интерфейсе каждый микроконтроллер должен иметь свой набор элементов управления, который повторяет реальный на самой плате.

2.2 Описание лабораторного стенда и разработка структурной схемы системы

В данной работе предлагается использовать полностью микросервисную архитектуру, которая поможет добиться всех поставленных требований.

Микросервисная архитектура подразумевает модульный подход к разработке программного обеспечения. Он базируется на обеспечении удаленного по стандартизированным протоколам использования распределённых, слабо

связанных, легко заменяемых компонентов (микросервисов) со стандартизированными интерфейсами [18].

Стоит отметить, что уже есть примеры удаленных лабораторий, построенных на микросервисах, но в них решаются свои специфические задачи и для данной работы будет определен свой набор микросервисов [14].

Прежде всего, необходимо ввести определение лабораторного стенда, а вместе с ним определить первый микросервис. Под лабораторным стендом понимается совокупность устройств, плат и ПО позволяющая дистанционно управлять микроконтроллером. В перечень ПО входит первый микросервис, а именно сервер управления оборудованием. Он будет управлять платами удаленно, получая команды по сети и передавая их на плату с помощью скриптов, после чего возвращать результат выполнения команд. При этом данный веб-сервер должен иметь возможность управлять несколькими микроконтроллерами независимо от их типа.

Сама схема лабораторного стенда создавалась совместно с рабочей группой. Пример подробной схемы лабораторного стенда, задействованного в данной работе, изображен на рисунке 2.

Рисунок 2 - Схема лабораторного стенда

Лабораторный стенд состоит из следующих аппаратных элементов:

- персональный компьютер;

- IP-камера;

- плата управления;

- конечные устройства – набор из плат STM32, STK-500, Altera de1 Soc.

К ПК напрямую подключаются все платы лабораторного стенда. Для взаимодействия с платой управления и конечным устройством на ПК установлены соответствующие драйверы и скрипты, разработанные другими членами рабочей группы. Также предполагается развернуть на нем разрабатываемый сервер управления оборудованием.

IP-камера выполняет функцию наблюдения за конечным устройством стенда. У неё имеется встроенное программное обеспечение, в том числе web-сервер, необходимый для передачи видеопотока.

Плата управления предназначена для управления периферийными устройствами конечного устройства, с которыми должен физически

взаимодействовать пользователь (например, нажатие кнопки). На плате управления имеется специальная прошивка, являющаяся драйвером конечного устройства. С помощью неё происходит управление периферийными устройствами.

От системы необходимо добиться того, чтобы в нее можно было добавлять как можно больше микроконтроллеров. Однако число портов для подключения плат у любого компьютера ограничено совсем небольшим количеством. Решить данную проблему можно обеспечив возможность добавлять в систему любое количество независимых друг от друга лабораторных стендов. Такой подход даст не только возможность добавлять любое количество плат, но и позволит территориально размещать лабораторные стенды в разных местах. Помимо того, что это решает проблему масштабируемости, оно также обеспечивает балансировку нагрузки между лабораторными стендами. Итак, был описан первый микросервис, а именно микросервис управления оборудованием.

Для работы с БД можно создать микросервис, и оптимальнее всего реализовать его с использованием CMS. Зачастую любая CMS из коробки имеет множество возможностей, которые упростят разработку системы.

CMS может осуществлять контроль доступа к данным, также она имеет административную панель, в которой можно изменять структуру БД и данные. Помимо сказанного, CMS умеет автоматически генерировать API для работы с созданными таблицами, через которые другие микросервисы могут работать с данными.

Итак, в системе будет множество независимых друг от друга лабораторных стендов, к которым надо как-то обращаться для работы с платами. Сделать это можно было бы напрямую, но в целях безопасности каждого лабораторного стенда, необходимо ограничить доступ к ним по сети, и оставить возможность обращения, к каждого стенду напрямую только с

одного ip адреса [20]. Для этого необходимо создать еще один микросервис, который будет выполнять роль посредника между клиентами и лабораторными стендами. Помимо общения с лабораторными стендами, этот микросервис также будет взаимодействовать с микросервисом с CMS. Также он может стать связующим звеном при интеграции системв с другими удаленными лабораториями в будущем [24].

Далее необходимо реализовать клиент. Сделать это можно разными способами и самый оптимальный вариант выполнить его в виде SPA приложения. Это даст определенные плюсы. Во-первых, SPA приложение обеспечивает кроссплатформенность, так как оно является обычным веб-сайтом. Запустить его можно в любом браузере, а он в свою очередь есть на любой платформе. Во-вторых, отделение разработки клиентской части от серверной, что обеспечивает их модульность по отношению друг к другу. В третьих, сама разработка будет происходить с давно проверенными и повсеместно используемыми языками CSS, HTML, JavaScript и TypeScrtipt.

В виду того, что в качестве клиента было решено использовать SPA приложение, то для его обслуживания и выдачу пользователю нужен какой-то веб-сервер. Среди описанных выше микросервисов уже присутствует веб-сервер, который может выполнять эту задачу, а именно сервер посредник.

Осталось добавить последний микросервис в систему, которым является веб-сервер передачи видеопотока или же, другими словами, медиа сервер. Данный сервис нужен для снижения нагрузки с каждой камеры в случае, если доступ к камере понадобиться одновременно нескольким пользователям. Также он нужен для изолирования отдельных камер с лабораторных стендов и преобразования данных из исходного RTSP протокола в более современный и понятный браузеру протокол. Так как разработка с нуля такого сервера может занять очень большое число времени, то самым оптимальным вариантом является поиск и использование в системе готового сервиса с открытым исходным кодом. При поиске

главным параметром медиа сервера является способность проксировать исходный видеопоток с использованием скоростного протокола, который проще всего встроить в html. Наиболее подходящим для этого протоколом является WebRTC.

В результате была получена структурная схема системы показанная на рисунке 3.

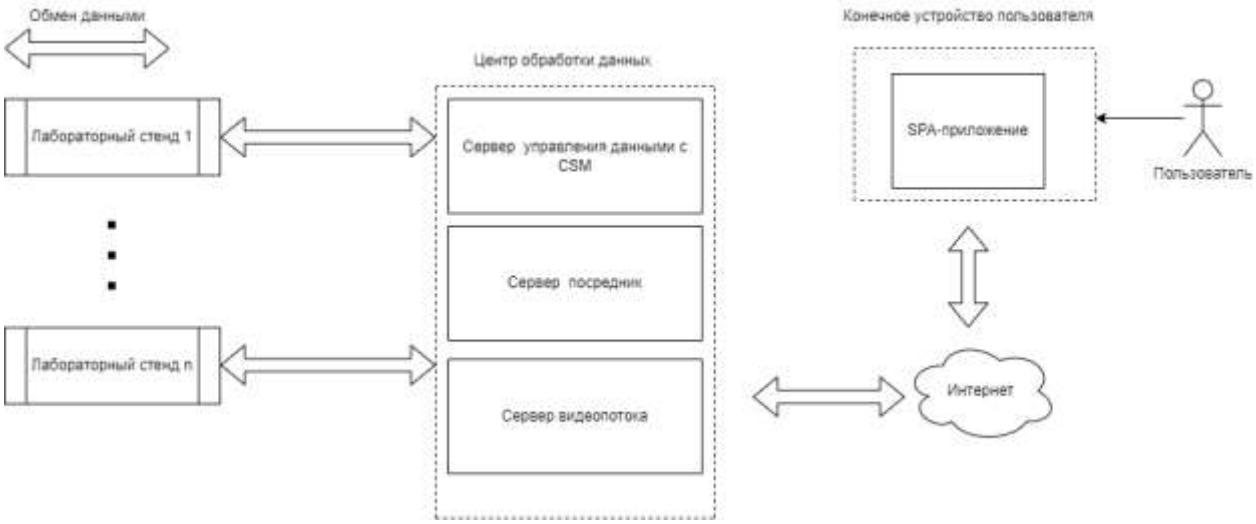

Рисунок 3 – Структурная схема системы

2.3 Выбор способов и принципов взаимодействия между компонентами системы. Моделирование взаимодействия компонентов

Исходя из диаграммы прецедентов можно сделать вывод, что для всех прецедентов кроме прецедента «Работа с оборудованием» нет нужды в большой скорости отклика, поэтому для них можно использовать стандартный HTTP протокол, что упростит разработку и поддержку работоспособности всей системы. Также стоит отметить, что для этих

прецедентов нет смысла как-то задействовать микросервисы лабораторных стендов и медиа сервер.

Что касается главного варианта использования «Работа с оборудованием», то лучшим вариантом является использование протокола WebSocket. Основным преимуществам протокола WebSocket, по сравнению с HTTP, является поддержание соединения TCP в открытом состоянии, что снижает задержку связанную с передачей пакетов и дает возможность вести двухстороннее общение. Открытое TCP соединение также избавляет от необходимости проверки авторизации пользователя при каждом запросе, как это было бы в случае использования HTTP. Эта особенность WebSocket позволит улучшить один из главных параметров системы удаленного доступа, а именно быстродействие [25]. Кроме того двухстороннее общение позволит передавать информацию о всех действиях с микроконтроллером всем подключенным к сессии пользователям, что обеспечит возможность управления микроконтроллером несколькими пользователям в рамках одной сессии.

Теперь стоит подробно разобрать как будет происходить сама работа с оборудованием. Прежде чем предоставлять пользователю доступ к сессии необходимо проверить авторизован ли он для этого. Чтобы осуществить проверку сервер посредник должен обраться к серверу управления данными и передать ему id сессии, к которой пользователь пытается подключиться, а также токен пользователя. В ответ сервер управления данными, в случае если пользователь авторизован, должен передать серверу посреднику всю информацию об оборудовании относящемуся к сессии, а именно: ip-адрес лабораторного стенда, com порт и тип микроконтроллера, com порт микроконтроллера arduino, id микроконтроллера. В результате сервер посредник установит WebSocket соединение с клиентом. В случае провала авторизации сервер управления данными должен вернуть ошибку, в результате чего сервер посредник откажет в установлении WebSocket

соединения с клиентом. Для прохождение самой авторизации пользователь должен быть либо тем для кого была создана сессия, либо иметь роль преподавателя. Таким образом, преподаватель будет иметь право подключиться к любой сессии с оборудованием.

Когда сервер посредник и клиент установят соединение, пользователь сможет передавать команды на оборудование. Сами команды от сервера посредника до лабораторного стенда будут передаваться по HTTP, а не по WebSocket. HTTP был выбран по нескольким причинам. Во-первых, сервер посредник и микросервис лабораторного стенда находятся в одной локальной сети, территориально располагаются в одном здании, плюс к этому, микросервис лабораторного стенда не имеет никакой авторизации, что в совокупности говорит о том, что скорость передачи HTTP будет адекватной и обеспечивать достаточное быстродействие. Во-вторых, осуществлять саму разработку, отладку и валидацию запросов гораздо проще при работе с HTTP.

Скрипты с командами для оборудования выполняются асинхронно. Поэтому когда команды достигают сервера управления оборудованием. Они должны быть помещены в очередь, а после выполнения сервер управления оборудованием должен вернуть актуальное состояние оборудования на котором выполнялась команда.

Имитационная модель сценария работы с оборудованием представлена на рисунке 4 в виде диаграммы последовательности.

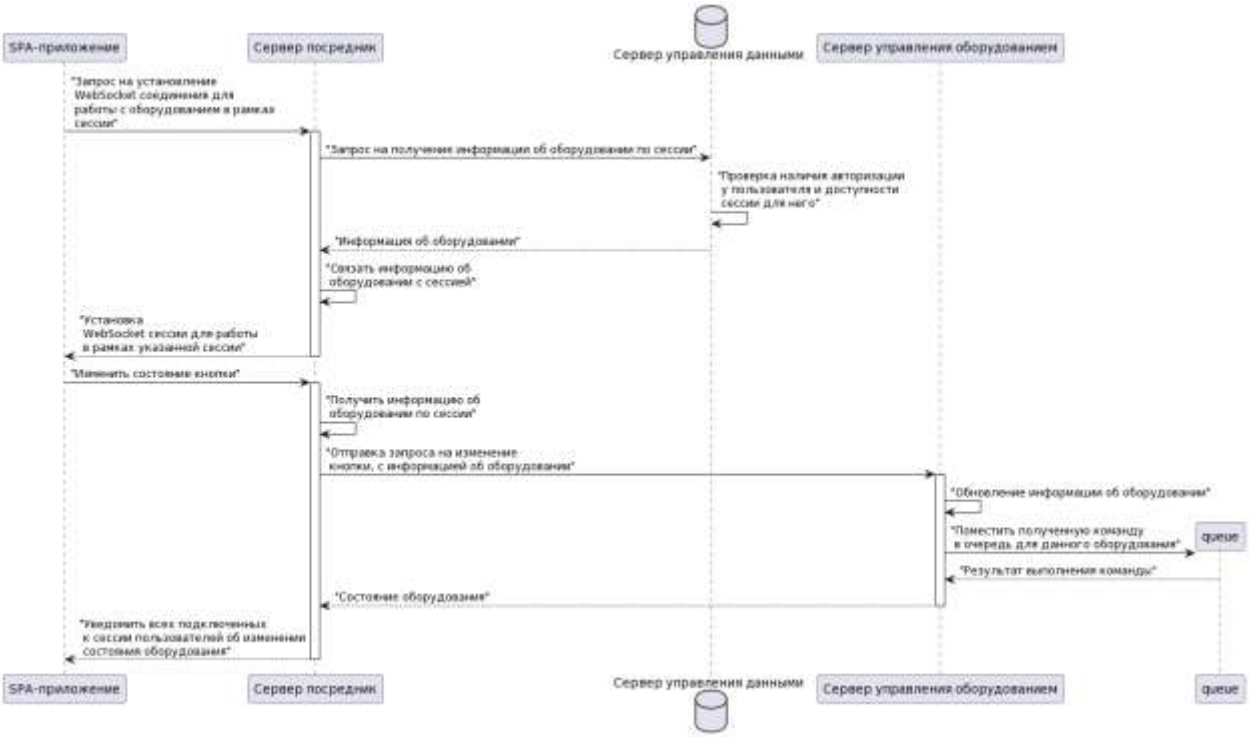

Рисунок 4 – Диаграмма последовательности для варианта взаимодействия пользователя с оборудованием

2.4 Выбор средств разработки

Одним из важнейших этапов проектирования является выбор средств разработки. От него зависит сложность и длительность разработки системы, поэтому к нему нужно подойти крайне ответственно.

Для упрощения и ускорения процесса разработки, разумнее всего, по возможности, использовать один язык для всех компонентов разрабатываемой системы. Исходя из того, что исходный код SPA приложения можно написать только на языке, который поддерживают браузеры, то выбор невелик. В данный момент они поддерживают только язык JavaScript и, соответственно, можно выбрать либо его, либо другой язык, код которого может быть транслирован JavaScript. Учитывая это, в качестве основного языка разработки, был выбран TypeScript.

TypeScript является надмножеством JavaScript, он сохраняет все плюсы чистого JavaScript и дополняет его типизацией, что дает множество дополнительных преимуществ, среди которых поддержка автоматического дополнения кода и выявление ошибок на этапе компиляции. И в дополнении можно сказать, что TypeScript является отличным инструментом для написания серверных приложений.

Выбрав основной язык разработки можно перейти к подбору фреймворков для каждого компонента системы.

В качестве фреймворка для написания SPA приложения был выбран Angular. Angular - это фреймворк, написанный на языке TypeScript и применяемый для создания одностраничных клиентских приложений (SPA-приложение). Angular был выбран по причине того, что код на нем имеет удобную для разработки пользовательского интерфейса MVVM архитектуру и в его арсенале из коробки доступно множество различных директив и функций, которые многократно упрощают манипуляции с DOM-деревом.

Фреймворком для написания сервера посредника и сервера лабораторного стенда был выбран NestJS. NestJS — это масштабируемая серверная среда JavaScript, созданная на основе TypeScript. Сам фреймворк имеет модульную архитектуру и предоставляет современные инструменты для создания высокопроизводительных приложений с использованием различных высокоуровневых абстракций. Уже из коробки он включает в себя библиотеку Express.js и дает удобный интерфейс для работы с ней в виде декораторов. NestJS основан на node.js и стоит подчеркнуть то, что уже существуют успешные реализации системы удаленного доступа с использованием node.js [10, 11, 26].

Немаловажным фактом является то, что создатели NestJS вдохновлялись архитектурой Angular, и знание одного из них облегчает понимание и освоения другого.

Для последнего компонента системы, а именно сервера управления данными была выбрана CMS Strapi. Хоть она и написана на чистом JavaScript, что немного усложняет написание кода, но при этом она имеет большие возможности по расширению встроенного функционала. Strapi из коробки работает с базой данных SQLite, но имеет возможность подключить и другую СУБД. Работа с БД осуществляется по средствам встроенного административного интерфейса, который позволяет не только изменять данные, но и менять структуру таблиц данных.

Контроль доступа к данным в CMS Strapi осуществляется с использованием JWT стратегии. Права пользователей и доступ к данным настраивается в административной панели.

2.5 Выводы по главе

В ходе проектирования архитектуры были прежде всего составлены и проанализированы функциональные требования к системе, было введено понятие лабораторного стенда и рассмотрен его состав. Далее были поэтапно разобраны основные проблемы, которые необходимо решить в результате проектирования, и на основании которых определены основные компоненты системы и способы взаимодействия между ними. Взаимодействие компонентов системы было продемонстрировано с помощью имитационного моделирования, по основному сценарию взаимодействия пользователя с оборудованием. Затем была составлена итоговая структурная схема системы. В конце концов, на основании всего этого были выбраны оптимальные средства разработки для каждого компонента системы.

3 Проектирование компонентов системы

3.1 Проектирование сервера управления оборудованием

Для того чтобы сервер управления оборудованием мог поддерживать любое заранее неизвестное число плат разных типов, необходимо решить несколько важных задач:

- добавить способ хранения состояния плат;
- добавить способ обновления информации о платах. Любая плата имеет набор важных параметров, без которых с ней нельзя будет полноценно работать. В процессе эксплуатации могут, например, измениться порт самой платы и управляющей платы arduino или же может быть подключена новая плата, информация о которой на сервере отсутствует;
- организовать очереди команд для каждой отдельной платы. На пользовательском интерфейсе команды могут формироваться гораздо быстрее чем выполняться. В качестве примера можно привести быстрое нажатие на кнопку. Может случиться так, что скрипт, выполняющий перевод кнопки в нажатое состояние, не успеет завершить свое выполнение в тот момент, когда поступит команда на перевод кнопки в не нажатое состояние. Поэтому необходимо придумать алгоритм, который бы запускал команды в порядке очереди.

Прежде всего, для каждого типа оборудования необходимо создать свои REST API, что позволит отделить логику выполнения запросов для каждого вида платы. В результате, для каждого вида плат будут запускаться свои скрипты.

Для обновления и добавления информации о платах, было решено использовать query параметры, в которых сервер посредник должен передавать всю информацию о плате, на которую отправляется запрос. Это позволит всегда иметь актуальную информацию об оборудовании. Удобнее

всего сделать это можно с помощью интерцепторов. Интерцепторы – это функции, которые выполняются перед обработчиком маршрута и продолжают выполнение сразу после него. В них можно читать содержимое параметров запроса и соответственно обновлять информацию об оборудовании, для которого был предназначен запрос.

Для реализации алгоритма по формированию очереди команд, поступающих на оборудование, стандартных средств JavaScript предназначенных для работы с асинхронностью недостаточно. Основная трудность заключается в передаче информации в контекст выполнения каждого нового запроса о том, была ли завершена предыдущая команда и когда она завершится. В качестве решения этой проблемы предлагается использовать возможности реактивного программирования предоставляемые библиотекой rxjs. Используя класс ReplaySubject библиотеки rxjs можно уведомлять подписчика, которым выступает новый запрос о том, была ли завершена предыдущая команда. Поскольку плат может быть несколько, необходимо использовать словарь, где в качестве ключей будут выступать id оборудования, а в качестве значения - экземпляр класса ReplaySubject, который уведомит нового подписчика о завершении текущей команды.

Далее представлено детальное описание алгоритма: каждый новый запрос на выполнение команды, прежде чем запускать соответствующий скрипт, должен сначала получить объект ReplaySubject из словаря и подписаться на генерируемое им событие завершения команды, а затем создать новый объект ReplaySubject, заменив им предыдущее значение в словаре. Это даст гарантию, что следующий запрос для этой платы будет ожидать завершения предыдущего. Итак, когда самая первая команда в очереди закончит своё выполнение, в объекте ReplaySubject, созданном для этой команды, запускается событие, говорящее следующей команде о своем завершении, после чего следующая команда начнет свое выполнение и это

будет происходит с каждой новой командой. В результате, все запросы для платы будут выполнятся в порядке очереди.

3.2 Проектирование сервера посредника

При проектировании сервера посредника необходимо решить ряд задач:

- обеспечить возможность аутентификации через учетную запись СФУ, с помощью протокола доступа к каталогам ldap и, занесение информации о пользователях в БД системы удаленного доступа;
- осуществление контроля доступа к оборудованию;
- создание WebSocket сессий для пользователей, которые работают с оборудованием и, перенаправление команд от пользователей к соответствующим серверам управления оборудованием;
- уведомление пользователей, которые работают с одними и теми же платами об изменении их состояния.

Рассмотрим механизм аутентификации. Когда на сервер посредник от клиента приходят логин и пароль, прежде всего необходимо проверить их подлинность с помощью сервера управления данными, так как, возможно, такой пользователь уже пытался заходить в систему. При неудачной аутентификации на сервере управления данными, сервер посредник должен попытаться ассоциировать клиента с определённым объектом каталога с помощью функции bind, где в качестве параметров будут переданы логин и пароль пользователя. Это необходимо для проверки введенных пользователем данных, а также для последующего запроса всей необходимой информации о пользователе, от его имени. К числу необходимой информации относятся: ФИО, адрес электронной почты и название учебной группы. После получения этой информации, сервер посредник сможет

отправить запрос на сохранение этой информации вместе с логином и паролем на сервер управления данными.

В результате, сервер посредник получит jwt токен, который будет передан пользователю. В случае неудачи на любом вышеописанном этапе сервер посредник должен отказать пользователю во входе в систему.

Такой механизм аутентификации позволит добавлять пользователей, у которых нет учетной записи СФУ, а также осуществлять успешную аутентификацию при изменении пароля от учетной записи СФУ [23].

Контроль доступа к оборудованию будет работать следующим образом: для получения доступа к сессии с оборудованием пользователю необходимо при инициализации WebSocket сессии передать в handshake свой jwt-токен, а также id сессии с оборудованием, к которой нужно получить доступ. Сервер посредник по средствам middleware должен передать эти данные серверу управления данными. Далее, если токен действителен, а у пользователя есть доступ к сессии, посредник получит информацию об оборудовании, сохранит ее и разрешит установку WebSocket сессии.

Отправка большинства команд для плат будет осуществляться с помощью WebSocket событий. При этом никакую дополнительную проверку по наличию у пользователя доступа проводить не нужно, так как эта операция была выполнена перед создании WebSocket сессии.

Сервер посредник при реагировании на событие должен получить заранее сохраненную информацию об оборудовании и осуществить запрос на нужный сервер управления оборудованием. Когда команда будет выполнена, сервер посредник получит информацию об актуальном состоянии платы. Далее посредник должен будет инициировать WebSocket событие, и передать всем подключенным к сессии пользователям ее состояние.

Отдельно стоит упомянуть еще пару запросов для сервера управления оборудованием, среди которых запрос на прошивку платы и запрос на

проверку доступности сервера с платой. Оба этих запроса будут выполнятся через http протокол. Запрос на проверку доступности сервера с платой будет осуществляться еще до создания WebSocket сессии. Он нужен чтобы проверить сетевую доступность сервера.

Запрос на прошивку платы будет выполняться отдельно, так как сама по себе операция прошивки довольно длительная и скорость передачи данных для этого запроса не имеет большого значения. Помимо этого, есть проблема в том, что передача файлов по WebSocket затруднена. По этому протоколу можно передавать только текст, в следствии чего придется делать дополнительные преобразования по переводу файла в строку в формате base64 и обратно.

3.3 Проектирование сервера управления данными

Как уже было оговорено сервер управления данными основан на CMS Strapi. В Strapi для каждой созданной таблицы в БД также создаются REST API для CRUD операций и плюс к этому API для авторизации.

На рисунке 5 представлена ER-диаграмма БД. На основе каждой таблицы в диаграмме будут автоматически сгенерированы REST API.

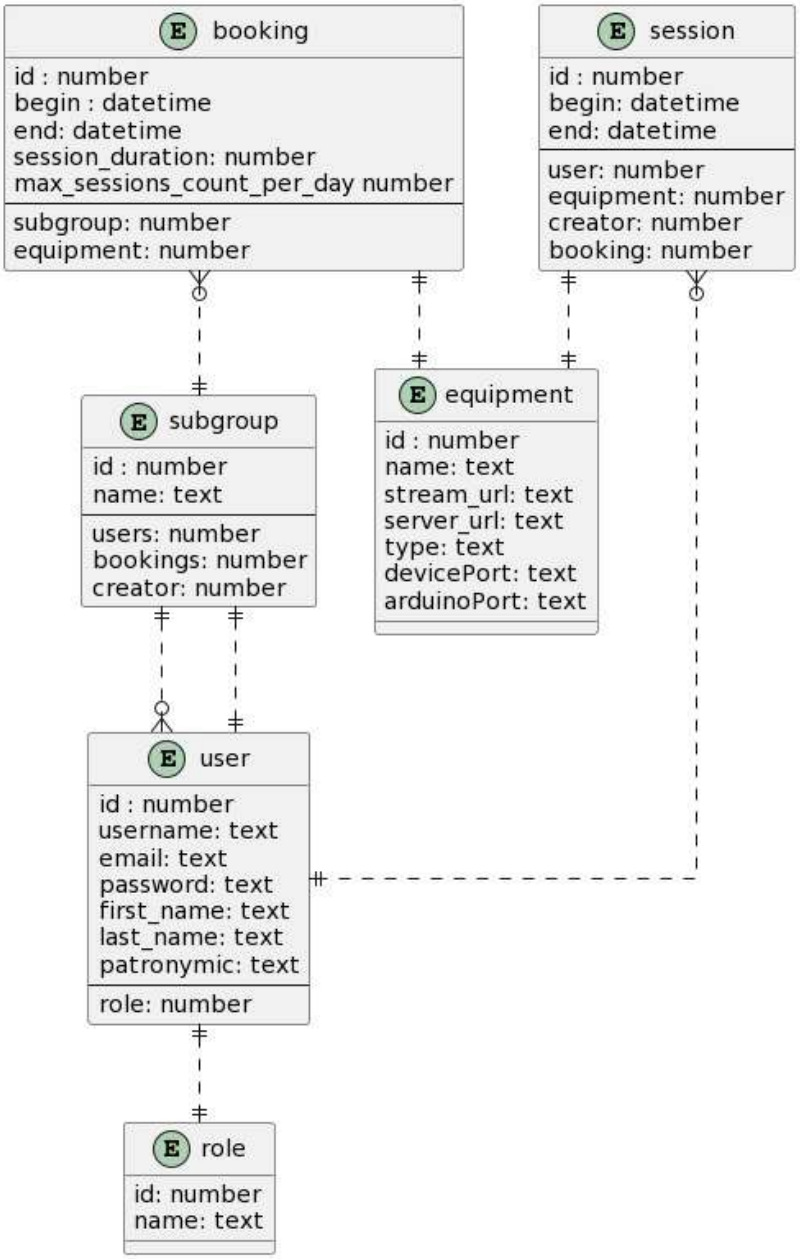

Рисунок 5 - ER-диаграмма БД

Однако для функционирования системы автоматически сгенерированных API недостаточно. Поэтому нужно будет создать новые API и доработать существующий набор.

Необходимо создать следующий дополнительный набор API:

- получение всех сессий текущего пользователя;
- получение всех подгрупп, в которых состоит текущий пользователь;

- создание брони для роли преподаватель, по которой студенты смогут создавать сессии;
- создание сессии по брони для студента;
- создание сессии для любого пользователя от роли преподавателя;
- проверка доступности сессии для текущего пользователя по id сессии.

3.4 Проектирование клиентского приложения

В системе предполагается наличие двух основных ролей, а именно преподавателя и студента. Функционал страниц и количество страниц будут отличаться в зависимости от роли пользователя.

К общим для обеих ролей страницам можно отнести следующие:

- страница аутентификации;
- страница со списком сессий для текущего пользователя;
- страница сессии с оборудованием. На данной странице должны располагаться элементы управления, соответствующие виду оборудования, видеотрансляция и журнал действий с оборудованием;
- страница подгруппы со списком броней и списком участников подгруппы.

К страницам для роли преподаватель можно отнести:

- страница со списком всех начавшихся сессий, к которым можно подключиться;
- страница со списком всех сессий, которые были созданы текущим пользователем, где есть возможность создания, редактирования и удаления сессий;
- страница со списком всех подгрупп, с возможностью фильтрации по текущему пользователю в качестве создателя подгруппы;

- страница подгруппы с дополнительным функционалом, а именно: создание брони на оборудование, изменение состава подгруппы и названия, добавление участников подгруппы по наименованию учебной группы.

К страницам для роли студент можно отнести:

- страница со списком всех подгрупп, в которых состоит текущий пользователь;
- страница подгруппы с дополнительным функционалом, а именно с созданием сессии по брони.

3.5 Выводы по главе

В ходе главы были разобраны основные задачи стоящие перед каждым компонентом системы и разработаны алгоритмы для их решения.

В проектировании сервера управления оборудованием были определены способы для хранения состояния плат, обновления информации о платах, а также разработан алгоритм по созданию очереди команд для плат.

При проектировании сервера посредника был определен механизм аутентификации и контроля доступа к оборудованию, а также алгоритм по передаче команд от пользователя к конечному оборудованию с последующим уведомлением других пользователей об изменении его состояния.

Для сервера управления данными была составлена ER-диаграмма БД, на основе которой будет автоматически сгенерированы CRUD API, а также определен ряд API, которые нужно реализовать вручную.

Для клиентского приложения были определены наборы страницы для каждой роли.

4 Описание разработанной системы и тестирование

После завершения разработки была проведена развертка системы на серверах ИКИТ, доступ к которым можно получить с помощью VPN СФУ.

4.1 Настройка лабораторных стендов

Для добавления нового лабораторного стенда необходимо произвести монтаж плат к компьютеру, запустить сервер управления оборудованием, обеспечить доступность в сети запущенного сервера для сервера посредника. На рисунке 6 представлен общий вид лабораторного стенда.

Рисунок 6 – Общий вид лабораторного стенда

Для того, чтобы система могла работать с новым или измененным лабораторным стендом, необходимо настроить его в панели администратора CMS Strapi. Для этого в панели администратора нужно перейти во вкладку Equipments. После перехода, пользователю будет доступен список со всеми существующими лабораторными стендами. Данный список можно изменять путем удаления, редактирования и добавления лабораторных стендов. Страница со списком лабораторных стендов представлена на рисунке 7.

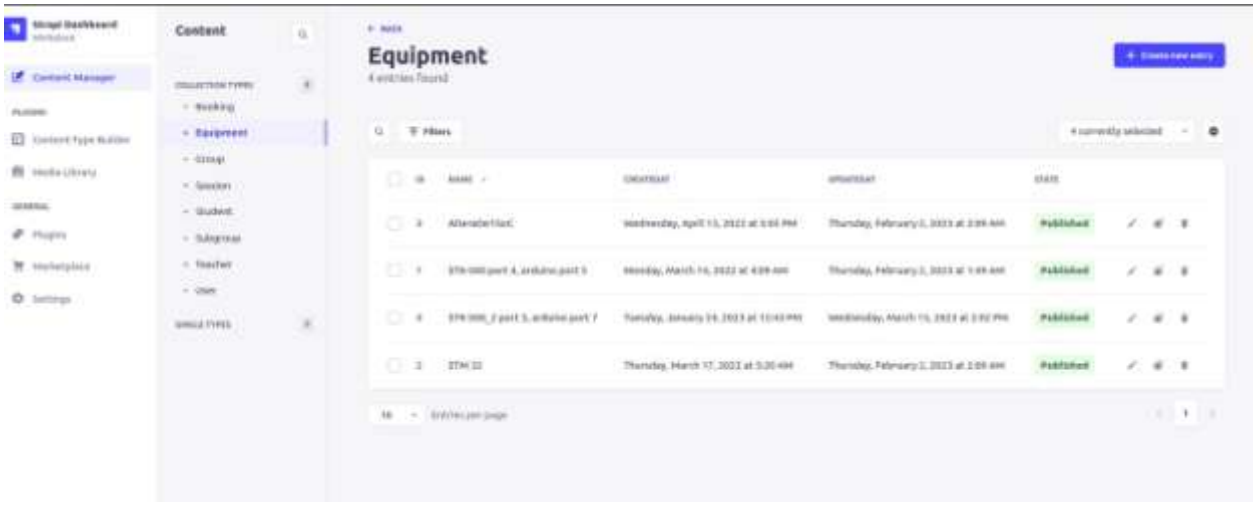

Рисунок 7 – Страница со списком лабораторных стендов в административной панели

Если нажать на кнопку редактирования лабораторного стенда, откроется страница с формой для изменения конфигурации стенда. Для того чтобы стенд начал взаимодействовать с системой, необходимо в форме указать корректные данные со следующей информацией:

- название стенда для отображения на пользовательском интерфейсе;
- ip адрес видеопотока с камеры;
- ip адрес сервера управления данными;
- тип платы;

- номер порта платы;
- номер порта управляющей платы arduino.

На рисунке 8 представлен настроенный лабораторный стенд для платы STK-500.

Рисунок 8 – Страница редактирования конфигурации лабораторного стенда в административной панели

4.2 Демонстрация сценариев работы системы с сессиями

Далее будет приведена иллюстрация работы системы по основным сценариям.

При авторизации с ролью преподаватель первой страницей куда попадает пользователь, является страница «Мои сессии» (рисунок 9). На этой странице перед пользователем представлены 2 списка. Первый список состоит из сессий, которые уже начались, и у пользователя есть возможность к ним подключиться. Второй список состоит из запланированных, но еще не начавшихся сессий.

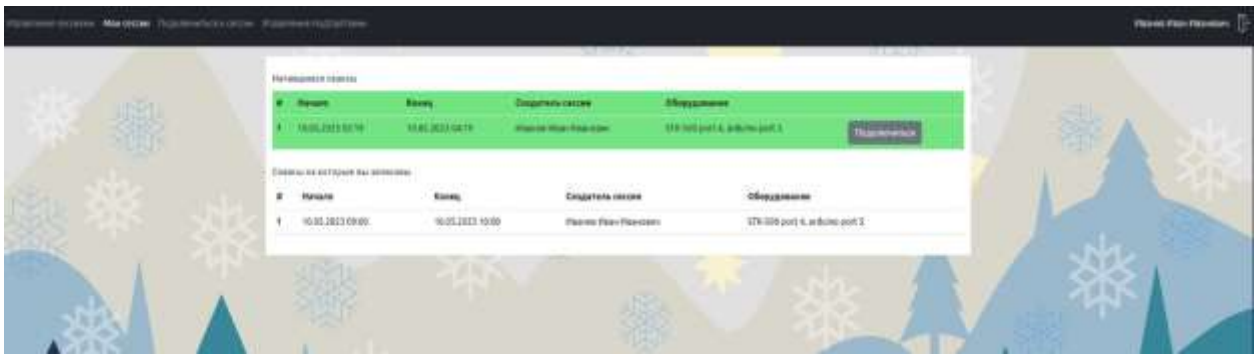

Рисунок 9 - Страница «Мои сессии» пользователя с ролью преподаватель

Чтобы создать новую сессию для себя или для студента, пользователь должен перейти на страницу «Управление сессиями» (рисунок 10). На данной странице пользователь может удалять, редактировать или создавать сессии.

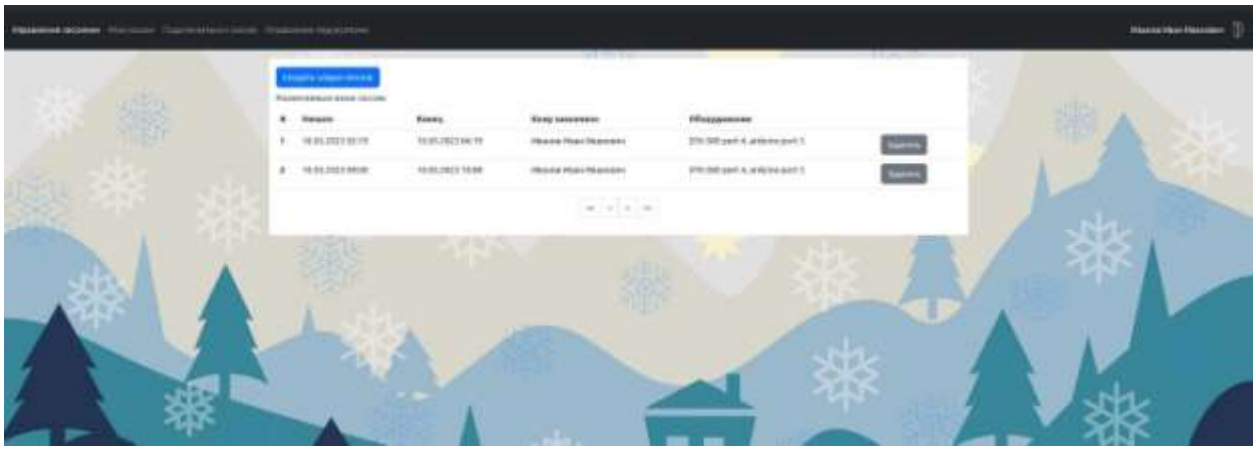

Рисунок 10 - страница управления сессиями

Пользователь может создавать и изменять сессии для как для себя, так и для студентов. Модальное окно редактирования сессии для текущего пользователя представлено на рисунке 11. В обоих формах предоставляется выбор даты проведения сессии, длительности, и оборудования. Форма имеет проверку на корректность введенных данных, например, одно из условий для создания сессии является проверка дат начала и окончания сессии на пересечение со временем уже существующих сессий. Для того чтобы

правильно рассчитать дату и длительность сессии, пользователю выводится список уже существующих сессий на выбранную дату проведения сессии.

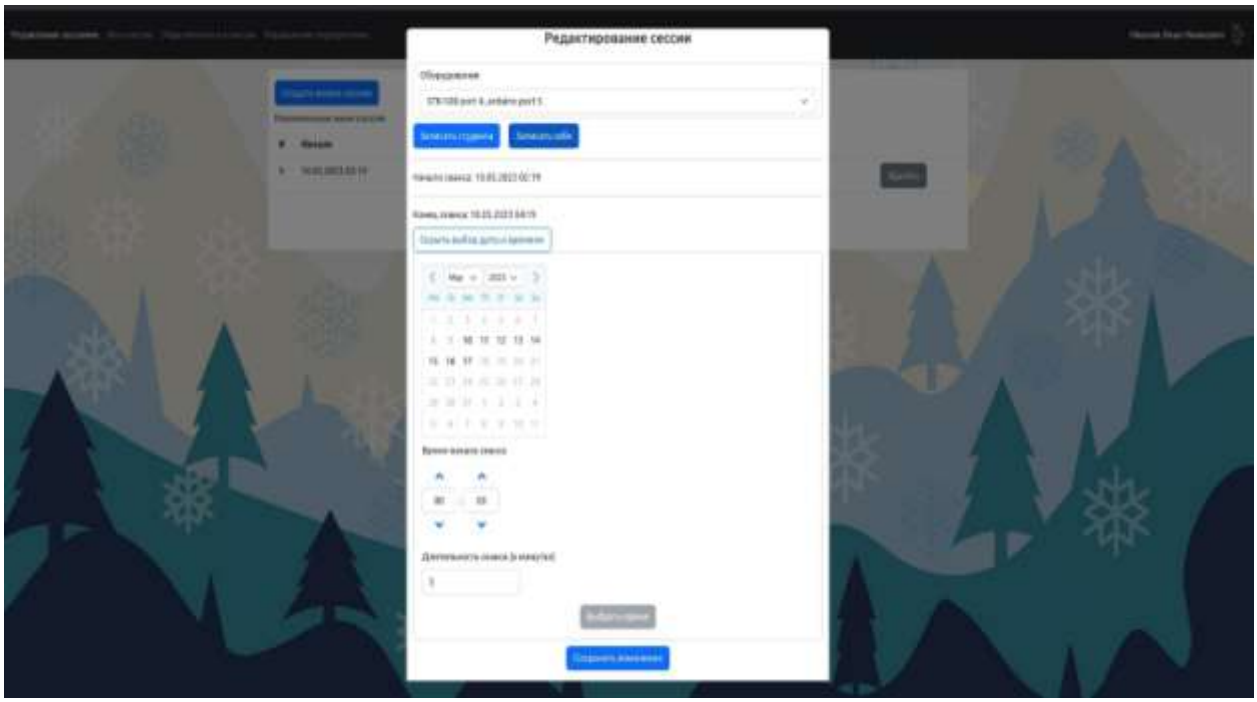

Рисунок 11 - Окно редактирования сессии для текущего пользователя

Создание сессии для студента отличается от формы сессии для текущего пользователя принципом работы и наличием дополнительных полей (рисунок 12). При создании сессии для студента необходимо ввести название учебной группы студента на английском языке, после чего, при правильном вводе названия, появится поле с выпадающим списком для выбора студента из групп. Остальной перечь полей и функционал аналогичен тому, что находится в форме сессии для текущего пользователя.

Отличие принципа работы в том, что когда создается сессия для студента, клиент обращается к посреднику, чтобы тот сам создал сессию. Это необходимо для того, чтобы посредник сам проверил существует ли добавляемый студент в БД системы, и в случае его отсутствии добавил его, а затем создал сессию.

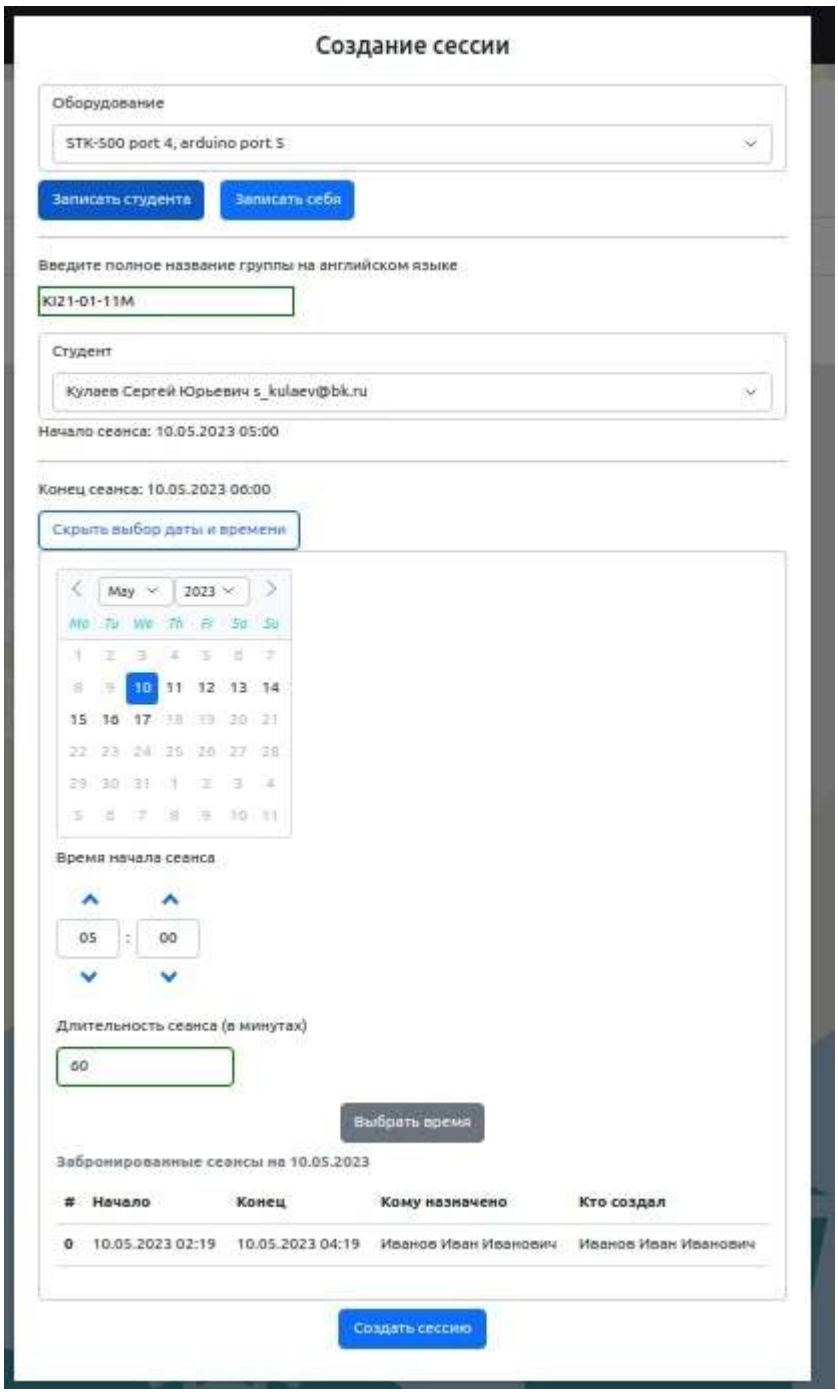

Рисунок 12 - Окно создания сессии для студента

Для подключения к сессиям других пользователей, преподаватель имеет страницу «Подключиться к сессии» (рисунок 13). На этой странице показываются все начавшиеся сессии и те, которые начнутся в ближайшие 10 минут. Дополнительно список сессий можно фильтровать по текущему пользователю в качестве создателя сессии.

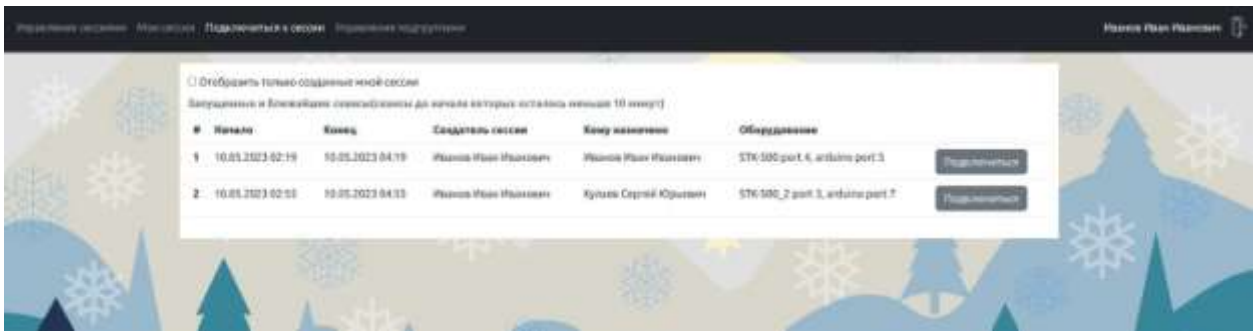

Рисунок 13 - Страница «Подключиться к сессии»

Последней страницей в панели навигации у преподавателя является страница «Управление подгруппами» (рисунок 14). На данной странице можно увидеть список всех существующих подгрупп. Список может быть отфильтрован по текущему преподавателю в качестве создателя. Также здесь имеется возможность перейти на страницу конкретной подгруппы или создать новую.

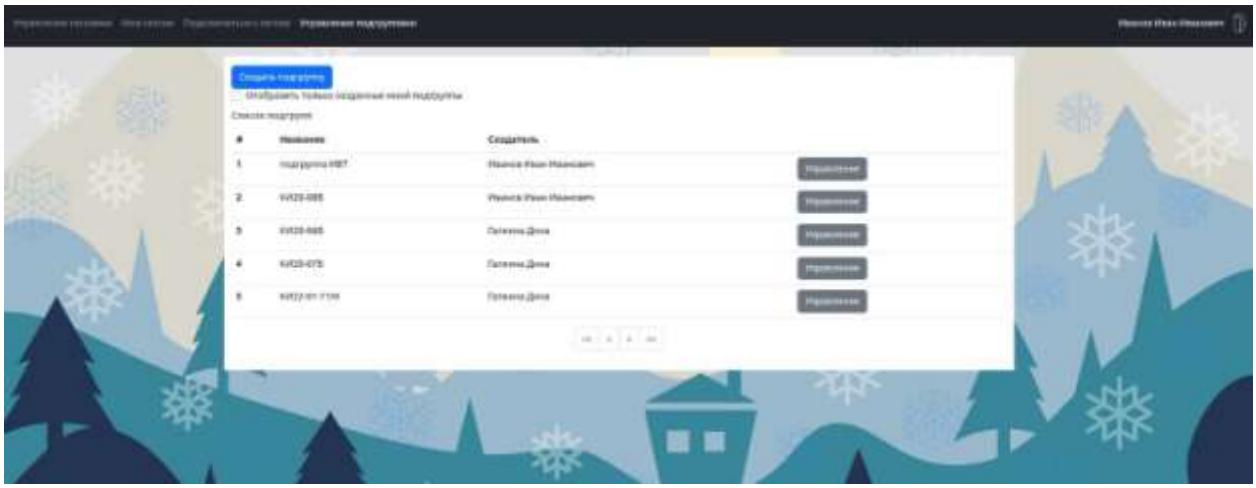

Рисунок 14 - Страница «Управление подгруппами»

На рисунке 15 проиллюстрирована страница подгруппы для роли преподавателя. Тут отображена основная информация о группе, а именно: название, ФИО создателя, список броней, состав подгруппы. Также преподавателю доступен набор действий среди которых: создать или

отредактировать бронь, добавить пользователей в группу из базы СФУ по учебной группе, редактирование названия и состава группы.

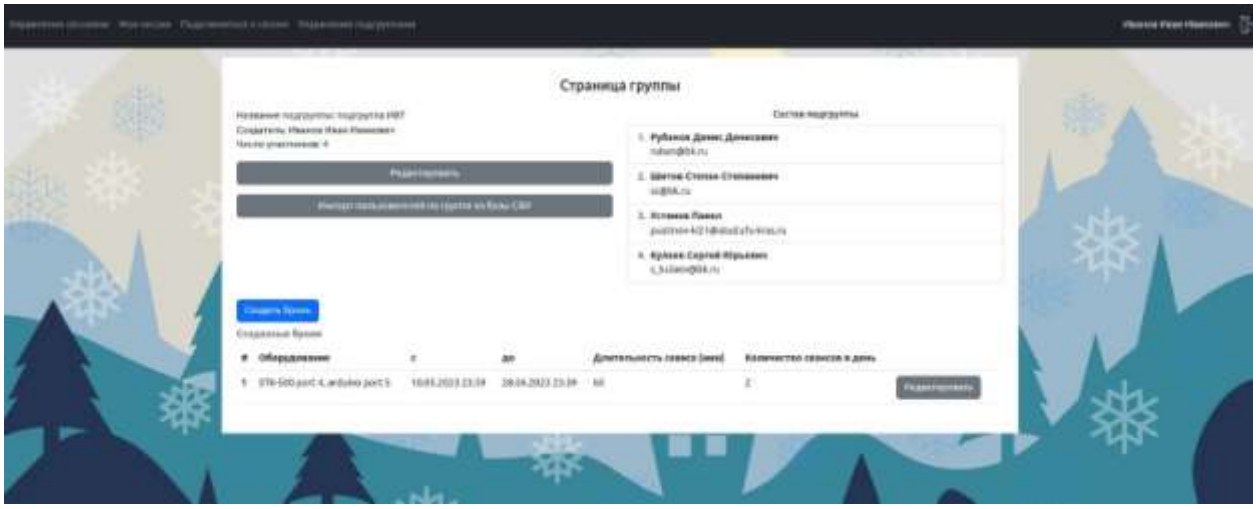

Рисунок 15 - Страница подгруппы для роли преподаватель

Если преподаватель нажмет на кнопку «Редактировать» - откроется модальное окно, показанное на рисунке 16. В данном модальном окне имеется возможность удалять участников и менять название подгруппы. Помимо этого, здесь доступно добавление участников, поиск которых осуществляется в БД системы.

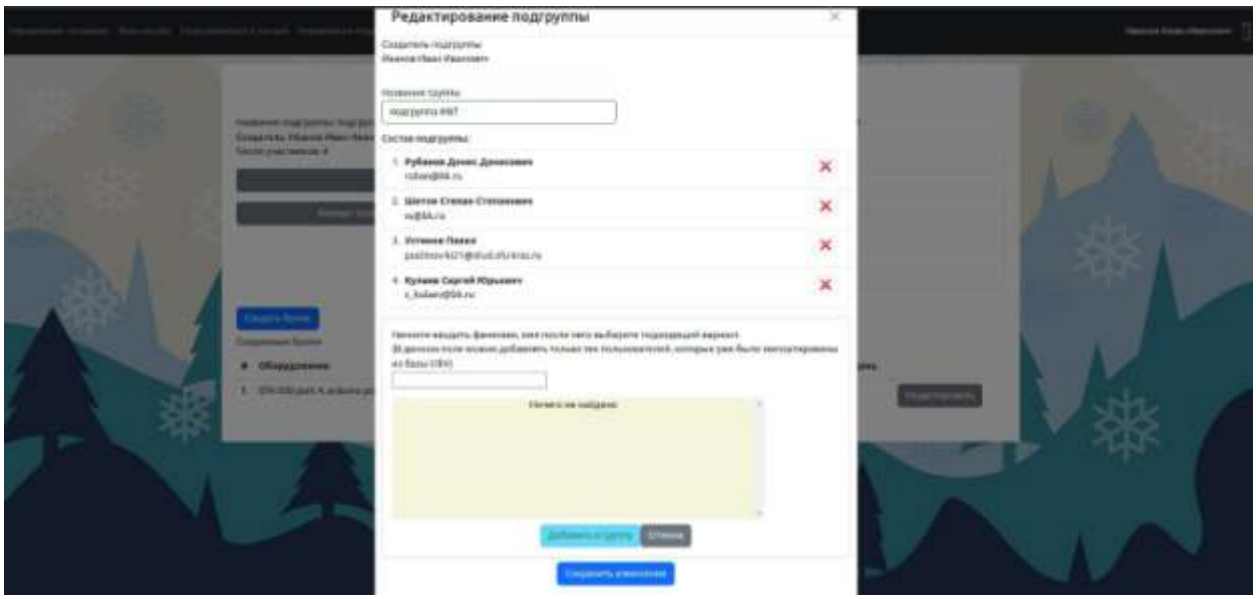

Рисунок 16 - Модальное окно редактирования подгруппы

Добавление участников в подгруппу возможно по учебной группе. Для этого пользователю нужно нажать на кнопку «Импорт пользователей по группе из базы данных СФУ». При нажатии появится модальное окно, в котором нужно ввести название группы на английском языке. При правильном вводе группы, появится список всех студентов этой группы, где можно выбрать одного или нескольких студентов из списка, и добавить их в подгруппу.

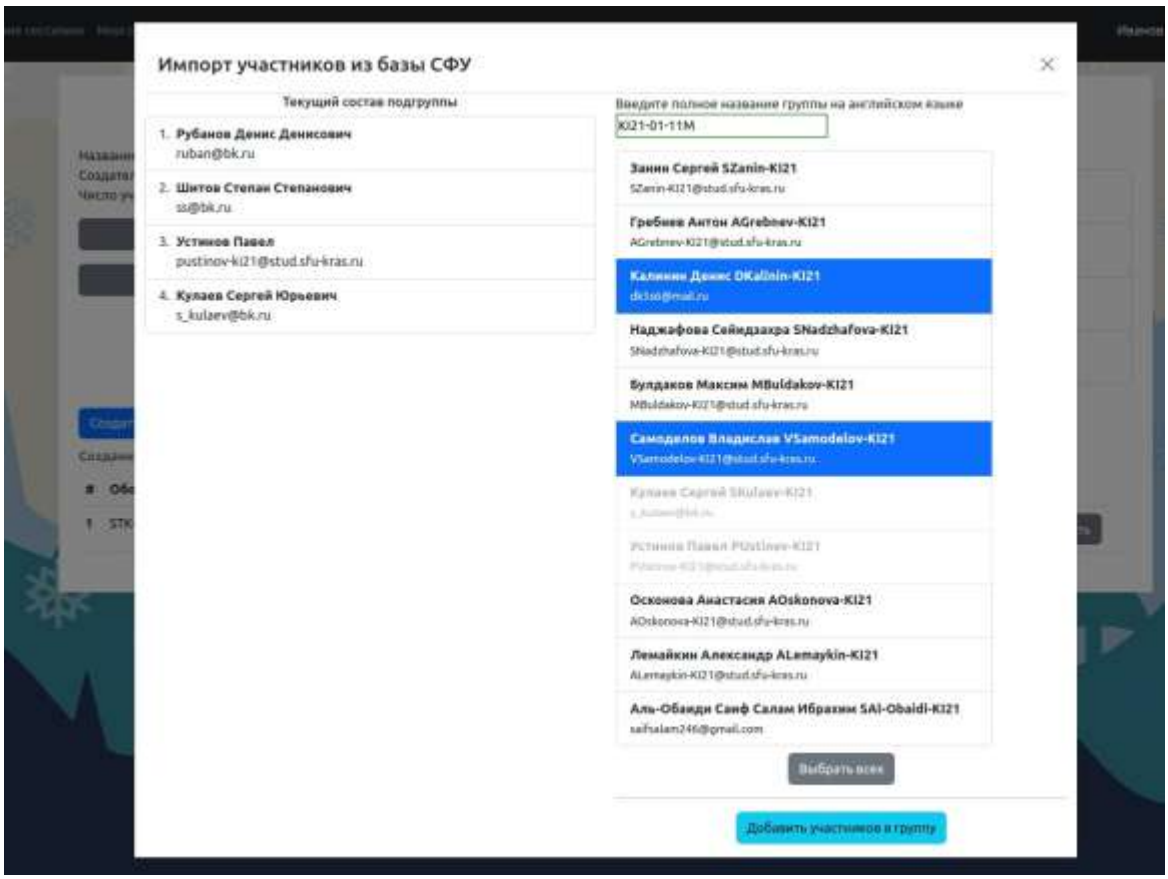

Рисунок 17 - Модальное окно добавления участников в подгруппу по названию учебной группы

На рисунке 18 показано модальное окно с формой для создания и редактирования брони. Здесь можно выбрать:

- временной интервал, в рамках которого участники подгруппы будут задавать время проведения сессии;
- оборудование;
- длительность создаваемых пользователями сессий;
- максимальное количество сессий в день.

После задания всех параметров и нажатия кнопки «сохранить» бронь сохраняется и появляется в списке броней подгруппы.

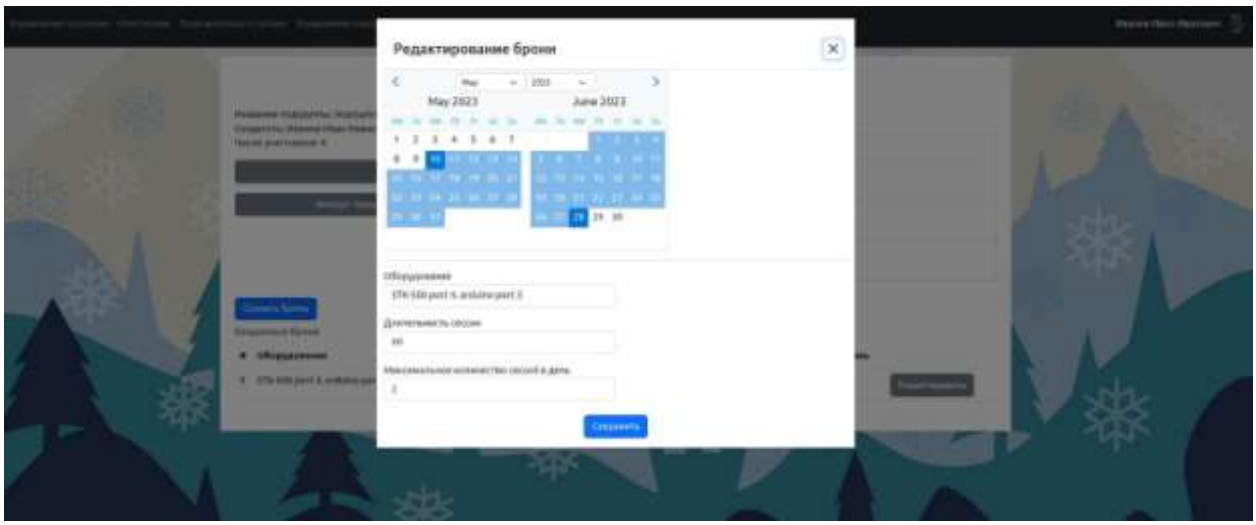

Рисунок 18 – Модальное окно редактирования брони

Если попытаться зайти на страницу подгруппы под пользователем с ролью студент, то будет отображена та же страница, что и для преподавателя, однако все кнопки, доступные преподавателю. на странице будут скрыты, кроме одной, которая доступна только студенту, а именно - «Записаться на сессию». Страница подгруппы для пользователя с ролью студент представлена на рисунке 19.

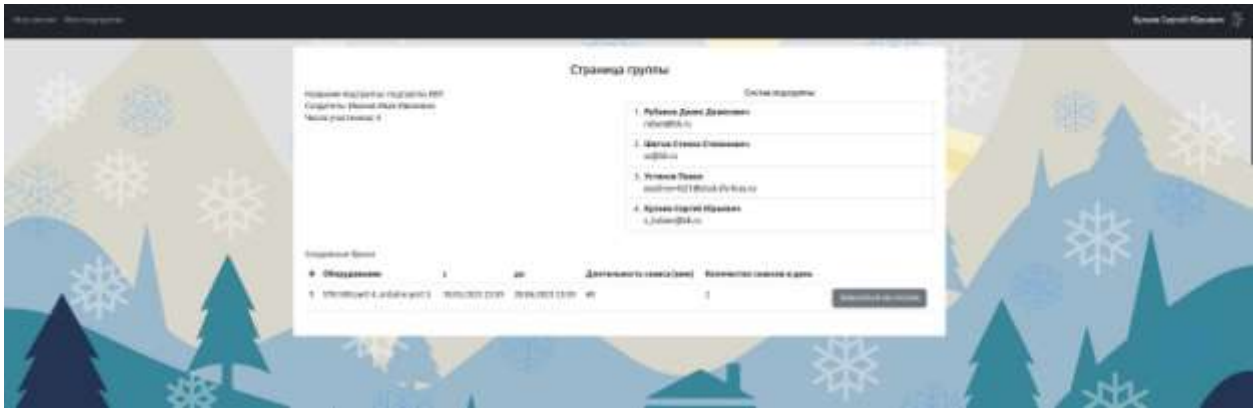

Рисунок 19 – Страница подгруппы для роли студент

Если в списке броней студент нажмет на кнопку «Записаться на сессию» - откроется модальное окно с формой показанное на рисунке 20. В форме студенту достаточно выбрать дату и время начала сессии, а время окончания сессии будет рассчитано автоматически. Также в форме

присутствуют информационные поля, среди которых: оборудование, длительность сессии, максимальное количество сессий в день, оставшееся количество сессии на выбранную дату.

Дополнительно для того, чтобы пользователь смог правильно выбрать дату и время начала сессии, в модальном окне присутствует список всех запланированных сессий на указанную дату.

После ввода корректных данных, пользователю станет доступна кнопка «Создать», после нажатия по которой сессия будет сохранена и отображена на странице «Мои сессии».

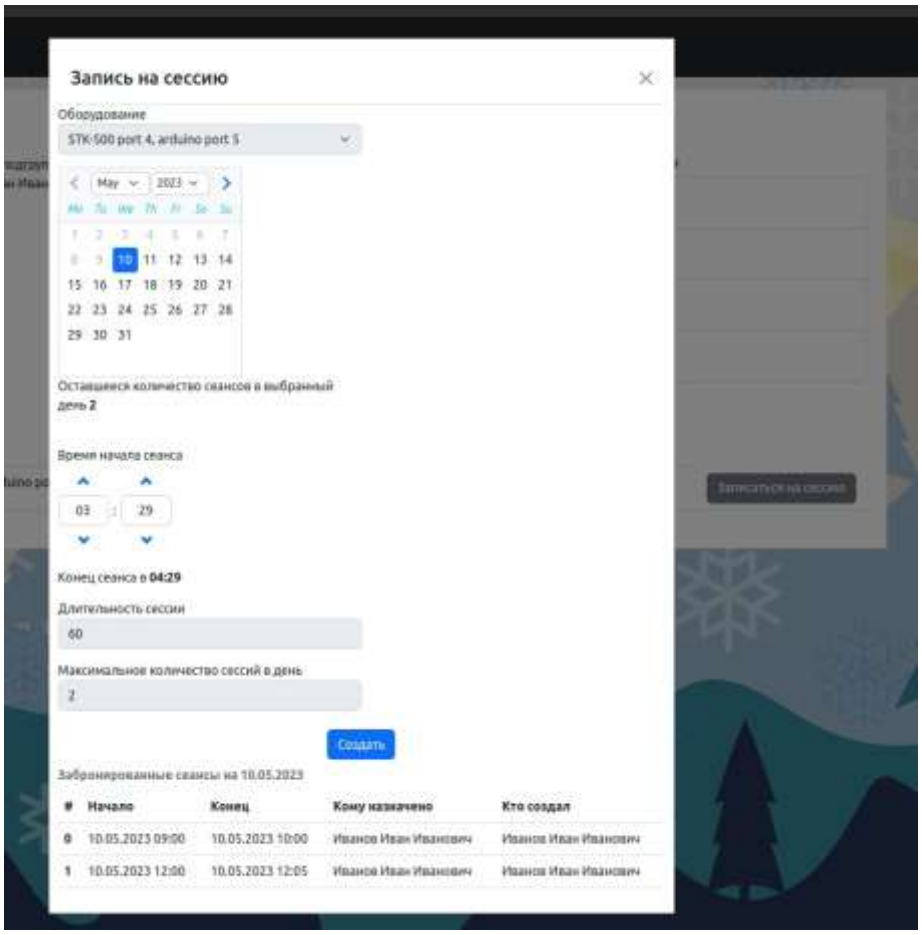

Рисунок 20 – Модальное окно создания сессии с помощью брони в подгруппе

4.3 Демонстрация и тестирование работы с платой в рамках сессии

Страница сессии состоит из следующих элементов:

- видеотрансляция, показывающая плату с которой ведется работа;
- таймер, ведущий обратный отчет до окончания сессии, по истечении которого, пользователь будет перенаправлен на страницу «Мои сессии»;
- консоль с журналом вывода действий с платой;
- кнопка «Очистить оборудование», которая удаляет с сервера последний загруженный файл, и удаляет текущую прошивку с платы;
- кнопка «Прикрепить код программы». При нажатии по кнопке, появляется окно для выбора файла, содержащего скомпилированный код программы;
- кнопка «прошить оборудование выбранным файлом». При нажатии по кнопке, прикрепленный файл с программой загружается на сервер управления оборудованием, после чего программа прошивается в нужную плату;
- кнопка «Перепрошить оборудование уже загруженным файлом». Кнопка прошивает плату программой из последнего загруженного на сервер файла;
- информационные блоки, в которых есть: время начала сессии, время окончания сессии, ФИО пользователя для которого сессия была создана, название оборудования с которым ведется работа;
- набор элементов управления, соответствующий типу платы, с которой ведется работа. В набор могут входить кнопки, переключатели или переменный резистор.

Демонстрация сессии с платой STK-500 представлена на рисунке 21. На странице присутствует 8 кнопок и переменный резистор. В плату была прошита программа из файла Test_LEDs.hex, с кодом для работы со светодиодами. При нажатии кнопок и изменении значения резистора на интерфейсе светодиоды меняют свое значение.

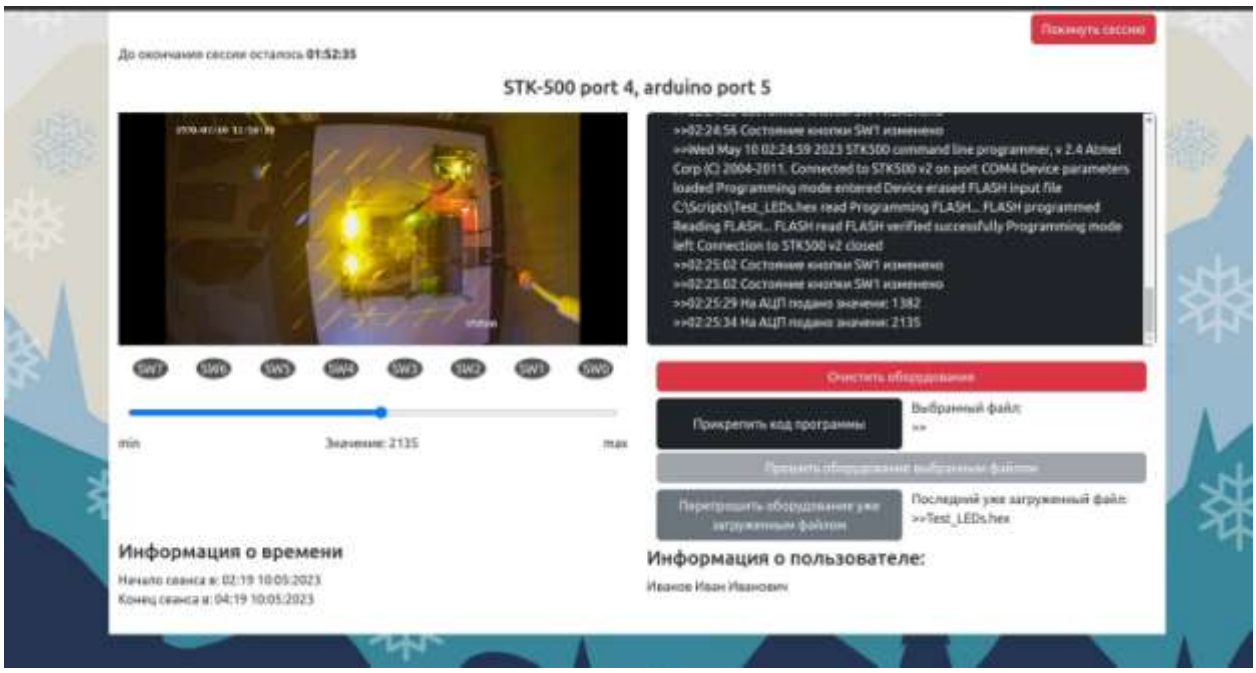

Рисунок 21 – страница сессии с платой STK-500

На рисунке 22 представлена страница сессии с платой Altera DE1-SoC. На странице можно наблюдать 8 переключателей и 4 кнопки.

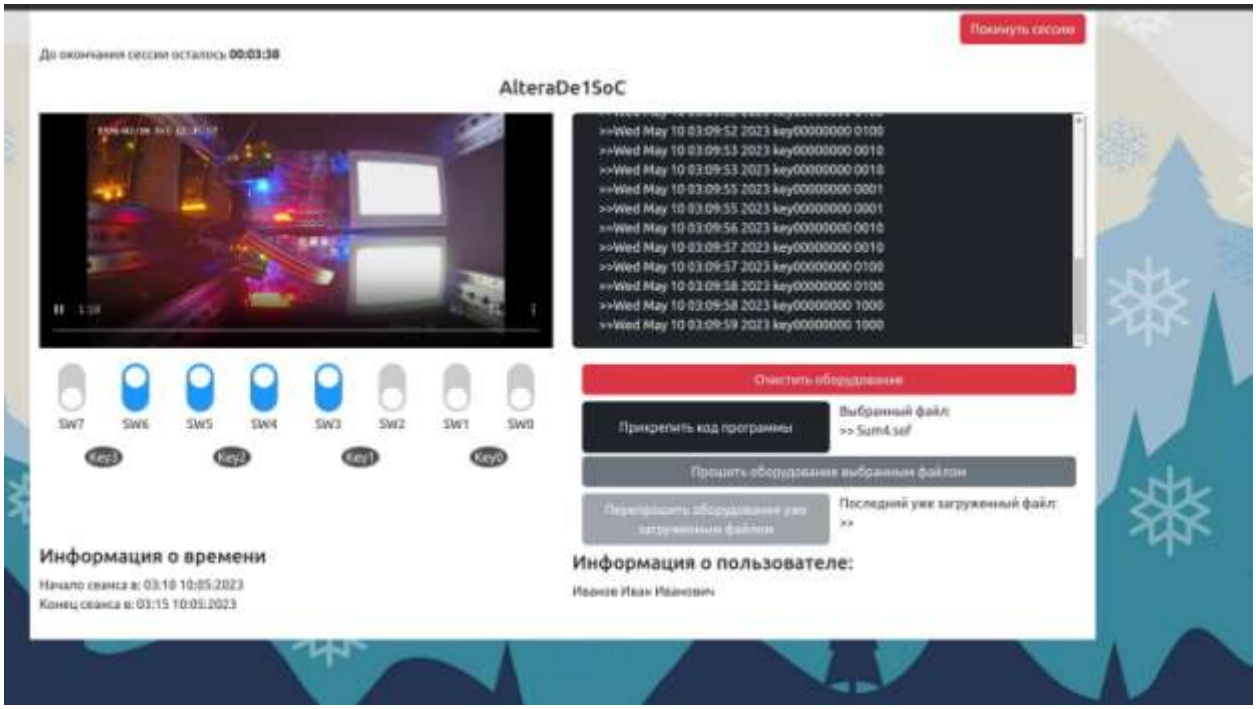

Рисунок 22 - страница сессии с платой Altera DE1-SoC

Для выяснения задержки выполнения команд измерялись 3 метрики на 10 командах. Первая метрика отображает длительность выполнения скриптов на сервере управления оборудованием, то есть аппаратную задержку выполнения команды. Вторая метрика отображает общее время выполнения команды на клиенте. Третья метрика показывает сетевую задержку. Для замеров использовалась плата STK-500. Результаты измерений приведены в таблице 1.

Таблица 1 – Длительность выполнения команд

Общая длительность выполнения команды, мс	**Аппаратная длительность выполнения скрипта, мс**	**Сетевая задержка, мс**
486	387	99
330	280	50
290	245	45
319	273	46
207	164	43
218	182	36
190	150	40
236	190	46
220	180	40

По результатам измерений можно сделать следующие выводы:

- самая первая команда выполняется дольше всего по всем 3 метрикам;
- наибольшая составляющая задержки выполнения команды приходится на аппаратное выполнение скрипта;
- среднее время выполнения команд 249,6 мс;
- средняя сетевая задержка составляет всего 44,5 мс.

4.4 Выводы по главе

Была проведена демонстрация и тестирование пользовательского интерфейса, а также всей системы в целом. Все компоненты системы взаимодействуют без ошибок. Разработанная архитектура позволила легко заменять и добавлять новые платы в лабораторные стенды.

ЗАКЛЮЧЕНИЕ

В результате выполнения проекта были решены следующие задачи:

1) изучена предметная область, проанализированы аналоги разрабатываемой системы;

2) сформулированы функциональные требования к системе;

3) предложена масштабируемая архитектура системы удаленного доступа, которая позволяет легко добавлять и изменять лабораторные стенды;

4) разработаны принципы планирования сессий с оборудованием;

5) предложен и смоделирован способ организации взаимодействия компонентов системы при работе с оборудованием;

6) спроектированы и реализованы такие компоненты системы как SPA-приложение, сервер управления данными, сервер посредник, сервер управления оборудованием;

7) проведена развертка разработанной системы на сервере ИКИТ СФУ;

8) проведено тестирование всех компонентов системы.

9) результаты тестирования показали нормальное функционирование всех частей разработанной системы. Однако есть недочеты в виде средней аппаратной задержки выполнения скриптов в размере 249,6 мс.

Таким образом, все поставленные в рамках ВРК задачи успешно решены, цель магистерской диссертации достигнута. Полученные технические решения могут быть использованы для создания аналогичных систем удаленного доступа. Использование разработанных решений позволит им добиться низкой сетевой задержки и масштабируемости.

СПИСОК СОКРАЩЕНИЙ

БД – база данных

ВКР – выпускная квалификационная работа

ПК – персональный компьютер

ПО – программное обеспечение

CSS – Cascading Style Sheets

CMS – система управления содержимым

SPA – одностраничное клиентское приложение

Список использованных источников

1. Евдокимов, Ю. К. Дистанционные автоматизированные учебные лаборатории и технологии дистанционного учебного эксперимента в техническом вузе / Ю.К.Евдокимов, А.Ю.Кирсанов, А.Ш.Салахова // Открытое образование. – 2009. – № 5. – С. 101–116.
2. Ёхин М. Н. Организация многопользовательского удаленного доступа к распределенной гетерогенной системе лабораторного оборудования на основе схем программируемой логики для дистанционных практикумов по цифровой схемотехнике / М. Н. Ёхин, М. М. Степанов // Современные информационные технологии и ИТ-образование. – 2017. – Т. 13. – №. 4. – С. 65-81.
3. Колесников Д. А. Создание автоматизированной лабораторной установки с удаленным доступом на базе платформы Arduino: магистерская диссертация по направлению подготовки: 03.04.02 Физика / Д. А. Колесников ; Томский государственный университет. - Томск, 2020. - 51 с.
4. Латыпова В. А. Методики проведения и проверки лабораторных работ при смешанном и дистанционном автоматизированном обучении / В. А. Латыпова // Инженерный вестник Дона. – 2015. – Т. 37. – №. 3. – С. 35.
5. Мангушев А. В. РЕАЛИЗАЦИЯ MQTT КЛИЕНТА НА БАЗЕ ОДНОКРИСТАЛЬНОГО МИКРОКОНТРОЛЛЕРА ДЛЯ ЗАДАЧ УДАЛЕННОГО УПРАВЛЕНИЯ ОБОРУДОВАНИЕМ / А. В. Мангушев // Известия Южного федерального университета. Технические науки. – 2022. – №. 3 (227). – С. 75-84.
6. Менщиков А. О. Применение технологий LabVIEW для разработки лабораторных практикумов удаленного доступа по электротехнике / А. О. Менщиков, О. А. Доценко; Томский государственный университет. - Томск, 2018. - С. 275-277.

7. Новиков А. В. Лабораторный учебный комплекс с возможностью удаленного доступа через Интернет / А. В. Новиков, Е. С. Повернов, Е. В. Сыпин // Образовательные технологии. – 2006. – №. 2. – С. 94-101.

8. Сницарук Д. Г. Лабораторный стенд для исследования цифровых систем обработки информации на базе микроконтроллера STM32 / Д. Г. Сницарук, В. В. Коняшов // Энерго- и ресурсосбережение: промышленность и транспорт. – 2019. – №. 2. – С. 44-49.

9. Alkhaldi. T. A review of contemporary virtual and remote laboratory implementations: observations and findings / T. Alkhaldi, I. Pranata, R. I. Athauda // Journal of Computers in Education. – 2016. – № 3. – С. 329–351.

10. Bermúdez-Ortega J. Remote web-based control laboratory for mobile devices based on EJsS, Raspberry Pi and Node. js / J. Bermúdez-Ortega, E. Besada-Portas, J. A. López-Orozco, J. A. Bonache-Seco, J. M. de la Cruz // IFAC-PapersOnLine. – 2015. – Т. 48. – №. 29. – С. 158-163.

11. Besada-Portas E. Lightweight node. js & ejss-based web server for remote control laboratories / E. Besada-Portas, J. Bermúdez-Ortega, L. de la Torre, J. A. López-Orozco, J. M. de la Cruz // IFAC-PapersOnLine. – 2016. – Т. 49. – №. 6. – С. 127-132.

12. Chacon J. Ejs, jil server, and labview: An architecture for rapid development of remote labs / J. Chason, H. Vargas, G. Farias, S. Dormido // IEEE Transactions on Learning Technologies. – 2015. – Т. 8. – №. 4. – С. 393-401.

13. Chand P. A Low-Cost System for Remote Access and Control of Automation Equipment / P. Chand, M. Al-Rawi, S. James, J. Antony, J. Jose // Machines. – 2021. – Т. 9, № 7. – С. 138.

14. Guimaraes E. G. Design and implementation issues for modern remote laboratories / E. G. Guimarães, E. Cardozo, D. H. Moraes, P. R. Coelho // IEEE transactions on learning technologies. – 2010. – Т. 4. – №. 2. – С. 149-161.

15. Heradio R. Virtual and remote labs in education: A bibliometric analysis / R. Heradio, L. de la Torre, D. Galan, F. J. Cabrerizo, E. Herrera-Viedma, S. Dormido // Computers & Education. – 2016. – Т. 98. – С. 14-38.

16. Jacko P. Remote IoT Education Laboratory for Microcontrollers Based on the STM32 Chips / P. Jacko, M. Bereš, I. Kováˇcová, J. Molnár, T. Vince, J. Dziak, B. Fecko, Š. Gans, D. Kováˇc // Sensors. – 2022. – Т. 22, № 4. – С. 1440.

17. Lavayssière C. Laborem Box: A scalable and open source platform to design remote lab experiments in electronics / C. Lavayssière, B. Larroque, F. Luthon // HardwareX. – 2022. – Т. 11. – С. e00301.

18. Lewis J. Fowler M. Microservices: a definition of this new architectural term // MartinFowler. com. – 2014. – Т. 25. – №. 14-26. – С. 12. – URL: https://martinfowler.com/articles/microservices.html (дата обращения: 12.05.2023).

19. Li W. J. JustIoT Internet of Things based on the Firebase real-time database / W-J. Li, C. Yen, Y-S. Lin, S-C. Tung, S. Huang // 2018 IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE). – IEEE, 2018. – С. 43-47.

20. Melosik M. Remote Prototyping of FPGA-Based Devices in the IoT Concept during the COVID-19 Pandemic / M. Melosik, M. Naumowicz, M. Kropidlowski, W. Marszaiek // Electronics. – 2022. – Т. 11. – №. 9. – С. 1497.

21. Mohammed A. K. Remote controlled laboratory experiments for engineering education in the post-COVID-19 era: Concept and example / A. K. Mohammed, H. M. El Zoghby, M. M. Elmesalawy // 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES). – IEEE. - 2020. – С. 629-634.

22. Mohan M. S. Remote laboratory / M. S. Mohan, P. Karthikeyan, D. Ram Kumar, M. Rupesh // International Journal of Engineering and Advanced Technology (IJEAT). – 2019. – Т. 8. – №. 6S. – С. 554-559.

23. Monzo C. Remote laboratory for online engineering education: The rlab-uoc-fpga case study / C. Monzo, G. Cobo, J. A. Moran, E. Santamaria, D. Garcia-Solorzano // Electronics. – 2021. – Т. 10. – №. 9. – С. 1072.

24. Orduña P. An extensible architecture for the integration of remote and virtual laboratories in public learning tools / P. Orduña, D. G. Zutin, S. Govaerts, I. Lequerica, P. H. Bailey, E. Sancristobal, C. Salzmann, L. Rodriguez-Gil, K. DeLong, D. Gillet, M. Castro, D. Lópezde-Ipiña, J. Garcia-Zubia // IEEE Iberoamericana de Tecnologias del Aprendizaje. – 2015. – Т. 10. – №. 4. – С. 223-233.

25. Reid D. P. Open-source remote laboratory experiments for controls engineering education / D. P. Reid, J. Burridge, D. B. Lowe, T. D. Drysdale // International Journal of Mechanical Engineering Education. – 2022. – Т. 50. – №. 4. – С. 828-848.

26. Sáenz J. Open and low-cost virtual and remote labs on control engineering / J. Sáenz, J. Chacón, L. De La Torre, A. Visioli, S. Dormido // Ieee Access. – 2015. – Т. 3. – С. 805-814.

27. Soundra Pandian K. K. Remote-access real-time laboratory: process monitoring and control through the internet protocol / K. K. Soundra Pandian, M. Rao, S. Khandekar // International Journal of Mechanical Engineering Education. – 2008. – Т. 36. – №. 3. – С. 207-220.

28. Tho S. W. Technology-enhanced science learning through remote laboratory: System design and pilot implementation in tertiary education / S. W. Tho, Y. Y. Yeung // Australasian Journal of Educational Technology. – 2016. – Т. 32. – №. 3.

29. Tripathi P. K. Design and implementation of web based remote laboratory for engineering education / P. K. Tripathi, J. Mohan, K. V. Gangadharan // International Journal of Engineering and Technology. – 2012. – Т. 2. – №. 2. – С. 270-278.

30. Weiszflog M. Transforming laboratory experiments for digital teaching: remote access laboratories in thermodynamics / M. Weiszflog, I. K. Goetz // European Journal of Physics. – 2021. – Т. 43. – №. 1. – С. 015701.

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
«22» 06 2023 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

«Система удаленного доступа к лабораторному оборудованию»
тема

09.04.01 «Информатика и вычислительная техника»
код и наименование направления

09.04.01.11 «Вычислительные системы и сети»
код и наименование магистерской программы

Научный руководитель	, 22.06.2023 подпись, дата	доцент каф.ВТ, канд. тех. наук должность, ученая степень	Титовский С.Н. инициалы, фамилия
Выпускник	, 22.06.2023 подпись, дата		Кулаев С.Ю. инициалы, фамилия
Рецензент	, 22.06.2023 подпись, дата	доцент каф. СИИ ИКИТ СФУ, канд. физ.-мат. наук должность, ученая степень	Коченов Д.А. инициалы, фамилия
Нормоконтролер	, 22.06.2023 подпись, дата	доцент каф.ВТ, канд. тех. наук должность, ученая степень	Титовский С.Н. инициалы, фамилия

Красноярск 2023