

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

Вычислительная техника

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Профиль «Вычислительные машины, комплексы, системы
и сети»

БАКАЛАВРСКАЯ РАБОТА

на тему

**«ИНФОРМАЦИОННАЯ СИСТЕМА
ВЫПОЛНЕНИЯ ПРАКТИЧЕСКИХ ЗАДАНИЙ»**

Студент Антропов Д.В. 26.05.25
(подпись, дата)

Руководитель Юрова О.В. 30.05.25
(подпись, дата)

Нормоконтролёр Бычков А.С. 22.05.25
(подпись, дата)

Работа допущена к защите (протокол заседания кафедры от

28.05.2025 г. № 16)

Заведующий кафедрой Митрохин М.А.
(подпись)

Работа защищена с отметкой отлично (протокол заседания ГЭК от 28.06.25 № 2)

Секретарь ГЭК Гурин Е.И.
(подпись)

Пенза, 2025

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
«6» 05 2025 г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Антропов Данила Вадимович гр. 21BBB1 факультета вычислительной техники.

2. Тема работы: «Информационная система выполнения практических заданий»

Тема утверждена приказом ПГУ № 601/0 от "6" 05 2025 г.

3. Руководитель бакалаврской работы к.т.н., доцент каф. ВТ ПГУ Юрова О.В.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

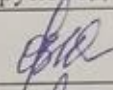
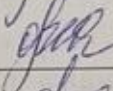
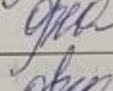
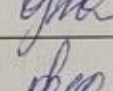
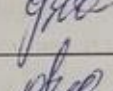
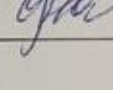
Необходимо создать программу информационной системы выполнения практических и проектных заданий, ориентированной на использование в образовательных учреждениях, корпоративных обучающих системах и рабочих коллективах. Программа должна реализовывать механизмы управления расписанием занятий и выполнения заданий в интегрированной среде.

Программа должна реализовывать систему ролевого доступа к системе и различный функционал в зависимости от роли, а также модуль оценки результатов, статуса выполнения и сроками выполнения заданий.

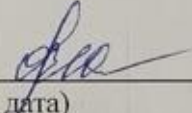
5. Перечень подлежащих разработке вопросов

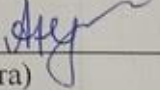
Требуется произвести анализ следующих предметных областей: изучить существующие методы и алгоритмы организации образовательных и рабочих процессов в режиме онлайн, изучить современные интеллектуальные алгоритмы, применимые в планировании занятий и мероприятий, изучить существующие среды для организации практической, лабораторной и проектной деятельности и описать наиболее перспективные и удачные из них. Также необходимо изучить функционал платформ визуального программирования и выбрать самую подходящую под реализацию разрабатываемой системы. Необходимо рассмотреть вопросы создания архитектуры программы, реализации пользовательского интерфейса в выбранных языках программирования, а также изучить возможности интеграции системы с различными сторонними сервисами для расширения функционала приложения.

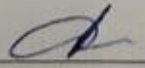
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Изучение предметной области	52	06.05.2025-09.05.2025	
2.	Анализ требований	13	10.05.2025	
3.	Выбор технологий, применяемых при проектировании и разработке	52	11.05.2025-14.05.2025	
4.	Описание алгоритмов	39	15.05.2025-17.05.2025	
5.	Разработка программы выполнения практических заданий	104	18.05.2025 - 25.05.2025	
6.	Оформление пояснительной записки	64	26.05.2025 - 30.05.2025	

Дата выдачи задания " 6 " мая 2025 г.

Руководитель бакалаврской работы 30.05.25  Юрова О.В.
(подпись, дата)

Задание к исполнению принял студент 26.05.25  Антропов Д.В.
(подпись, дата)

Работу к защите допустить
Декан факультета 30.05.2025  Фионова Л.Р.
(подпись, дата)

Реферат

Бакалаврская работа содержит 90 листов, 34 рисунка, 3 таблицы, 35 источников и 3 приложения.

ВЕБ-ПРИЛОЖЕНИЕ, NODE-RED, ВИЗУАЛЬНОЕ ПРОГРАММИРОВАНИЕ, БАЗА ДАННЫХ, КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ

Цель работы заключается в разработке программы, которая реализует возможность выполнения практических и проектных заданий и ориентирована на использование в образовательных учреждениях, а также визуализирует предстоящие занятия в удобном для обучающихся или сотрудников формате.

Объектом исследования в работе являются системы организации дистанционного обучения и выполнения заданий.

В работе применены методы разработки веб-приложений с использованием HTML и JavaScript, а также программная реализация с использованием платформы визуального программирования Node-Red.

В работе рассмотрены существующие системы организации образовательных и рабочих процессов в режиме онлайн, проведён обзор современных платформ визуального программирования, протестированы различные среды разработки и готовые решения систем, а также определен наилучших свойств, необходимых подобной системе, для применения в разрабатываемой программе.

					ПГУ 1.090301.03.001 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Информационная система выполнения практических заданий. Пояснительная записка	Лит.	Лист	Листов
Разраб.		Антропов Д.В.		26.05				
Провер.		Юрова О.В.		30.05			2	90
Ил. Контр		Бычков А.С.		24.05				
Утверд.		Митрохин М.А.		30.05				
						ФВТ, каф. ВТ гр. 21ВВВ1		

Содержание

Перечень сокращений и обозначений	4
Введение	5
1 Основная часть	6
1.1 Анализ предметной области	6
1.2 Постановка задачи	12
1.3 Выбор инструментов разработки	14
2 Разработка программы выполнения практических заданий	25
2.1 Архитектура приложения	25
2.2 Проектирование и разработка	28
2.2.1 Настройка среды разработки	28
2.2.2 Описание разработанной программы	34
2.3 Тестирование и анализ результатов	42
3 Охрана труда	49
Заключение	55
Список использованных источников	56
Приложение А – UML – диаграммы программы	60
Приложение Б – Листинг программы	66
Приложение В – Презентация	87

Перечень сокращений и обозначений

ПЭВМ – персональная электронно-вычислительная машина

API (Application Programming Interface) – программный интерфейс приложения

IoT (Internet of Things) – концепция объединения физических устройств с доступом к интернету, которые работают с данными и обмениваются ими между собой без участия человека

JSON (JavaScript Object Notation) – текстовый формат файла для хранения и передачи данных

Moodle (Modular Object-Oriented Dynamic Learning Environment) – модульная объектно-ориентированная динамическая образовательная среда

SQL (Structured Query Language) – язык структурируемых запросов для работы с базой данных

					ПГУ 1.090301.03.001 ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

В условиях настоящего времени имеет место стремительное развитие информационных технологий и повсеместная цифровизация образовательных, творческих и производственных процессов, а также не перестают повышаться требования к организации этих процессов в различных формах (очно, дистанционно, смешанно). Это приводит к возрастанию необходимости в многофункциональных информационных системах, которые смогут и должны будут обеспечивать не только поддержку теоретического обучения, но и эффективную организацию практической и проектной деятельности учащихся и функционирующих сотрудников различных компаний.

Особую актуальность приобретают такие системы в вузах, колледжах, онлайн-школах, а также на корпоративных обучающих и рабочих платформах, где есть необходимость в отслеживании и контроле выполнения практических заданий, проектных работ, тестов, а также взаимодействии между обучающимися и преподавателями или руководителями и сотрудниками. Вопреки мнению, которое складывается при первом взгляде или просмотре функционала, подобные системы могут использоваться не только в образовательных учреждениях, но и на закрытых предприятиях для обучения персонала, организации внутренних курсов повышения квалификации, а также координации совместной работы над проектами или общей организации рабочего процесса нескольких сотрудников, работающих над одной задачей. В таких условиях может требоваться возможность развёртывания системы в изолированной локальной вычислительной сети.

Одним из современных решений этой задачи может стать веб-ресурс, предназначенный для выполнения практических и проектных заданий, с возможностью планирования занятий, добавления лабораторных и тестовых заданий и их выполнения в интерактивной среде. Также в необходимых функциональных возможностях этой системы могут быть автоматическая или частично-автоматическая проверка и оценка результатов, а также масштабируемость и интеграция в существующие учебные и производственные процессы.

					ПГУ 1.090301.03.001 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

1 Основная часть

1.1 Анализ предметной области

Современное образование и проектная деятельность, как в учебных заведениях, так и в корпоративной среде, требуют высокоорганизованных информационных систем, способных обеспечить:

- эффективное планирование занятий и мероприятий;
- централизованное управление заданиями и проектами;
- возможность выполнения заданий онлайн, в том числе с использованием интерактивных вычислительных сред;
- мониторинг успеваемости и активности участников;
- удобный обратный контроль и оценку заданий со стороны преподавателей или руководителей проектов;
- поддержка ролевой модели пользователей: администраторы, преподаватели, студенты (или сотрудники).

Информационная система, разрабатываемая в данной ВКР, решает именно эти задачи, предоставляя веб-ресурс для организации практической, лабораторной и проектной деятельности.

Существует несколько программных продуктов, реализующих схожие задачи и функциональные возможности:

Moodle (Modular Object-Oriented Dynamic Learning Environment) – система управления образовательными электронными курсами [1]. Представляет собой свободное веб-приложение, представляющее возможность создавать сайты для онлайн-обучения. Moodle позволяет:

- создавать онлайн-курсы посредством объединения текстовых документов, презентаций и видео в единый обучающий курс, который станет доступным для всех учащихся или конкретной группы;
- создавать тесты с помощью встроенного редактора и осуществлять их автоматическую проверку;
- общение между учащимися и преподавателями через комментарии к

курсам и встроенный форум.

К преимуществам этой системы относятся:

- бесплатность и открытый код;
- поддержка различных форматов обучения;
- большая база документации.

Недостатками являются:

- сложность в освоении;
- требовательность к серверу;
- ограниченный дизайн и устаревший интерфейс;
- проигрыш в простоте развёртывания по сравнению с облачными решениями;
- отсутствие возможности запуска кода.

На рисунке 1 показан демонстрационный сайт Moodle.

Moodle Demonstration Site

Вы не прошли идентификацию ([Вход](#))
Русский (ru)

Moodle Demonstration Site

Welcome! This site is for you to try using the most recent stable released version of Moodle and to explore some courses which demonstrate a few of Moodle's many features.

Here are some demo accounts for you to use:

- Admin - username *admin*, password *FunMoodling!*
- Teacher - username *teacher*, password *FunMoodling!*
- Student - username *student*, password *FunMoodling!*

The database and files are erased and restored to a clean state every hour on the hour, so don't worry if you make a mess.

Also note that you might not be the only person using one of the demo accounts at the same time, so you may see unexpected things happen occasionally.

Категории курсов

- Arabic**
الأنشطة
- Bulgarian**
Moodle Demo
- Dutch**
ICT en samenleving
- English**
Exchange Samples
Higher Education Film Studies Module
Moodle Features Demo
Romeo and Juliet Sample Unit
Understanding Moodle

Вход

Логин
Пароль

[Создать учетную запись](#)
[Забыли пароль?](#)

Пользователи на сайте
(последние 5 минут)
Sam Student
Гость
Oscar Javier Bachiller Sandoval

Demo courses needed!
Would you like to donate a demo course for use on this site, or translate the [Moodle Features Demo](#) course into your language? If so, see [Moodle demo courses](#).

Moodle Announcements

moodle
Moodle 1.9.8 and Moodle 1.8.12 have been released
Moodle books available in various languages
Teaching and Learning with Moodle
Moodle in healthcare
Cool Course Competition prizewinners

Рисунок 1 – Демонстрационный сайт Moodle

Google Classroom – бесплатный веб-сервис, разработанный Google, призванный упростить создание, распространение и оценку заданий безбумажным способом [2]. К основным функциям [3] платформы относятся:

- создание и управление классами и курсами;
- организация записи учащихся на курс;
- предоставление доступа к учебным материалам;
- оценка заданий и организация взаимодействия учащихся.

Преимуществами Google Classroom являются:

- интуитивно понятный интерфейс;
- интеграция с другими сервисами Google;
- бесплатность для образовательных учреждений;
- быстрая проверка заданий и выставление оценок;
- возможность планирования публикаций материалов и заданий;
- редактирование документов и комментирование работ;
- проведение онлайн-уроков;
- лёгкий обмен файлами.

Недостатки [4] Google Classroom:

- отсутствие инструментов для создания интерактивных курсов и встроенного тестирования;
- неработоспособность без интернета;
- нет возможности гибкой настройки;
- хранение информации на серверах Google, что может не подходить некоторым учреждениям;
- необходимость наличия аккаунта Google;
- отсутствие возможности запуска кода.

На рисунке 2 показан внешний вид Google Classroom.

Stepik – российская образовательная платформа и конструктор бесплатных и платных открытых онлайн курсов и уроков [5]. Создана в 2013 году при поддержке JetBrains и лаборатории алгоритмической биологии Санкт-Петербургского академического университета. Особенности платформы [6] являются:

- широкий выбор курсов по разным направлениям;

- интерактивный формат обучения, включающий в себя не только теоретические материалы, но и интерактивные задания, тесты и симуляции;
- гибкий график обучения с возможностью выбора самостоятельного изучения курсов в удобное время либо занятия с преподавателями в режиме реального времени;
- возможность создания своих курсов.

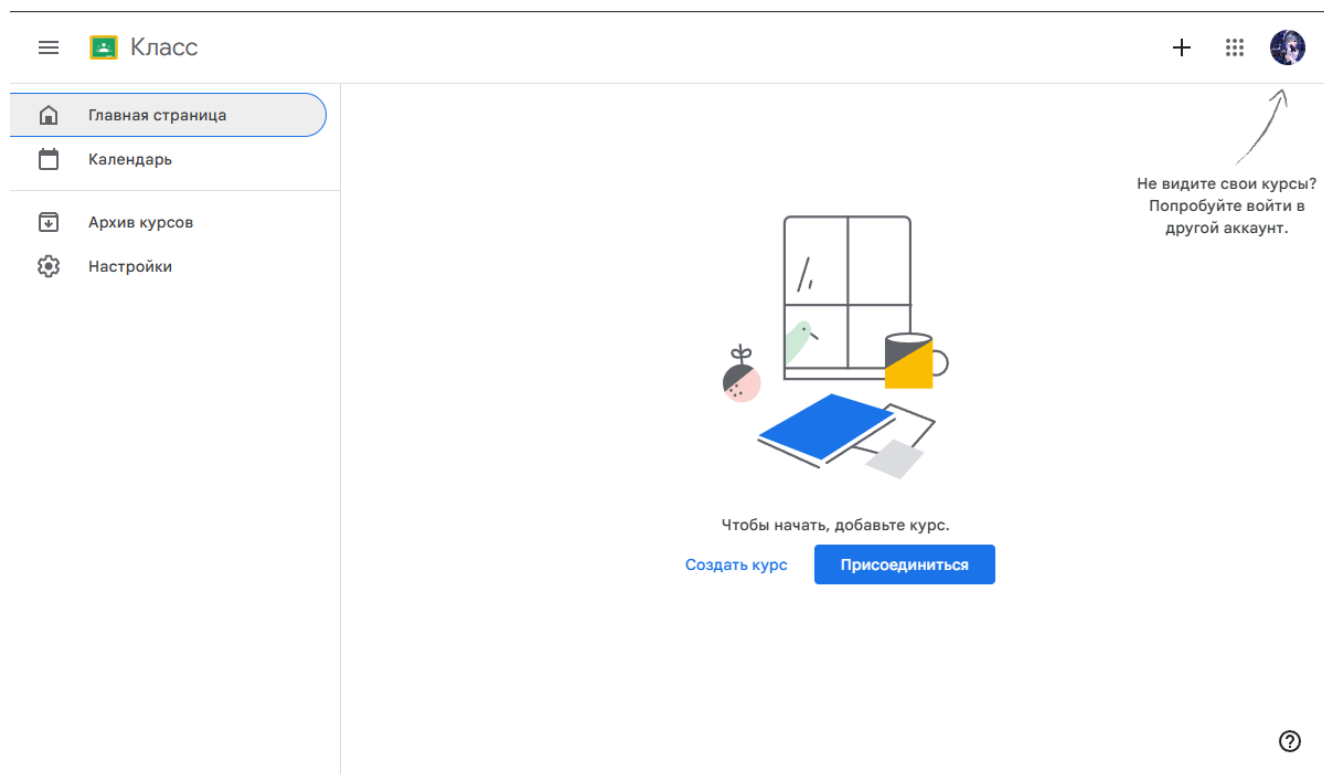


Рисунок 2 – Внешний вид Google Classroom

К преимуществам [7] Stepik относятся:

- специализация на онлайн курсах и программировании;
- возможность создания задач с автоматической проверкой и поддержка различных форматов вопросов;
- множество бесплатных курсов и открытость для всех желающих;
- возможность создания своих курсов и инструменты для командной работы;
- возможность обсуждения заданий с другими учащимися и авторами;
- поддержка геймификации.

Из недостатков Stepik можно назвать:

- ограниченные возможности для корпоративного обучения;

- минимальная кастомизация без возможности создания уникального стиля;
- нет встроенной системы вебинаров.

На рисунке 3 показан внешний вид Stepik.

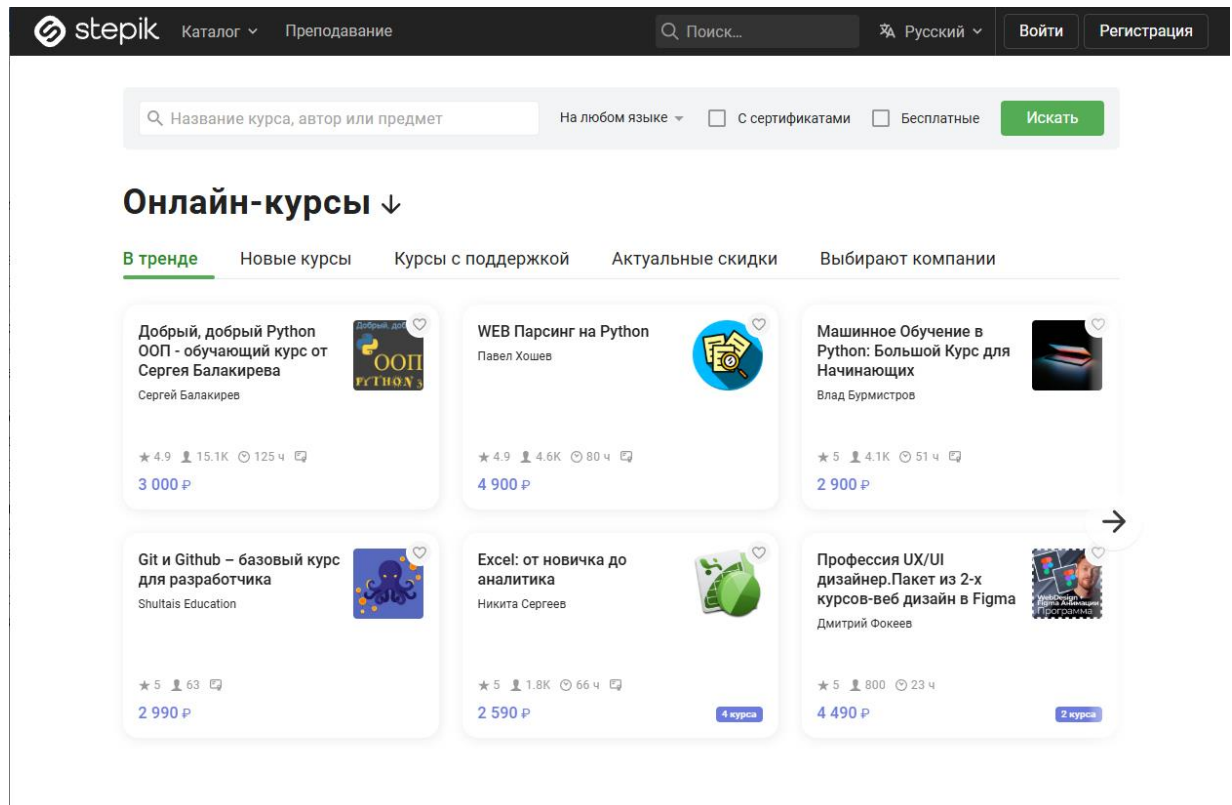


Рисунок 3 – Внешний вид Stepik

Trello – универсальное веб-приложение для управления проектами и задачами, который помогает организовать работу и улучшить продуктивность [8]. Основан на методологии Канбан, позволяя пользователям создавать доски для визуального отслеживания выполнения проектов, которые подразделяются на списки для представления различных этапов. Из возможностей [9] Trello можно выделить:

- доски, которые создаются для каждого проекта или области работы и могут быть публичными или приватными;
- списки, которые разделяют задачи на этапы и могут перемещаться и переименовываться;
- карточки, в которые добавляются задачи со сроками и ответственными и могут перемещаться между списками;

- метки и фильтры, позволяющие организовывать и классифицировать задачи по категориям;
- вложения и чек-листы, доступные внутри каждой задачи и позволяющие добавлять подзадачи или необходимые материалы.

Преимуществами [10] Trello являются:

- простой и понятный интерфейс и возможность начать работу без обучения;
- визуальное управление задачами;
- командная работа, включающая в себя редактирование в реальном времени, комментарии, назначение ответственных и контроль сроков выполнения.

Недостатки Trello:

- не подходит для сложных проектов из-за отсутствия многоуровневых задач и перегрузка доски при большом количестве задач;
- малая отчётность; нельзя поменять стиль досок под корпоративный;
- большая часть функций в платной версии.

На рисунке 4 показан внешний вид Trello.

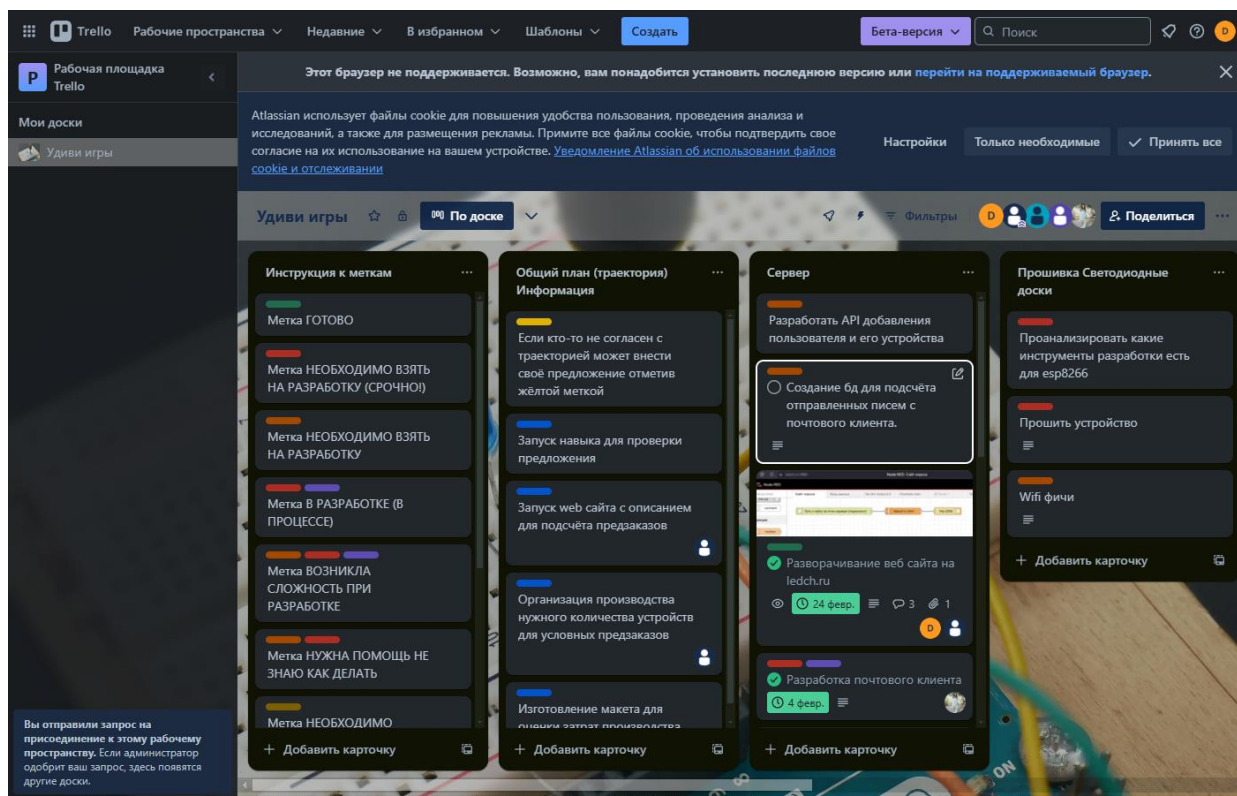


Рисунок 4 – Внешний вид Trello

Сводка данных об аналогах представлена в таблице 1.

Таблица 1 – Сводная таблица характеристик систем-аналогов

Аналогичная система	Достоинства	Недостатки
Moodle	Бесплатность и открытый код. Гибкость и настраиваемость. Поддержка различных форматов обучения.	Устаревший интерфейс. Сложная настройка и администрирование. Отсутствие возможности запуска кода.
Google Classroom	Бесплатность для образовательных учреждений. Интеграция с другими сервисами Google. Редактирование документов и комментирование работ.	Жёсткая привязка к сервисам Google. Отсутствие инструментов для создания интерактивных курсов и встроенного тестирования. Отсутствие возможности запуска кода.
Stepik	Интерактивный формат обучения. Гибкий график обучения. Поддержка геймификации.	Ограниченные возможности для корпоративного обучения. Минимальная кастомизация.
Trello	Простой и понятный интерфейс. Визуальное управление задачами. Гибкость в использовании.	Не подходит для сложных проектов. Малая отчётность. Большая часть функций в платной версии.

На этом фоне просматривается потребность в специализированной системе, совмещающей в себе возможности планирования, контроля, автоматического выполнения заданий и оценки. Особое внимание уделяется адаптивности под различные сценарии, такие как образовательные учреждения, онлайн-школы, корпоративные структуры. Именно такую систему представляет собой проектируемая в рамках этой выпускной квалификационной работы.

1.2 Постановка задачи

Целью выполнения данной выпускной квалификационной работы является разработка информационной системы выполнения практических и проектных заданий, ориентированной на использование в образовательных учреждениях, корпоративных обучающих системах и рабочих коллективах.

Задачи, решаемые в рамках работы:

- разработка веб-интерфейса, удобного и интуитивно понятного для пользователей с различным уровнем подготовки работы с системой;
- разработка архитектуры веб-системы, включающей:
- механизм управления расписанием занятий, проектных этапов и сроками выполнения заданий;
- возможность добавления лабораторных и проектных заданий;
- реализация выполнения заданий в интегрированной среде;
- система ролевого доступа к системе и различный функционал в зависимости от роли;
- модуль оценки результатов: ручной, автоматический и комбинированный;
- возможность развёртывания системы как в интернете, так и в изолированной локальной сети предприятия;
- реализация прототипа системы, тестирование его на различных сценариях, представляющих собой образовательные и корпоративные процессы.
- оценка перспектив масштабирования системы и возможности интеграции с внешними сервисами.

Технические требования ко клиентской части приложения:

- процессор: не менее 4 ядер;
- оперативная память: не менее 2 Гб;
- дисковое пространство: не менее 4 Гб;
- видеокарта: интегрированная.

Технические требования ко клиентской части приложения:

- процессор: не менее 4 ядер;
- оперативная память: не менее 4 Гб;
- дисковое пространство: не менее 8 Гб;
- видеокарта: интегрированная.

Ожидаемые результаты:

Система, позволяющая осуществлять контроль выполнения различных задач в соответствии с графиком, указанным в расписании, а также выполнять совместную

					ПГУ 1.090301.03.001 ПЗ	Лист
						13
Изм.	Лист	№ докум.	Подпись	Дата		

деятельность в рамках задачи и производить оценку результатов этой деятельности.

1.3 Выбор инструментов разработки

Для реализации системы выполнения практических заданий необходимо выбрать надёжные, гибкие и удобные в использовании инструменты, обеспечивающие быструю разработку. Архитектура приложения будет построена по модульному принципу, что будет обеспечивать возможность дальнейшей модернизации проекта и расширения его функциональности. Основное внимание уделяется выбору платформы программирования и системы управления базами данных, так как они являются ключевыми компонентами системы.

Современные системы автоматизации и мониторинга часто строятся с применением платформ визуального программирования, что значительно упрощает процесс разработки за счёт графического интерфейса и минимизации объёма кода. Такие платформы позволяют быстро разрабатывать, отлаживать и масштабировать решения в следствие визуального представления логики обработки данных. Учитывая задачи проекта, выбор платформы должен учитывать поддержку различных протоколов, расширяемость и простоту эксплуатации.

В качестве платформы для реализации может быть выбран Node-Red – инструмент потокового программирования, первоначально разработанный IBM Emerging Technology Services [11]. В настоящее время является частью JS Foundation. Node-Red позволяет объединять аппаратные устройства, API и онлайн сервисы различными способами. В основе работы системы лежит событие, которое генерируется либо внешним источником, либо процессом внутри Node-Red. Из особенностей инструмента [12] можно выделить:

- работа в среде исполнения Node.js;
- для создания и редактирования потока используется браузер;
- создание приложения происходит путём перетаскивания необходимых узлов из палитры в рабочую область и соединением их между собой;
- по одному нажатию кнопки приложение разворачивается и запускается.

Преимущества платформы являются:

- интуитивно понятный интерфейс и отсутствие необходимости в глубоких знаниях программирования;
- большое количество готовых блоков для работы с API, базами данных и др., а также возможность создавать собственные;
- простота развёртывания и открытый код; множество примеров и готовых решений;
- возможность написания автоматизированных процессов.

К недостаткам Node-Red можно отнести:

- сложность отладки в связи с отсутствием встроенного дебаггера;
- некоторая сложность в локализации ошибок из-за визуального представления кода;
- запутанность при использовании большого количества блоков и реализации сложных алгоритмов;
- отсутствие типизации и строгой структуры.

На рисунке 5 показан внешний вид Node-Red.

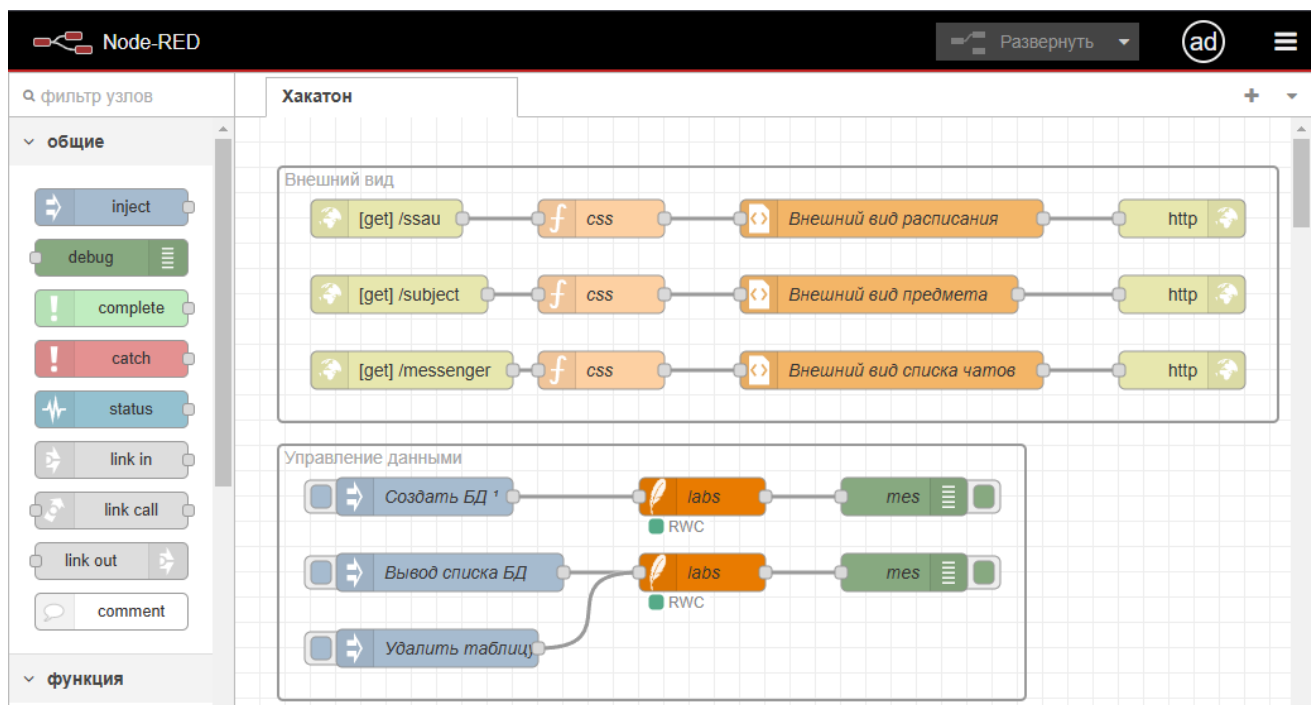


Рисунок 5 – Внешний вид Node-Red

У Node-Red существует несколько аналогов, каждый из которых обладает своими преимуществами и недостатками. Для выбора средства разработки была проделана работа по изучению этих аналогов.

Apache NiFi – программный продукт с открытым кодом, написанный на языке Java и предназначенный для управления и автоматизации потока данных между системами [13]. Изначально проект разрабатывался NSA и назывался Niagara Files, но в 2014 году был передан в Apache Software Foundation. Эта платформа позволяет создавать маршруты для данных: брать данные из одного источника, преобразовывать их и отправлять в другое место для дальнейшего использования. Особенности [14] Apache NiFi:

- поддержка сбора данных из различных источников, в том числе файлы разных форматов, базы данных и др.;
- множество процессоров для операций над данными; поддержка очередей;
- потоковая и пакетная обработка данных.

Из преимуществ Apache NiFi можно назвать:

- визуальное программирование;
- поддержка сложных маршрутизаций;
- работа с разными типами данных;
- гибкость и масштабируемость;
- встроенный мониторинг потоков данных;
- восстановление после сбоев;
- открытый код.

Недостатками платформы являются:

- высокий порог входа, в том числе необходимость понимания принципов потоков данных и сложность работы без обучения;
- требовательность к ресурсам;
- отсутствие встроенных инструментов дебаггинга;
- необходимость ручного управления;
- ограниченная документация.

На рисунке 6 показан внешний вид Apache NiFi.

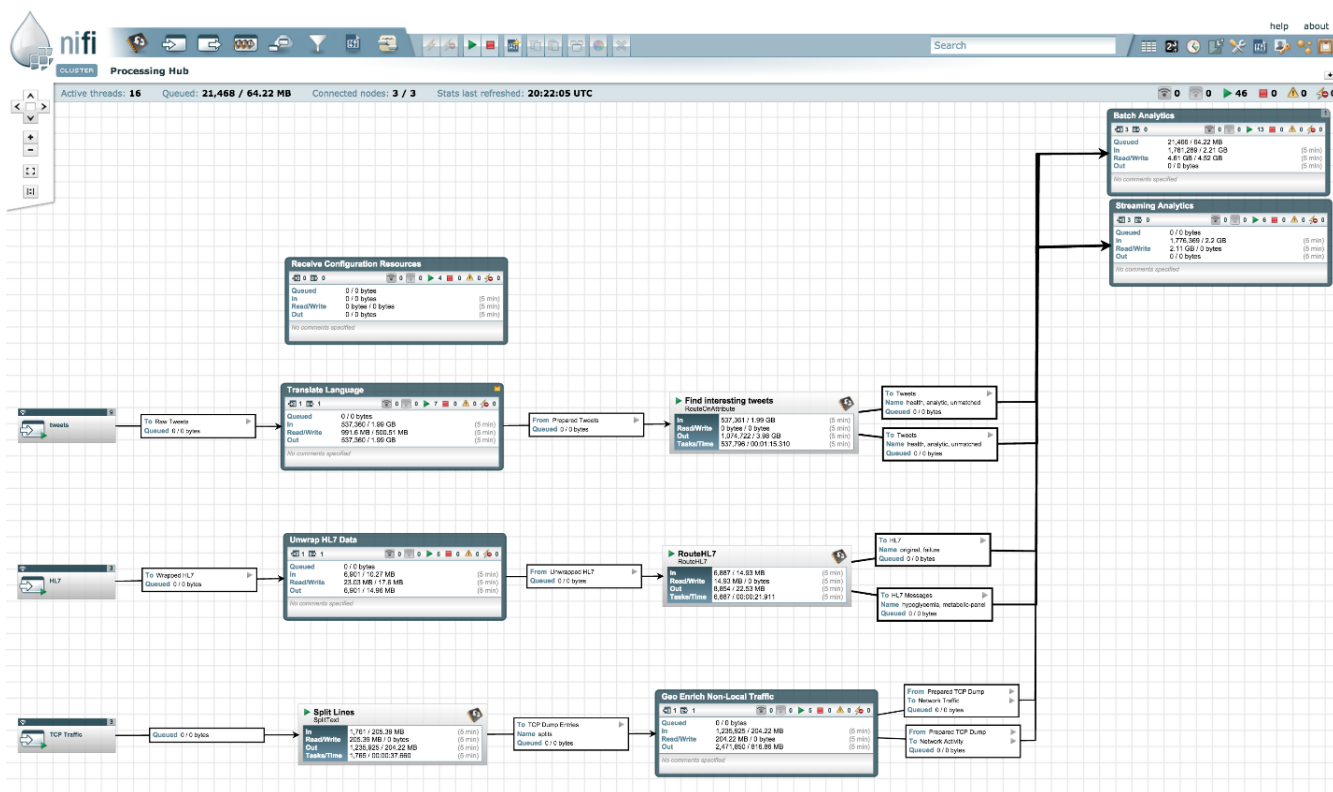


Рисунок 6 – Внешний вид Apache NiFi

N8N – бесплатный и расширяемый инструмент автоматизации рабочих процессов, позволяющий интегрировать различные приложения и сервисы, а также реализовывать серверную логику приложений [15]. Некоторые особенности [16] N8N:

- интеграции с популярными приложениями и сервисами, такими как GitHub, Telegram и др.;
- создание рабочих процессов с помощью блоков, которые могут быть триггерами или действиями;
- поддержка искусственного интеллекта и машинного обучения;
- доступность в виде облачной или локальной версии.

Из преимуществ [17] платформы можно выделить:

- позволяет создавать сложные цепочки задач с условиями, циклами и ветвлениями;
- можно развернуть на своём сервере;
- большое количество встроенных блоков; открытый код;
- удобный интерфейс и подробное логирование;

Изм.	Лист	№ докум.	Подпись	Дата

- гибкость и масштабируемость.

Недостатками N8N выделяют:

- для развёртывания и настройки нужны навыки администрирования серверов;
- нет бесплатной облачной версии;
- меньше готовых шаблонов, чем у конкурентов;
- нет встроенного планировщика задач.

На рисунке 7 показан внешний вид N8N.

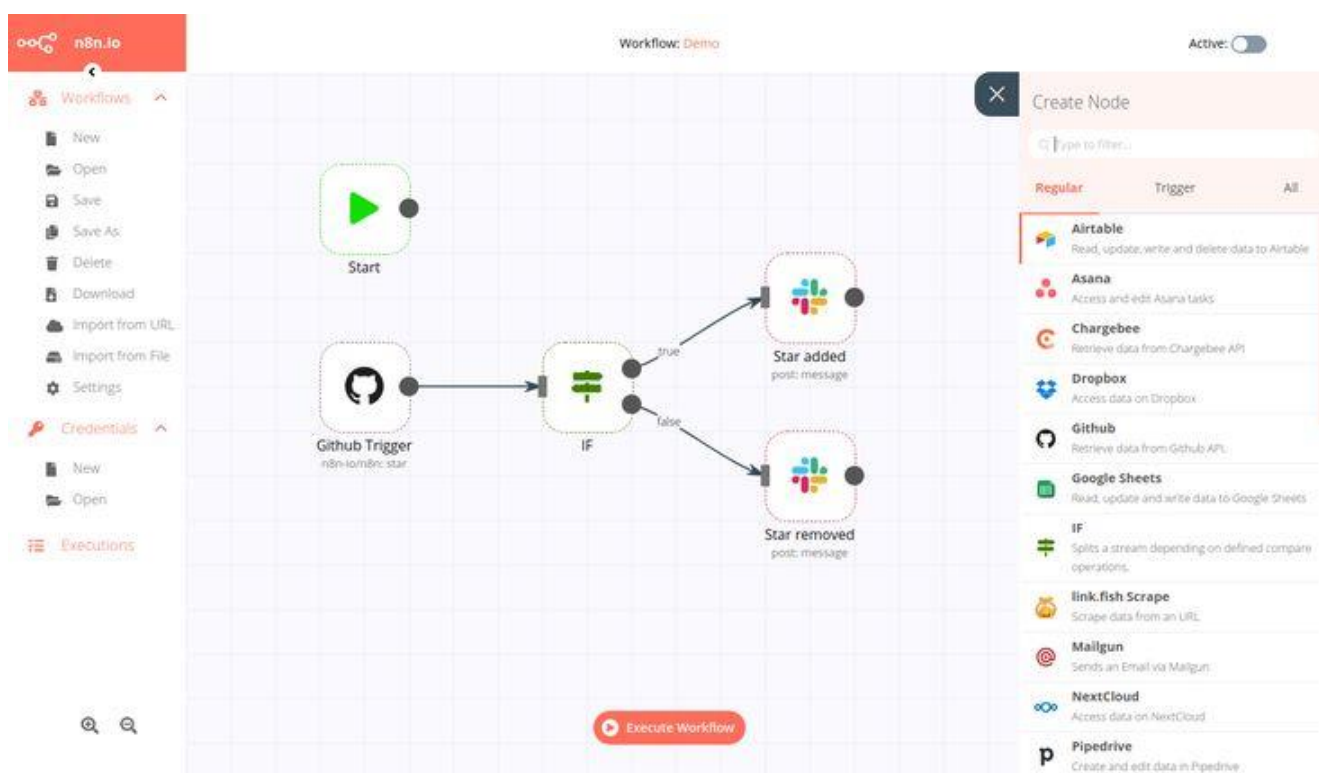


Рисунок 7 – Внешний вид N8N

Zapier – платформа для автоматизации без кодирования, которая позволяет пользователям подключать и автоматизировать рабочие процессы между различными веб-приложениями и службами [18]. С её помощью можно настраивать периодическое выполнение процессов по возникновению какого-либо события без программирования. Все автоматизации создаются с помощью сценариев, для создания которых нужно задать минимум два параметра: триггер и действие.

Преимущества [19] Zapier:

- простота использования, которая заключается в интуитивно понятном интерфейсе и возможности использовать платформу даже нетехническим пользователям;
- огромное количество интеграций;
- готовые шаблоны автоматизации; облачный сервис, доступный из любого места через веб-интерфейс;
- возможность создавать сложные цепочки действий с условиями.

Недостатками Zapier являются:

- ограничения бесплатного тарифа в малое количество задач;
- высокая стоимость платных тарифов;
- ограниченная логика;
- сложность отладки;
- нет возможности использования на своём сервере.

На рисунке 8 показан внешний вид Zapier.

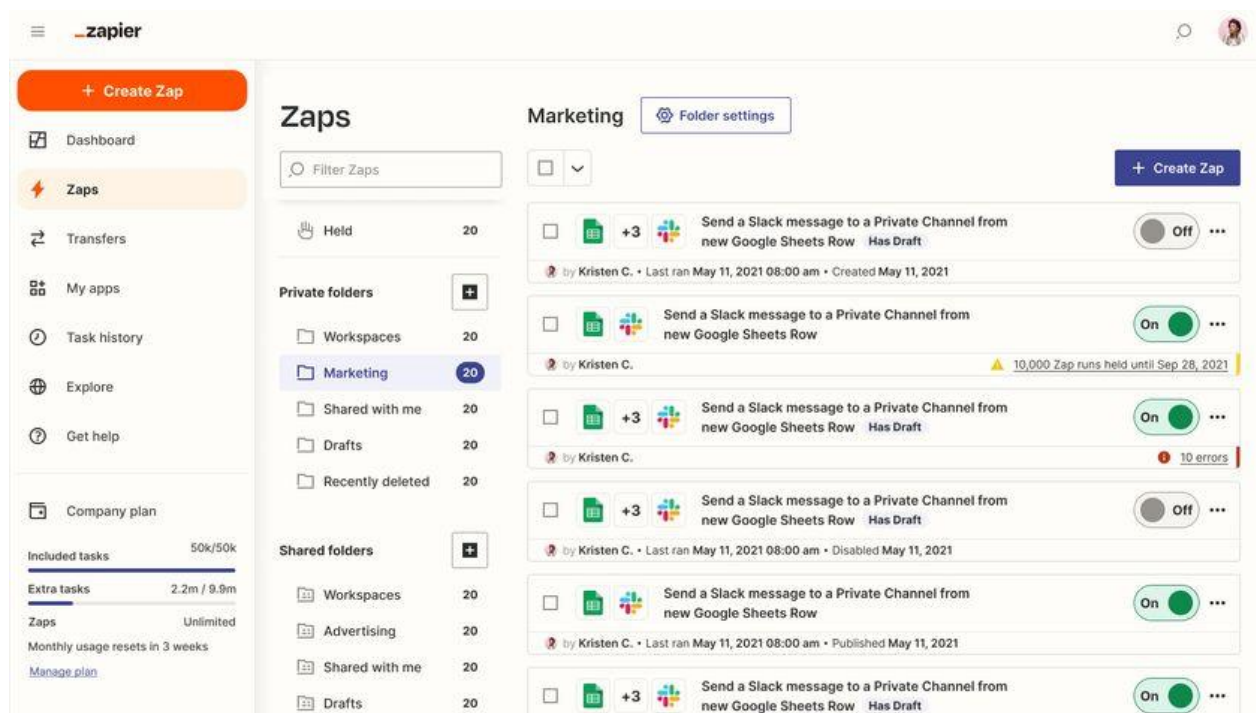


Рисунок 8 – Внешний вид Zapier

Flogo (TIVCO) – проект с открытым исходным кодом для разработки лёгких бессерверных приложений, направленный на упрощение разработки и

развёртывания микросервисов и приложений IoT [20]. Программная часть написана на языке Go Lang, что позволяет её подходящим выбором для запуска служб в средах с ограниченными ресурсами. Некоторые особенности [21] Flogo:

- архитектура предполагает чистое разделение кода и конфигурации, что приводит к тому, что разработчики объявляют поток сообщений во внешнем файле перед компиляцией;

Преимуществами [22] Flogo (TIBCO) можно назвать:

- легковесность и высокая производительность, что позволяет запускать на устройствах с малой мощностью и не потреблять много ресурсов;

- визуальный конструктор потоков;

- открытый код;

Недостатками платформы являются:

- меньше готовых коннекторов;

- сложность отладки;

- меньше документации и сообщество, чем у конкурентов;

- зависимость от лицензий.

На рисунке 9 показан внешний вид Flogo (TIBCO).

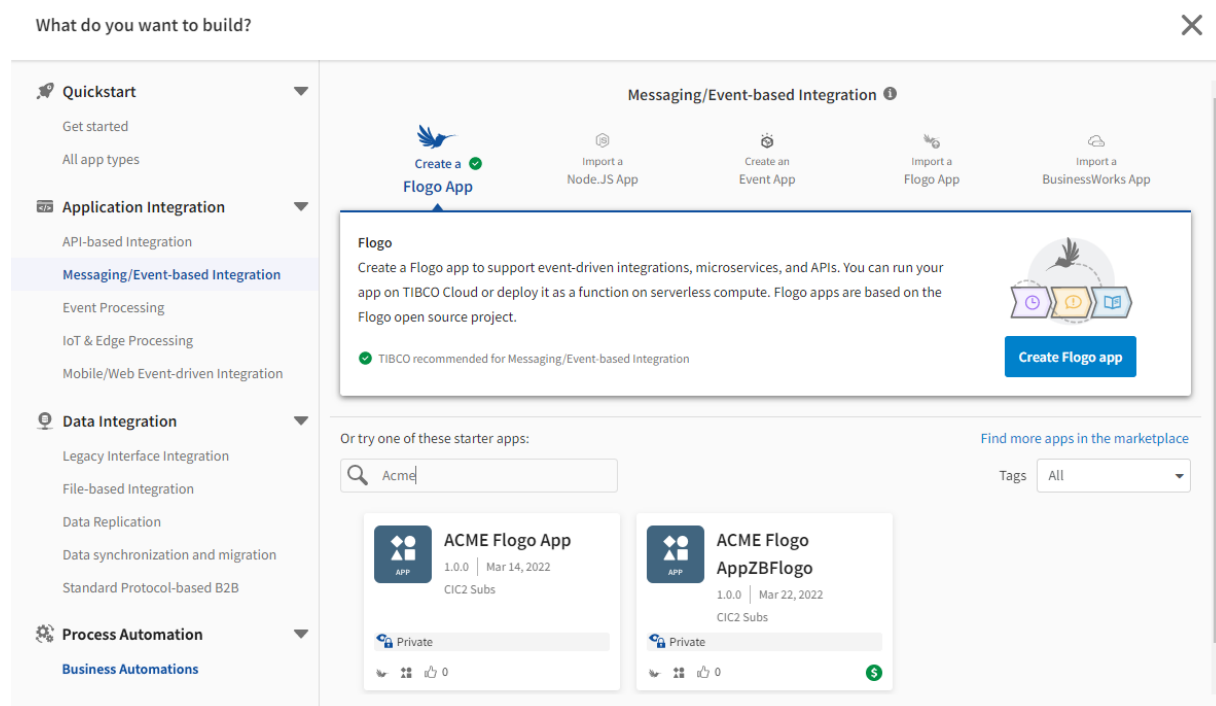


Рисунок 9 – Внешний вид Flogo (TIBCO)

Сводка данных о платформах-аналогах приведена в таблице 2.

Таблица 2 – Сводная таблица характеристик платформ-аналогов

Аналогичная система	Достоинство	Недостаток
Node-Red	Интуитивно понятный интерфейс. Множество примеров и готовых решений. Простота развёртывания.	Сложность отладки. Отсутствие типизации и строгой структуры. Сложность в локализации ошибок.
Apache NiFi	Поддержка сбора данных из различных источников. Встроенный мониторинг потоков данных. Поддержка сложных маршрутизаций.	Высокий порог входа. Ограниченная документация. Требовательность к ресурсам. Необходимость ручного управления.
N8N	Сложные цепочки задач с условиями, циклами и ветвлениями. Удобный интерфейс и подробное логирование.	Нужны навыки администрирования серверов. Меньше готовых шаблонов, чем у конкурентов.
Zapier	Простота использования. Огромное количество интеграций. Облачный сервис.	Ограничения бесплатного тарифа. Стоимость платных тарифов. Сложность отладки.
Flogo (TIBCO)	Легковесность и высокая производительность. Минимальные задержки.	Меньше документации и сообщество, чем у конкурентов. Меньше готовых коннекторов. Сложность отладки.

При анализе аналогичных систем был сделан вывод о том, что сочетание визуального подхода, лёгкости расширяемости, развёртывания и эксплуатации делает Node-Red оптимальным выбором для разработки системы. Node-Red также имеет встроенную поддержку sqllite, имеет широкий выбор библиотек, гибкий при построении логики обработки данных и событий и имеет большое количество документации и готовых шаблонов, что добавляет обоснованности выбора его в качестве основы для проекта выполнения практических заданий.

Выбор системы управления базами данных играет одну из ключевых ролей и обеспечении надёжных хранения, обработки и доступа к информации о предметах, расписании и лабораторных работах. В условиях ограниченных вычислительных ресурсов и необходимости автономной работы предпочтение

отдаётся лёгким и встраиваемым решениям. В данном проекте основными критериями являются компактность, простота использования, поддержка SQL-запросов и интеграция с платформой Node-Red.

Для этих целей доступна для выбора SQLite – лёгкая встраиваемая СУБД, не требующая серверного окружения и обеспечивающая высокую производительность для задач системы. Все данные в SQLite хранятся в одном файле, что упрощает работу с базой и позволяет легко переносить данные из одного места в другое, что может быть очень важно для системы выполнения практических заданий. Из преимуществ SQLite хочется отметить:

- высокая скорость, которая достигается вызовом компонентов в одном процессе с приложением;

- нулевая конфигурация;
- кроссплатформенность;
- высокая производительность;
- поддержка стандартных SQL-запросов.

Из недостатков:

- не подходит для высоконагруженных систем;
- ограниченная масштабируемость.

Из альтернативных СУБД в первую очередь можно отметить MySQL – свободная реляционная СУБД. Из особенностей:

- поддержка различных типов данных;
- индексация;
- безопасность с разным уровнем доступа;
- поддержка репликации и кластеризации.

Преимуществами MySQL являются:

- высокая производительность на высоконагруженных системах;
- масштабируемость;
- гибкость и совместимость;
- хорошая бесплатная версия.

Недостатки СУБД:

- ограниченная функциональность;

- требует тонкой настройки;
- лицензионные ограничения.

Ещё одним аналогом является PostgreSQL – более мощная реляционная СУБД с открытым кодом и расширенным функционалом. Особенности являются:

- поддержка стандарта SQL;
- гибкость с возможностью создания функций, операторов и др.;
- поддержка JSON и XML;
- вертикальная и горизонтальная масштабируемость.

Преимущества PostgreSQL:

- мощный функционал, включающий в себя оконные функции, рекурсивные запросы, массивы и пользовательские типы данных;
- гибкость и расширяемость;
- производительность и оптимизация;
- масштабируемость; активное сообщество.

Из недостатков:

- высокий порог входа;
- медленнее в простых операциях;
- ограниченная поддержка.

Сводка данных о СУБД-аналогах приведена в таблице 3.

Таблица 3 – Сводная таблица характеристик СУБД-аналогов

Аналогичная система	Достоинства	Недостатки
SQLite	Высокая скорость. Нулевая конфигурация. Простота развёртывания.	Не подходит для высоконагруженных систем. Ограниченная масштабируемость.
MySQL	Высокая производительность на высоконагруженных системах. Гибкость и совместимость.	Лицензионные ограничения. Требует тонкой настройки.
PostgreSQL	Мощный функционал. Гибкость и расширяемость. Производительность и оптимизация.	Ограничения бесплатного тарифа. Стоимость платных тарифов. Сложность отладки.

Аналогичная система	Достоинства	Недостатки
Flogo (TIBCO)	Легковесность и высокая производительность. Минимальные задержки.	Высокий порог входа. Медленнее в простых операциях.

В результате проведенного анализа был сделан вывод о выборе SQLite в качестве СУБД при разработке данной системы как оптимального решения для компактной системы, не требующей сложной архитектуры и обладающей ограниченным объёмом данных и пользователей. Одним из важных факторов стала совместимость с Node-Red и отсутствие необходимости в отдельном сервере, однако, в случае разработки системы с повышенной загруженностью, имеет смысл рассмотреть переход на более совершенную СУБД.

					ПГУ 1.090301.03.001 ПЗ	Лист
						24
Изм.	Лист	№ докум.	Подпись	Дата		

2 Разработка программы выполнения практических заданий

2.1 Архитектура приложения

Приложение состоит из нескольких компонентов, взаимодействующих между собой для реализации отображения расписания занятий, различных предметов и преподавателей по этим предметам, практических заданий и платформы для их выполнения. Архитектура включает в себя следующие основные блоки:

- блок внешнего вида системы;
- блок управления базой данных;
- блок данных;
- блок управления данными таблиц в БД.

Архитектура программы представлена на рисунке 10.

Система выполнения практических заданий построена на визуальной среде программирования Node-Red, которая обеспечивает гибкую интеграцию между визуальной частью, обработкой и управлением учебными данными. Первичным звеном системы выступает блок визуализации данных, он же блок внешнего вида. Этот блок отвечает за формирование графического интерфейса, через который пользователь получает представление о возможностях системы и возможность пользоваться всеми доступными функциями системы. Через этот интерфейс студент или преподаватель может просматривать расписание занятий, список доступных предметов, а также отслеживать лабораторные работы, привязанные к каждому предмету. Визуальный интерфейс оформлен с использованием HTML и CSS, что позволяет получение доступа к информации в структурированном виде. Пользователь может в реальном времени отслеживать актуальные изменения в информации, что добавляет удобства при использовании системы.

Следующим ключевым компонентом является блок обработки и предоставления данных, который обеспечивает логику выборки, агрегации и форматирования данных, необходимых для отображения и дальнейшей работы с системой. Он обеспечивает обработку запросов, определение запрашиваемой

сущности и генерацию ответов в структурированном виде. Этот модуль формирует логическую модель предметной области: связывает лабораторные работы с соответствующими предметами, преподавателями, статусами выполнения и прочими характеристиками. Он играет роль преобразователя, подготавливая данные в нужной форме для их передачи в визуальные компоненты.

Блок управления базой данных реализует инструменты для создания, удаления и изменения таблиц в базе данных, используемых системой. В рамках этого модуля осуществляется инициализация структуры хранилища, поддерживается целостность данных и предоставляются сервисные команды, позволяющие взаимодействовать с таблицами базы данных в ходе работы приложения. В частности, данный блок позволяет систематизировать информацию, хранящуюся в таблице с лабораторными работами. Гибкая структура базы данных позволяет в любой момент адаптировать систему под новые дисциплины или учебные форматы.

Отдельно выделяется блок управления данными лабораторных работ, ориентированный на непосредственную работу с задачами, входящими в учебный план. Через него реализуются функции добавления новых лабораторных заданий, их удаления, изменения содержания и отслеживания статуса выполнения. Этот блок принимает пользовательские действия с лабораторной работой и применяет их к соответствующим записям в базе данных. Помимо этого, он позволяет отслеживать и логировать изменения статусов заданий, а также управлять временными параметрами лабораторных работ, что может послужить основой для внедрения механизмов уведомлений или адаптивного планирования. За счёт применения промежуточных преобразований данные проверяются на корректность, после чего формируются команды для выполнения в таблице. Это обеспечивает целостность данных и минимизирует возможность ошибок, а также добавляет возможность реализации различных форм сдачи лабораторных и ввода данных.

Все компоненты системы взаимодействуют по маршруту данных, организованному в рамках среды Node-Red. Визуальный интерфейс инициирует запросы, модуль обработки данных и модули работы с базой данных

подготавливают необходимую информацию и обеспечивают сохранность и корректность операций, после чего обновлённая информация возвращается в интерфейс пользователя.

Таким образом, архитектура системы сочетает в себе интуитивно понятную визуальную среду, чёткую логику обработки данных и инструменты управления хранилищем, что делает её эффективной для целей автоматизации выполнения практических заданий. Выбранная архитектура облегчает сопровождение и масштабирование системы, позволяет интегрировать внешние модули и расширять функциональность без необходимости полной переработки системы.

На рисунке 10 отображена схема архитектуры программы.

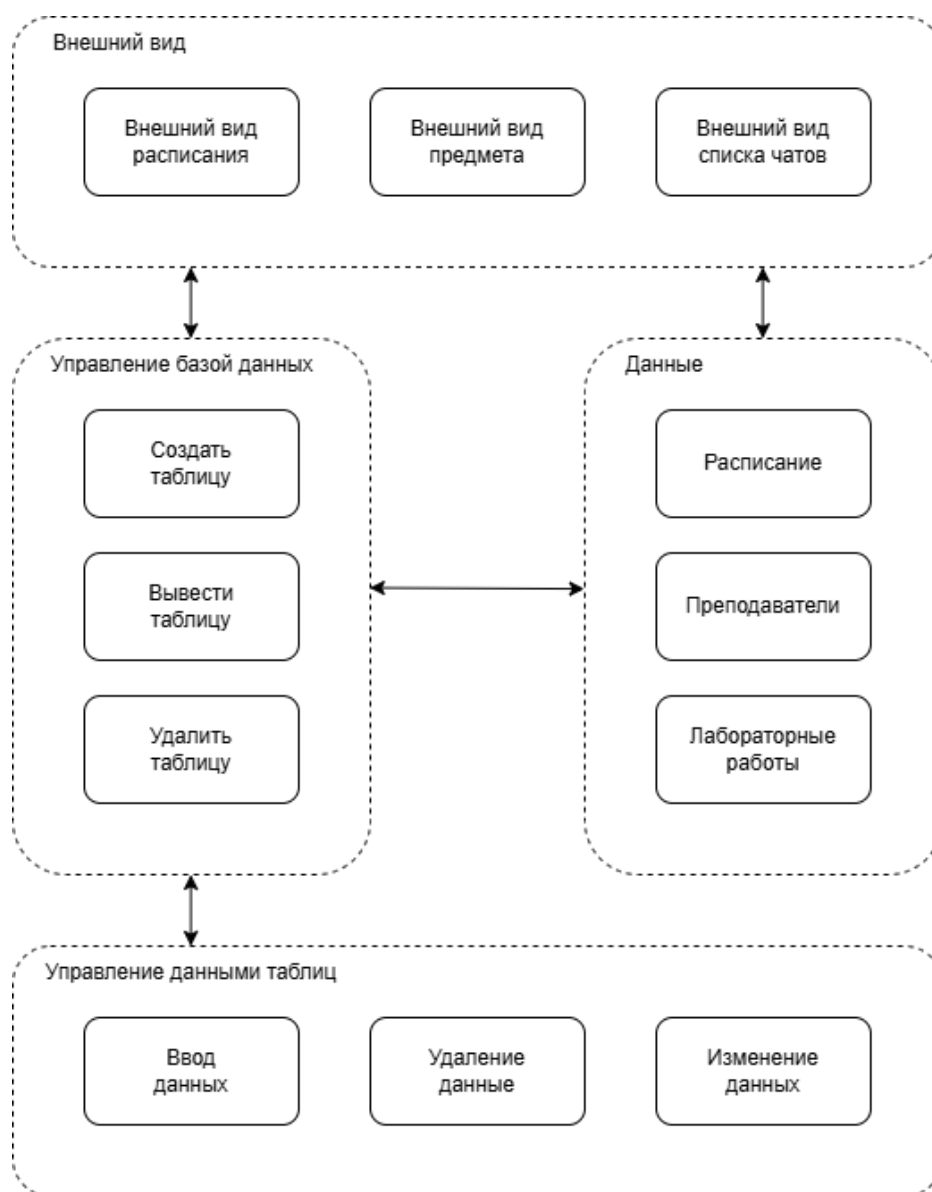


Рисунок 10 – Схема архитектура программы

2.2 Проектирование и разработка

2.2.1 Настройка среды разработки

Для реализации информационной системы была выбрана платформа Node-RED – визуальная среда программирования, позволяющая создавать потоки обработки данных. Она предоставляет интуитивно понятный интерфейс, где логика проекта строится с использованием отдельных узлов, называемых нодами. Эти ноды соединяются между собой и образуют цепочки выполнения.

Для начала работы необходимо установить несколько компонентов. Первым из них выступает Node.js, так как именно на ней базируется функционирование Node-RED. Node.js обеспечивает серверную среду выполнения JavaScript кода, а также включает в себя менеджер пакетов npm, с помощью которого осуществляется установка дополнительных модулей. Для установки Node.js переходим на официальный сайт и нажимаем кнопку Download (рисунок 11).

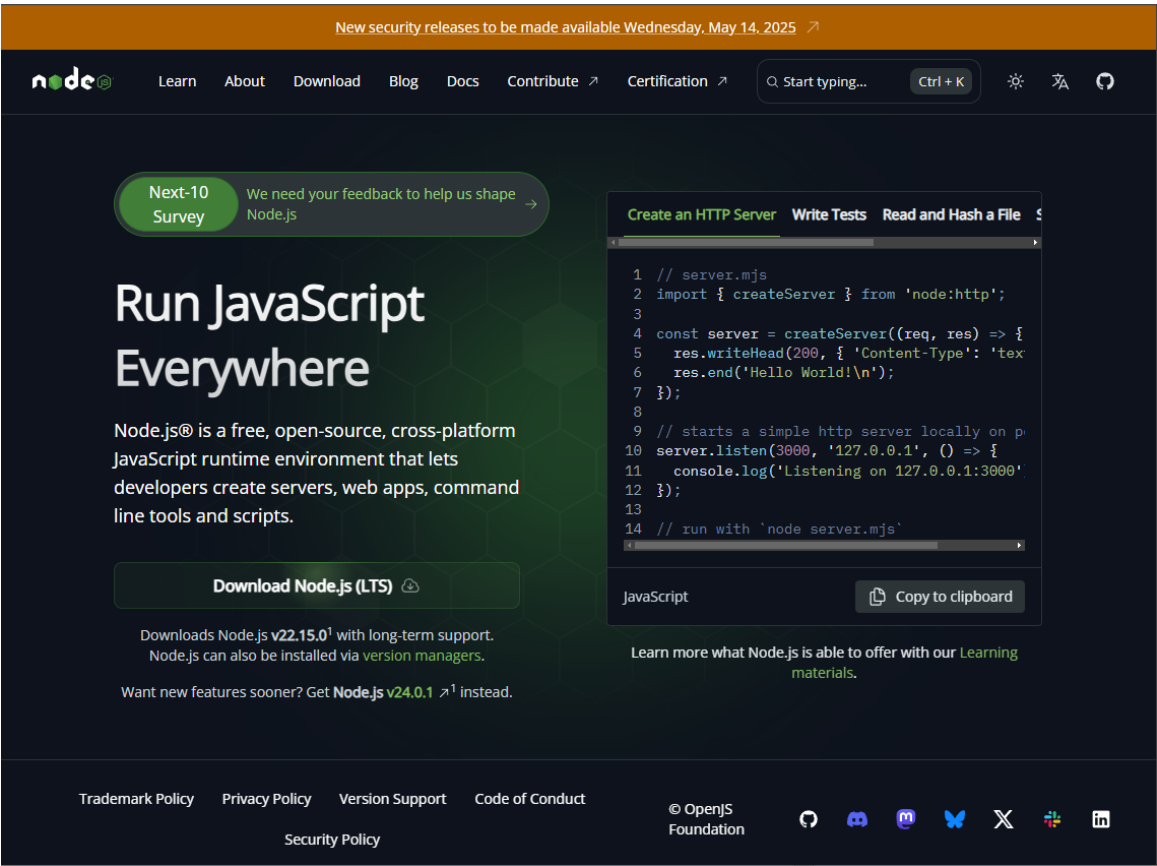


Рисунок 11 – Сайт Node.js

Далее открываем скачанный файл и действуем в соответствии с вариантами, предлагаемыми программой-установщиком. После этого приступаем к установке самого Node-RED [23]. Для этого в командной строке выполняем команду `npm install -g --unsafe-perm node-red`, после чего он становится доступен для запуска на локальной машине. Запуск Node-RED осуществляется через команду `node-red` в командной строке. После этого веб-интерфейс становится доступен по адресу `http://localhost:1880`. Визуальный интерфейс Node-RED [24] позволяет приступить к созданию проекта путём перетаскивания нужных нод на рабочую область и соединения их между собой (рисунок 12). Каждый узел можно настраивать, задавая параметры его работы. Из доступных возможностей можно установить путь до внешнего ресурса, формат данных или логику обработки. После построения потока достаточно нажать кнопку «Развернуть», чтобы изменения вступили в силу, и система начала работать в соответствии с заданной логикой.

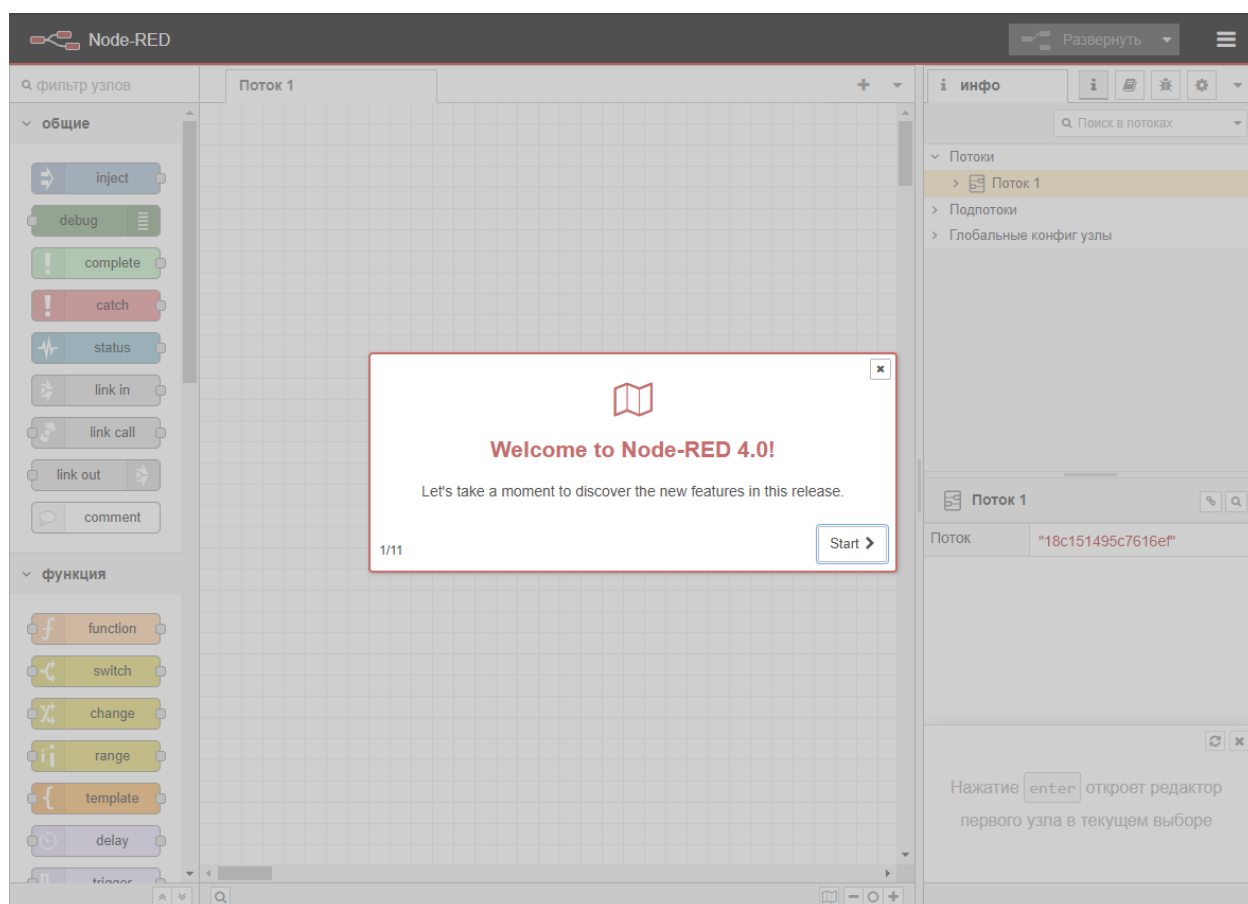


Рисунок 12 – Внешний вид Node-RED при первом запуске

Для того, чтобы выбрать готовый файл, достаточно нажать на кнопку в правом верхнем углу и выбрать «Импорт», после чего можно будет выбрать json-файл, содержащий проект. Панель действий представлена на рисунке 13.

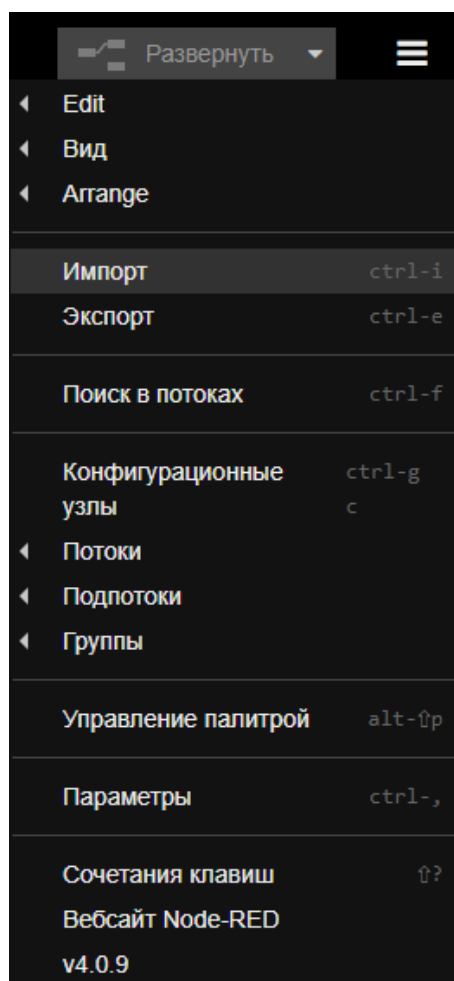


Рисунок 13 – Меню доступных действий

Создание полноценного проекта требует интеграции с внешними источниками данных, базами данных и другими сервисами [25]. Для этого в Node-RED предусмотрена возможность подключения дополнительных модулей, которые могут устанавливаться как через интерфейс среды, так и через командную строку. В интерфейсе «Управление палитрой» доступен раздел «Установить», в котором по названию модуля можно найти и подключить необходимый функционал (рисунок 14). В данном случае нам нужен модуль node-red-node-sqlite. Для установки через терминал используется команда `npm install`, после которой установленный модуль становится доступным в палитре нод [26].

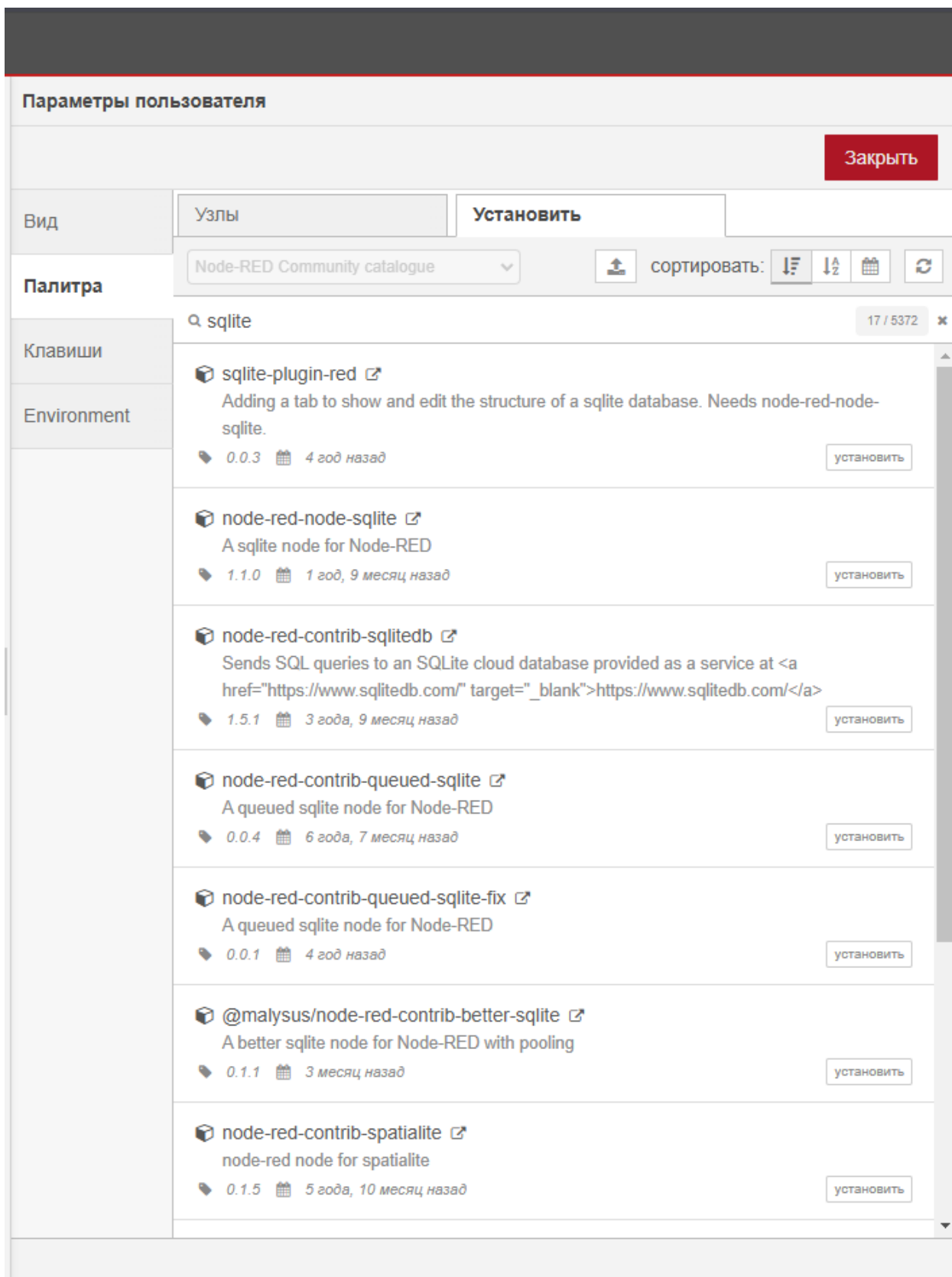


Рисунок 14 – Установка модуля

Одной из важнейших составляющих разработанной системы является хранение и обработка информации о практических заданиях, расписании занятий

и результатах выполнения. Для решения этой задачи была выбрана встроенная база данных SQLite, которая обеспечивает простоту использования и не требует развёртывания отдельного сервера. Для взаимодействия Node-RED с SQLite необходимо установить модуль node-red-node-sqlite. После установки в палитре появляется новая нода, предназначенная для выполнения SQL-запросов [27]. Подключение к базе данных осуществляется путём указания пути к файлу базы в настройках соответствующей ноды. После этого можно использовать SQL-команды для вставки, выборки, обновления и удаления данных прямо из потоков Node-RED. Запросы передаются в текстовом виде и могут содержать как статические выражения, так и переменные, подставляемые из других нод.

Node-RED обладает широкими возможностями, которые делают его мощным инструментом для разработки информационных систем. Среди этих возможностей следует отметить поддержку различных протоколов взаимодействия, таких как HTTP, WebSocket, что делает систему пригодной для интеграции с IoT-устройствами, веб-клиентами и внешними API. Также доступна работа с различными типами баз данных, включая не только SQLite, но и MySQL, PostgreSQL. С помощью специального модуля node-red-dashboard можно создавать пользовательские веб-интерфейсы с кнопками, графиками, формами и другими элементами управления. Кроме того, поддерживается выполнение пользовательского JavaScript-кода через ноды function, что позволяет реализовывать сложную логику обработки данных.

Таким образом, настройка среды разработки на базе Node-RED представляет собой несложный, но важный этап, который закладывает основу для последующей реализации проекта. Среда предоставляет достаточную гибкость и функциональность для построения как простых, так и масштабируемых информационных решений, обеспечивая быструю визуальную сборку, интеграцию с базами данных и другими внешними системами, а также создание интерактивных пользовательских интерфейсов.

Ещё одним компонентом, необходимым для работы системы, является Jupyter Notebook. Эта интерактивная веб-среда для написания и выполнения кода,

в нашем случае это код на языке Python. Jupyter требует установленного Python версии 3 и выше, который можно скачать с официального сайта, нажав кнопку «Download» (рисунок 15).

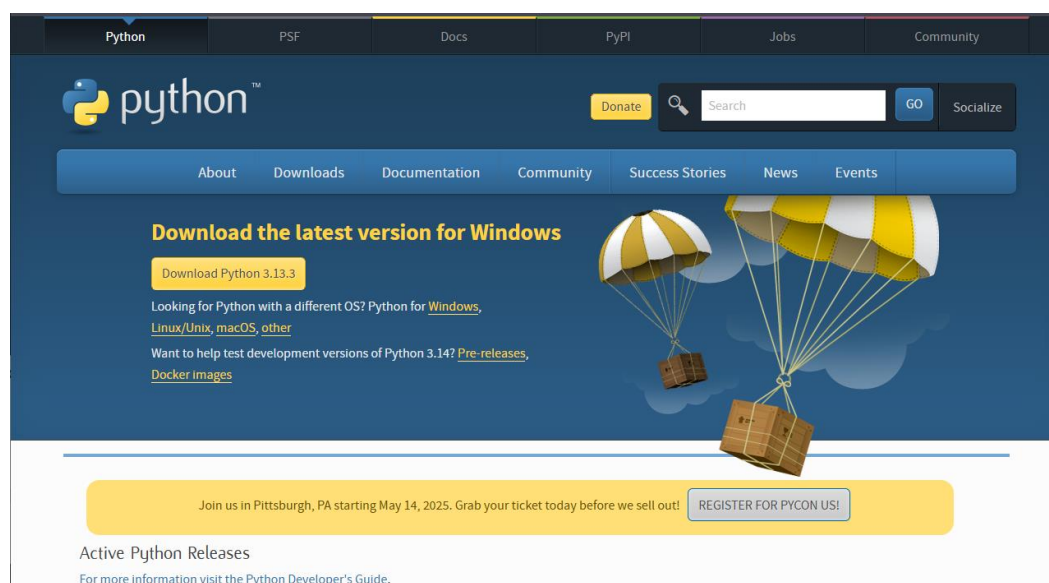


Рисунок 15 – Сайт Python

При установке нужно не забыть отметить галочку Add Python to PATH, чтобы была возможность использования его из командной строки. Далее открываем командную строку и выполняем команду `pip install notebook`. После этого командой `python -m notebook` запускаем сервер, который позволит нам использовать встраиваемый интерфейс запуска кода внутри браузера в нашей системе. Внешний вид представлен на рисунке 16.

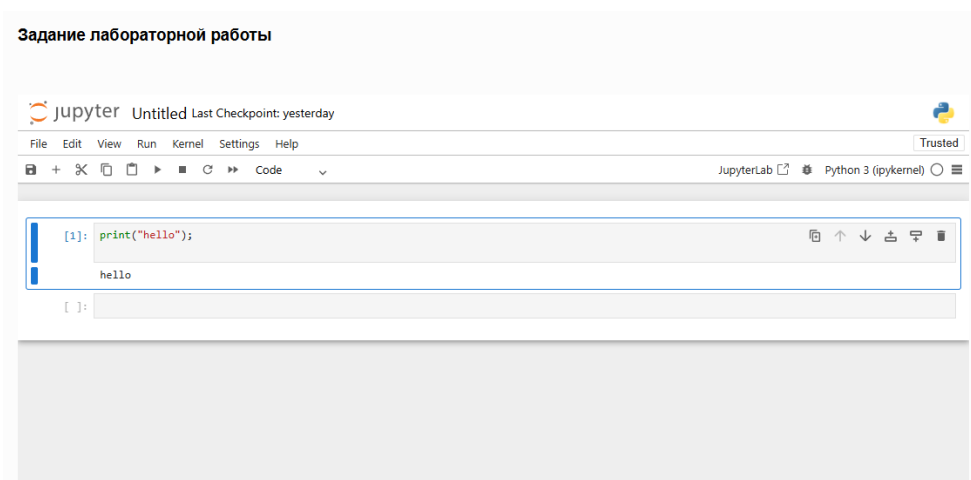


Рисунок 16 – Jupyter Notebook, встроенный в систему

Для корректной работы может потребоваться дополнительная настройка. Она осуществляется посредством изменения файла `jupyter_notebook_config.py`. Он должен находиться в папке `.jupyter` в паке пользователя. В случае его отсутствия его нужно сгенерировать, выполнив команду `python -m --generate-config`. Далее нужно открыть этот файл в текстовом редакторе и отредактировать следующие строки:

- `c.ServerApp.token = ""` – отключает обязательный токен;
- `c.ServerApp.password = ""` – отключает пароль;
- `c.ServerApp.disable_check_xsrf = True` # Отключает проверку XSRF;
- `c.ServerApp.allow_origin = '*'` – разрешает запросы с любого источника;
- `c.ServerApp.allow_remote_access = True` – разрешает удаленный доступ;
- `c.ServerApp.open_browser = False` – не открывать браузер автоматически;
- `c.ServerApp.tornado_settings = {`
 `'headers': {`
 `'Content-Security-Policy': "frame-ancestors 'self' *;"`
 `}`
`}` – разрешить встраивание в фреймы с любого сайта.

2.2.2 Описание разработанной программы

Далее приводится описание разработанных алгоритмов функционирования программного интерфейса, а также структура взаимодействия модулей и ресурсов системы. В компонентах был реализован удобный и интуитивно понятный пользовательский интерфейс, соответствующий современным требованиям, таким как:

- графический режим работы;
- наглядность предоставления информации;
- единые стандарты для основных действий и элементов;
- набор стандартных элементов управления;
- прямое взаимодействие с графическими элементами.

Интерфейс – это способ взаимодействия между программой и пользователем или другими программами. Графический пользовательский интерфейс (GUI) представляет собой среду, обеспечивающую взаимодействие приложений с

пользователем через многооконное представление данных. От удобства интерфейса зачастую напрямую зависит удобство пользования программой, возможность раскрыть её потенциал и воспользоваться всеми доступными функциями. Поэтому разработка удобного для пользователя интерфейса является весьма сложной и время-затратной работой, которая, однако, повышает комфорт и позитивный опыт при использовании системы.

В ходе проектирования системы одной из основных реализованных функций выступила комплексная регистрация и авторизация пользователей, а также разделение их по ролям. Система использует встроенную СУБД SQLite для хранения информации о пользователях. При инициализации приложения предусмотрено срабатывание узла, инициирующего создание таблицы users в случае, если её не существует. Таблица содержит основные поля: уникальный идентификатор пользователя, имя пользователя, пароль, группу, роль и дату регистрации. Благодаря использованию SQLite, база данных не требует отдельного сервера для работы, что делает архитектуру легче и автономнее. Дополнительно предусмотрены команды для вывода списка таблиц в базе, просмотра данных всех пользователей, удаления записей по идентификатору и очистка всей таблицы. Эти команды реализованы через визуальные потоки, объединяющие SQL-запросы с логикой их активации [28]. Для административных целей эти функции доступны к выполнению с сервера в любой момент, а также содержат блоки логирования, что облегчает их использование. Внешний вид модуля представлен на рисунке 17.

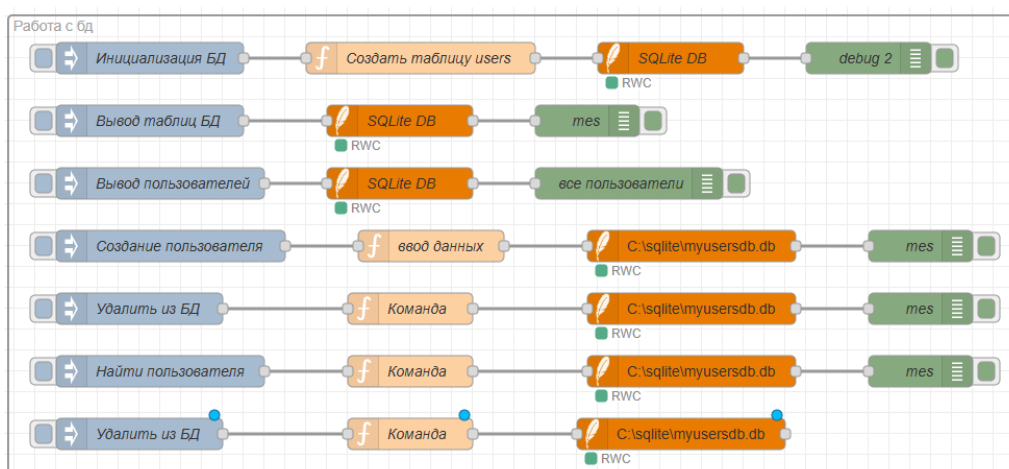


Рисунок 17 – Блок работы с БД

Регистрация пользователей реализуется как отдельный функциональный модуль (рисунок 18). Веб-интерфейс, построенный на HTML-шаблоне, позволяет пользователю в процессе регистрации выбрать роль – студент или преподаватель. При заполнении формы данные отправляются через POST-запрос, и на стороне сервера происходит первичная обработка входных параметров [29]. На этом этапе в базе данных проверяется наличие пользователя с указанным именем, чтобы избежать дублирования. Если пользователь с таким логином уже существует, запрос на добавление отменяется и пользователю возвращается сообщение об ошибке. В случае, если совпадений в БД не найдено, формируется SQL-команда вставки данных в таблицу, где дополнительно сохраняются роль и принадлежность к группе, которые заполняются в зависимости от вводимой пользователем информации и выбранной пользователем формы. После успешной регистрации реализуется автоматическая переадресация на страницу входа.

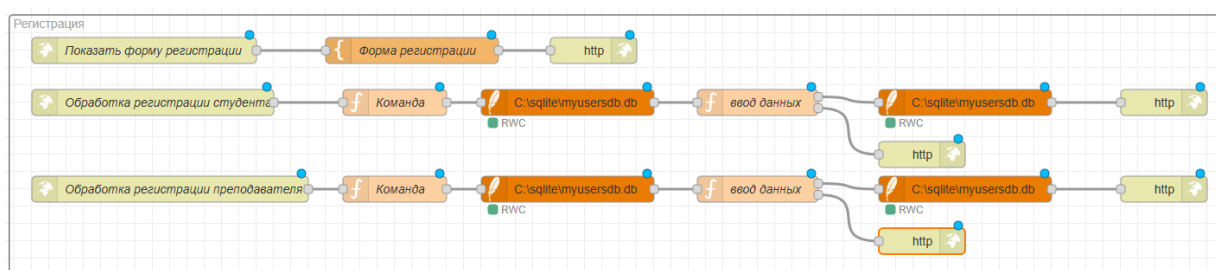


Рисунок 18 – Блок регистрации

Блок аутентификации (рисунок 19) отвечает за проверку учётных данных при входе. Когда пользователь отправляет форму авторизации, происходит извлечение введённых имени и пароля, которые сравниваются с данными, хранящимися в базе. Для этого используется SQL-запрос с фильтрацией по двум полям. В случае успешного совпадения формируется специальный HTTP-ответ с куки-файлом, содержащим уникальный маркер сессии, срок действия которого ограничен отдельным параметром. Данный куки устанавливается как ttpOnly, что защищает его от доступа со стороны клиентских скриптов. Также реализовано автоматическое перенаправление пользователя на защищённую страницу. Если же введённые данные неверны или пользователь не найден, система возвращает сообщение об ошибке без создания куки, оставаясь на странице авторизации.

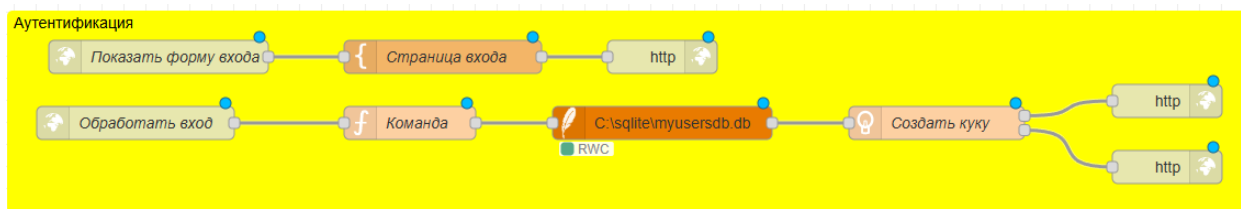


Рисунок 19 – Блок аутентификации

Завершение работы пользователя осуществляется через отдельную точку входа, доступную по запросу выхода [30]. При его активации создаётся HTTP-ответ, в котором происходит принудительное удаление установленной ранее куки. Это реализуется путём назначения срока действия в прошлом, тем самым гарантируя завершение сессии. После удаления сессионного маркера пользователь перенаправляется обратно на страницу входа, что обеспечивает логическую замкнутость пользовательского потока и предотвращает несанкционированный доступ к защищённым разделам системы после выхода (рисунок 20).

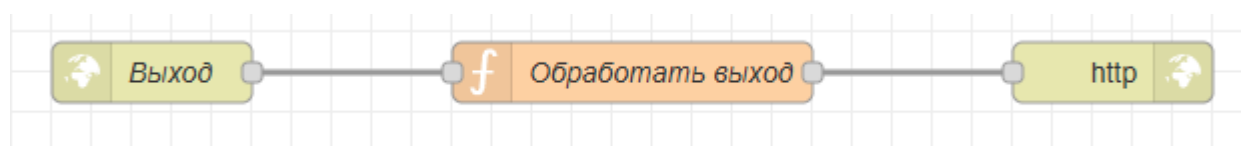


Рисунок 20 – Блок выхода

Вся логика взаимодействия между модулями построена на чётко определённой маршрутизации и передаче функций между узлами: от получения HTTP-запроса через веб-интерфейс до выполнения SQL-запросов к базе данных и возврата результатов клиенту. Система разработана таким образом, чтобы обеспечить целостность данных, простоту расширения и поддержку базовых принципов безопасности. Благодаря визуальной среде Node-Red [31], реализация получилась наглядной, легко сопровождаемой и гибкой к модификациям.

В рамках основной логики приложения, реализованной в Node-Red [32], разработан поток, обеспечивающий взаимодействие пользователей с ключевыми разделами системы: расписанием занятий, информацией о предметах и лабораторных работах. Этот поток объединяет работу с пользовательским интерфейсом, обработку данных на серверной стороне и доступ к базе данных,

в которой хранится информация о событиях и лабораторных заданиях. Таким образом, каждый элемент потока выполняет строго определённую функцию, обеспечивая полнофункциональную логику системы в рамках концепции визуального программирования.

Работа основного пользовательского интерфейса начинается с запроса на отображение страницы расписания. Доступ к этой странице, как и ко всем остальным в этом потоке, пользователь может получить только после прохождения авторизации [33]. Когда пользователь обращается к системе, отправляя HTTP-запрос, активируется цепочка узлов, формирующих HTML-документ, в который включаются стили, элементы навигации и клиентская логика отображения данных. На стороне клиента реализована система динамического построения таблицы расписания. При загрузке страницы выполняется JavaScript-код, который обращается к серверу с запросом на получение расписания [34]. Серверный поток обрабатывает этот запрос, извлекая из базы данных события, соответствующие текущей неделе и возвращает их в виде структурированного массива. Эти данные затем визуализируются в виде таблицы с разбивкой по дням и временным интервалам. Если в конкретной ячейке уже имеется занятие, в ней отображается информация о типе занятия, преподавателе, аудитории или онлайн-платформе. Если ячейка пуста, отображается кнопка добавления события, активирующая модальное окно, в которое предлагается ввести данные планируемого занятия. Таким образом, пользователь может в режиме реального времени как просматривать расписание, так и редактировать его, добавляя события.

Нода «Внешний вид расписания» содержит в себе HTML-шаблон, реализующий интерфейс страницы расписания. Документ начинается с базовой структуры: он включает мета-теги, заголовок, ссылку на иконку и блок стилей, куда подставляется CSS-код через шаблонизатор Mustache. Это позволяет гибко подключать файл стилей, который в данном случае находится в ноде [35], подключенной слева к текущей. Блок расписания представлен на рисунке 21.

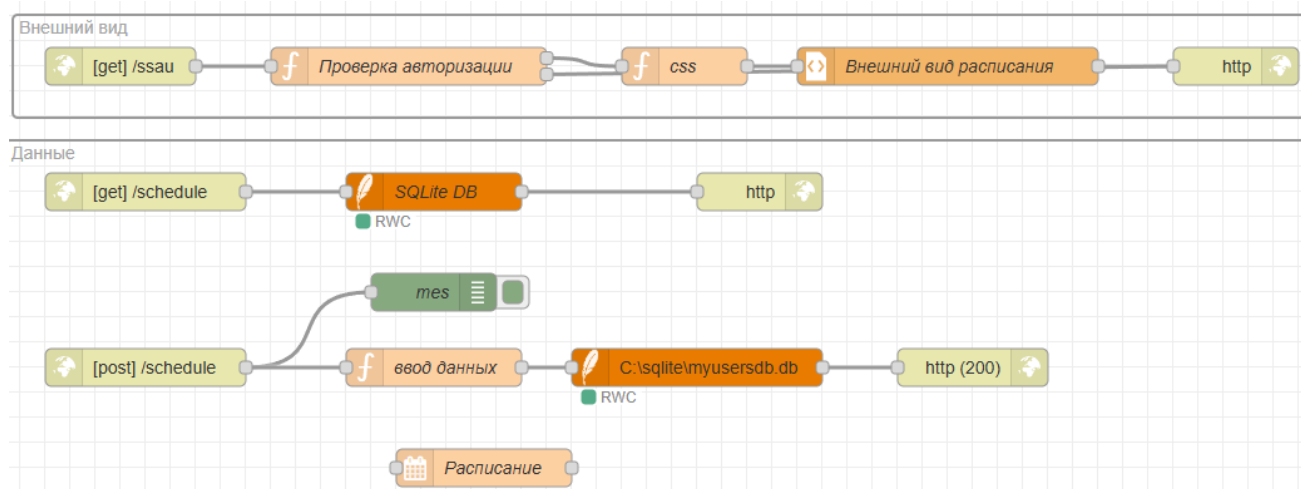


Рисунок 21 – Блок расписания

Одним из ключевых элементов является модальное окно с `id="eventModal"`, предназначенное для ввода новых событий в расписание. Оно содержит текстовые поля для ввода названия группы, названия предмета, имени преподавателя, аудитории, ссылки на конференцию и выбора платформы проведения события, а также выпадающий список с выбором типа занятия: лекция, лабораторная, практика или экзамен. Ниже расположены две кнопки – «Ок» вызывает функцию `saveEvent`, которая сохраняет новое событие, а «Отмена» вызывает `closeModal` и закрывает модальное окно.

Основная часть страницы формируется в теге `body`. Здесь расположен элемент `header` с логотипом, который выполняет роль верхней панели `topbar`. Ниже размещена боковая панель навигации `class="sidebar"`, содержащая логотип, кнопку для её разворачивания и список навигационных элементов – ссылки на главную страницу, мессенджер, расписание и выход из аккаунта. Визуальное оформление и интерактивное поведение панели реализуется с помощью JavaScript и CSS-классов.

Главное содержимое страницы представлено в блоке `main_content`. В нём отображается текущий учебный год с возможностью переключения недель с помощью стрелок по бокам от него. Также тут находится таблица расписания, которая строится динамически с помощью JavaScript и помещается в контейнер с `id="scheduleTable"`. Верхняя строка `tableHeader` содержит названия дней недели и даты, а последующие строки – временные интервалы и соответствующие события.

Функциональная логика расписания реализована через набор JavaScript-функций:

- getMonday – получает дату понедельника для недели, в которую входит переданная дата;
- formatDate – преобразует дату в строку формата ГГГГ-ММ-ДД, подходящую для сравнения и передачи на сервер;
- formatDisplayDate – форматирует дату для отображения в виде ДД.ММ.ГГ;
- updateYearLabel – обновляет строку с указанием учебного года и кнопками переключения недели;
- changeWeek – изменяет текущую неделю на указанное число недель вперёд или назад и обновляет расписание;
- openAddModal – открывает модальное окно для добавления события и сохраняет в его атрибутах дату и время выбранной ячейки;
- renderSchedule – ключевая функция, отвечающая за построение таблицы расписания. Она выполняет запрос к серверу, получает список событий и отображает их в таблице по дням недели и временным слотам. Если событие присутствует, отображается информация о нём с цветовой маркировкой по типу: лекции – зелёным, лабораторные – фиолетовым, практика – синим и экзамен – голубым. При отсутствии события в ячейке отображается кнопка «+», позволяющая вызвать модальное окно для добавления нового события;
- saveEvent – собирает данные из полей модального окна, проверяет обязательные поля и формирует объект события. Этот объект отправляется на сервер методом POST в формате JSON. После успешного сохранения окно закрывается, и таблица перерисовывается вызовом renderSchedule.

Нода «SQLite DB» при поступлении GET запроса обращается к БД и берёт оттуда таблицу events, которую отправляет обратно для отображения в расписании.

Нода «Ввод данных» формирует SQL-запрос на добавление в таблицу events информации о новом событии, полученную POST запросом.

Следующий ключевой элемент интерфейса – отображение информации о предмете. При нажатии на название предмета в расписании происходит

перенаправление на страницу предмета, где отображается дополнительная информация. На этой странице пользователю предоставляется доступ к описанию предмета, информации о преподавателе и набору лабораторных заданий. При загрузке этой страницы на клиенте выполняется серия запросов: один направлен на получение общей информации о предмете, другой – на извлечение всех лабораторных, связанных с данным предметом. Полученные данные используются для заполнения шаблона HTML, в котором отображаются карточки лабораторных работ, их статус, комментарии преподавателя и прогресс выполнения. Этот блок (рисунок 22) позволяет студенту отслеживать собственную успеваемость, просматривать задания и переходить к их детальному описанию и выполнению.

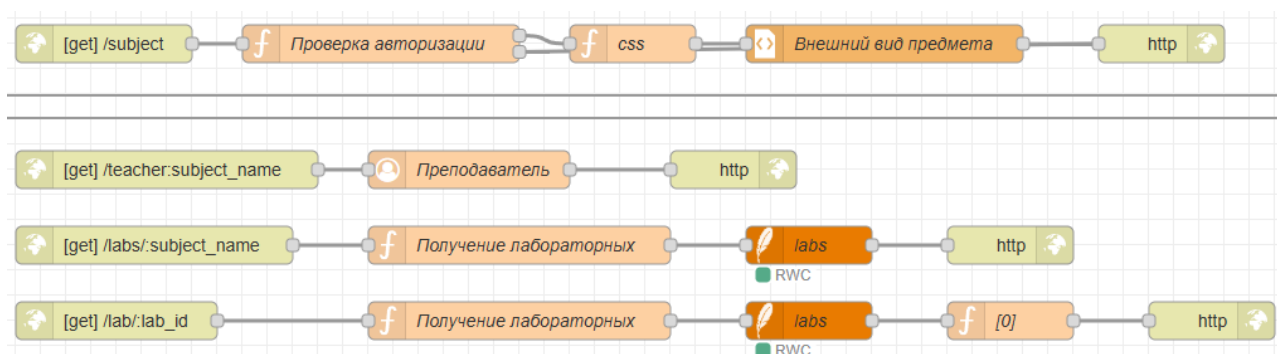


Рисунок 22 – Блок предмета

Важным компонентом архитектуры является модуль работы с базой данных, отвечающий за хранение, извлечение и обновление информации о событиях и лабораторных работах (рисунок 23). Он включает в себя потоки, реализующие CRUD-операции (создание, чтение, обновление и удаление) над таблицами базы. Для событий используются SQL-запросы, позволяющие получить все записи на определённую дату или интервал, а также добавить новое событие с указанными параметрами. Добавление нового занятия происходит через POST-запрос, отправляемый из модального окна страницы расписания, после чего данные записываются в таблицу базы. Аналогично реализована работа с лабораторными: система позволяет добавлять задания, редактировать их, прикреплять комментарии и изменять статус выполнения.

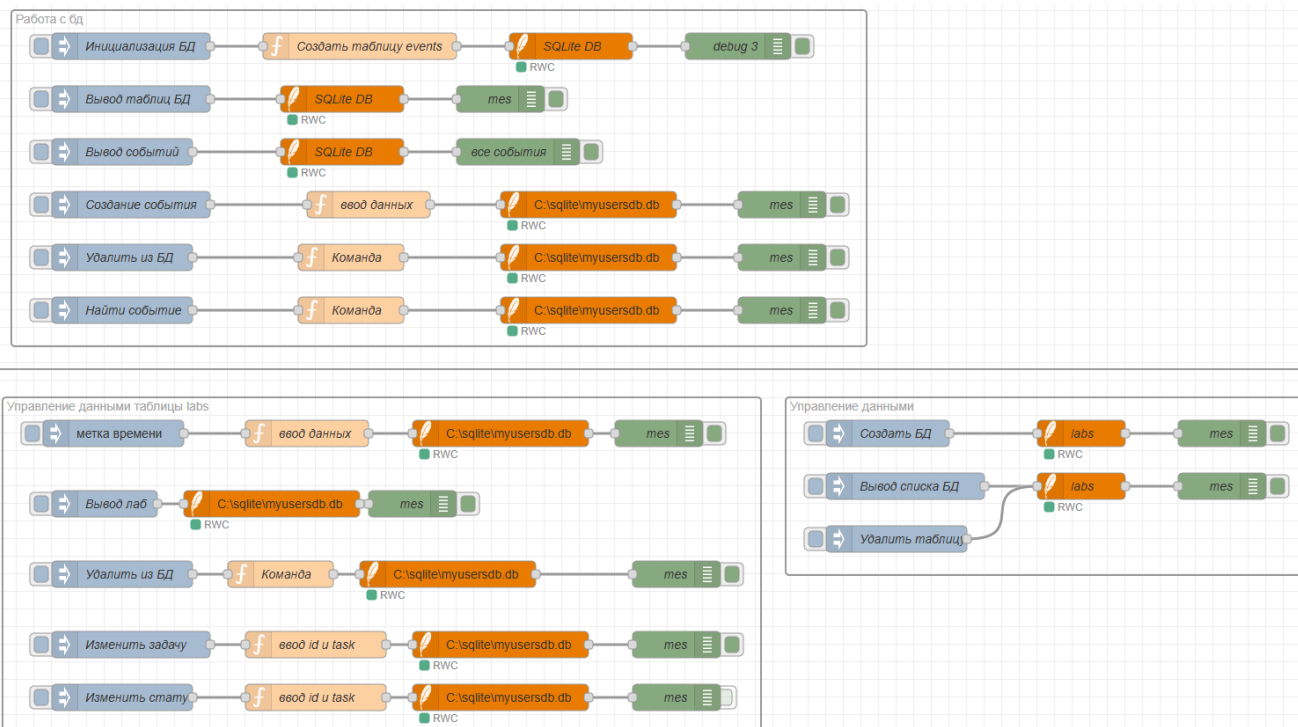


Рисунок 23 – Блок работы с базой данных

Таким образом, основной поток системы обеспечивает комплексное взаимодействие между пользователем и учебной информацией. Он объединяет фронтенд, построенный на HTML-шаблонах и JavaScript-логике, с серверной частью, реализованной в Node-Red, и обеспечивает устойчивое взаимодействие с базой данных. Такая организация позволяет создавать удобный, адаптивный и расширяемый интерфейс, где все ключевые данные доступны в несколько кликов, а их обработка и хранение остаются прозрачными и управляемыми.

2.3 Тестирование и анализ результатов

При первом запуске серверной части администратор получает доступ к потокам, реализующим работу приложения (рисунок 24). Слева в интерфейсе находится палитра функциональных блоков – нод, которые при необходимости можно перемещать на рабочую область для расширения функционала приложения.

Посередине находится рабочая область, на которой находятся ноды, которые организуют функционал приложения и реализуют управление данными. Справа находится панель, на которой отображается информация о выбранном блоке,

отладочные сообщения, документация по Node-Red и общая информация о проекте. Сверху видны потоки проекта, между которыми можно переключаться и которые можно выключать или активировать при необходимости. Кнопка «Развернуть» в правом верхнем углу позволяет запустить проект и сразу протестировать только что внесённые изменения.

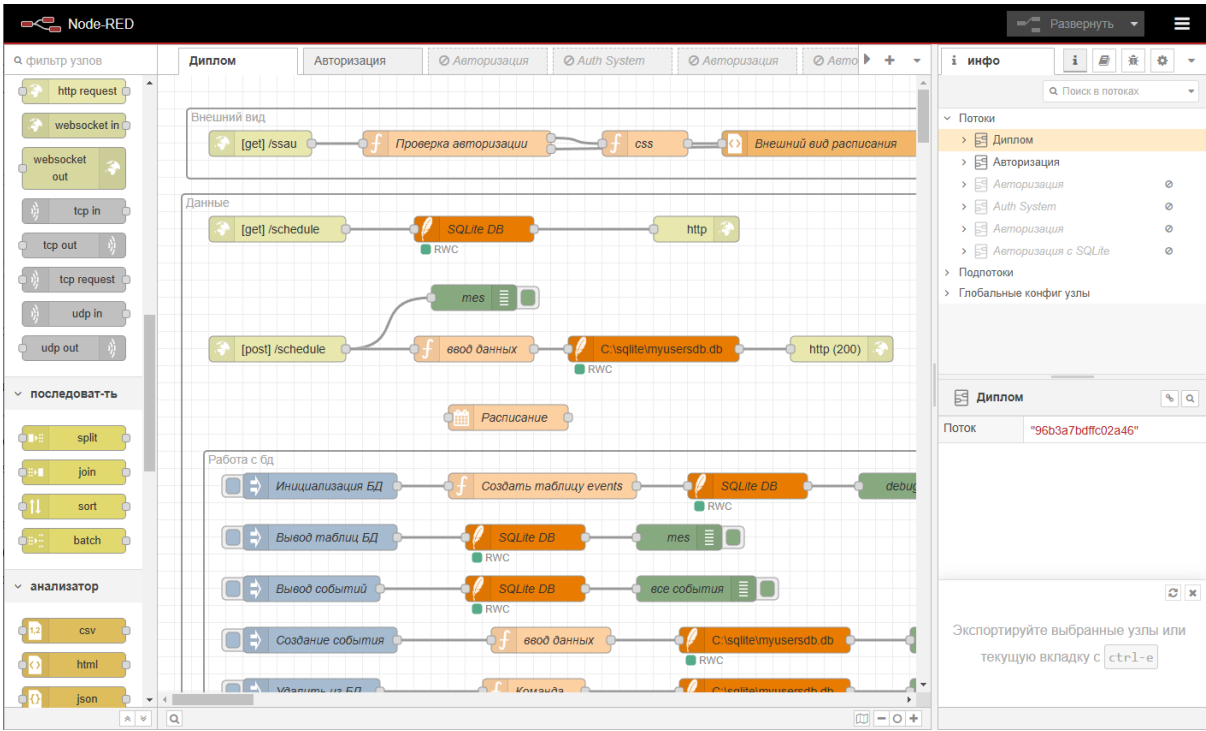


Рисунок 24 – Интерфейс серверной части

После запуска системы пользователь на клиентской стороне может перейти на веб-страницу, которая предоставит ему возможность использования всего функционала системы. В первую очередь пользователь попадёт на страницу авторизации, на которой ему будет предложено ввести свои логин и пароль (рисунок 25).

В случае, если пользователь ещё не обладает аккаунтом в системе и является новичком, на странице авторизации ему будет предложено зарегистрироваться. По нажатию на соответствующую кнопку пользователя перенаправит на страницу регистрации, где он сможет зарегистрироваться либо как преподаватель, либо как студент, в зависимости от его статуса (рисунок 26). В соответствии с выбранным вариантом при регистрации пользователю будет предложено заполнить разные поля, информация из которых пойдёт в его профиль.

Авторизация

Имя пользователя:

Пароль:

Авторизоваться

Нет аккаунта? [Зарегистрируйтесь](#)

Рисунок 25 – Интерфейс страницы авторизации

Зарегистрироваться

Как студент

Имя пользователя:

Пароль:

Группа:

Регистрация

Как преподаватель

Имя пользователя:

Пароль:

Регистрация

Уже есть аккаунт? [Войдите](#)

Рисунок 26 – Интерфейс страницы регистрации

После входа в систему пользователь по умолчанию попадает на страницу с расписанием (рисунок 27). Здесь он в зависимости от своей роли сможет наблюдать либо занятия, которые он ведёт как преподаватель, либо занятия, запланированные для его группы в случае, если он студент.

Также преподаватель сможет добавлять новые события в свободные места расписания. Для этого нужно нажать кнопку «+», которая появляется при наведении мыши на пустую ячейку таблицы (рисунок 28).

После нажатия кнопки добавления события на странице откроется новое окно, в которое пользователю будет предложено ввести данные о предстоящем событии (рисунок 29).

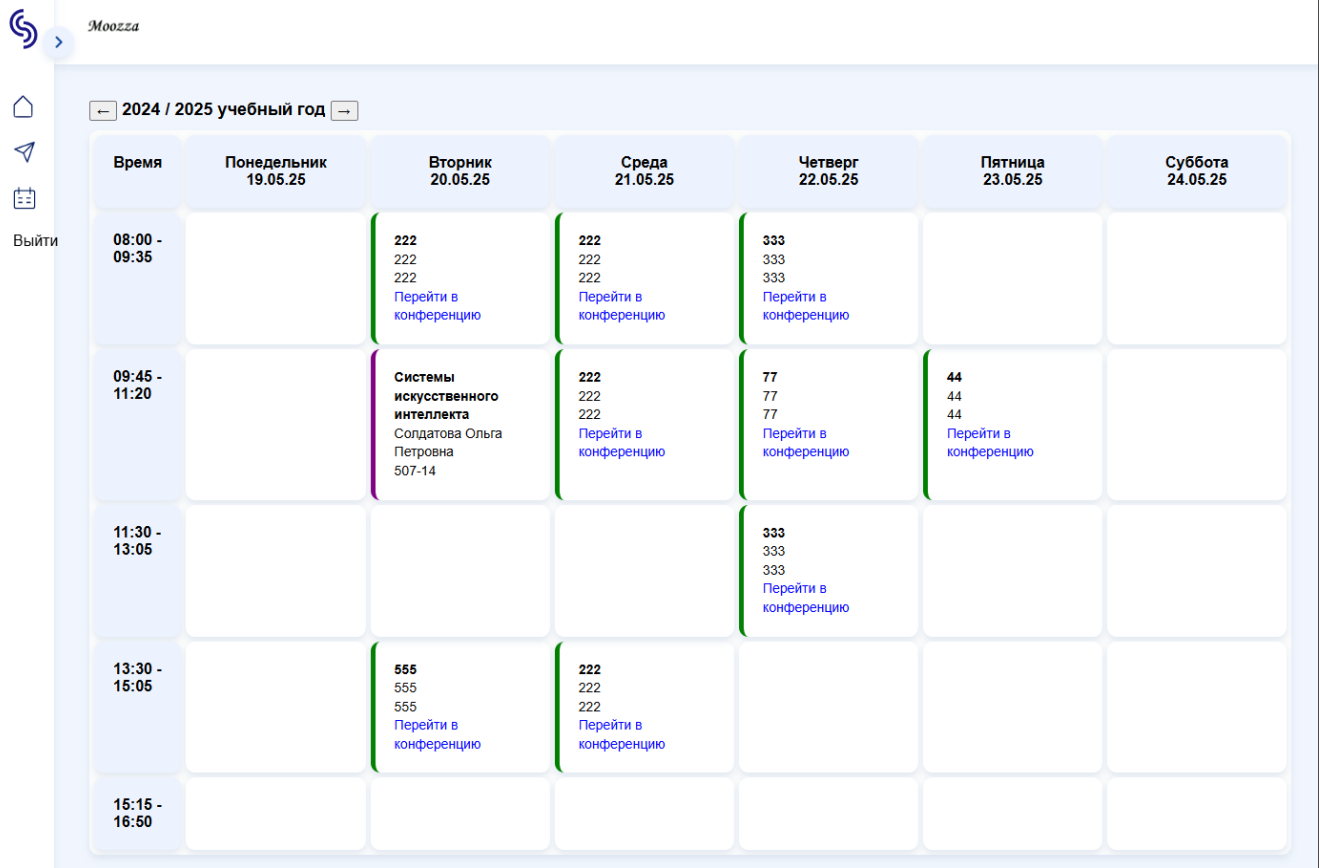


Рисунок 27 – Интерфейс страницы расписания

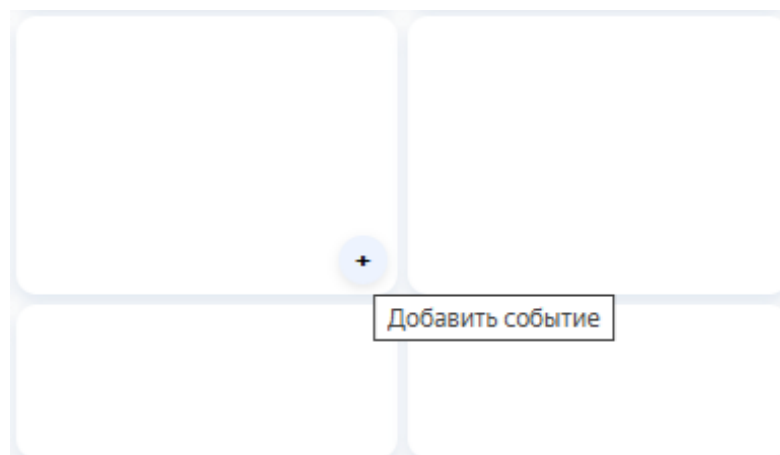


Рисунок 28 – Добавление нового события

Добавить событие

21BVB1	Системы искусственного ин	Солдатова Ольга Петровна	
507-14	Конференция	Платформа	Практика
Отмена			OK

Рисунок 29 – Окно с информацией о событии

В окне добавления события содержатся поля, в которые пользователь может ввести информацию о занятии, и кнопки «Ок» и «Отмена», по нажатию на которые соответственно произойдёт сохранение информации в таблицу и БД (рисунок 30), либо простое закрытие окна без влияния на данные.

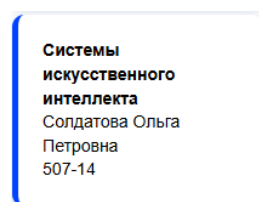


Рисунок 30 – Результат добавления записи в расписание

По нажатию на ячейку с событием пользователь попадёт на страницу, содержащую информацию о предмете, преподавателе и лабораторных работах (рисунок 31). Также на эту страницу можно попасть из списка предметов на странице с чатами.

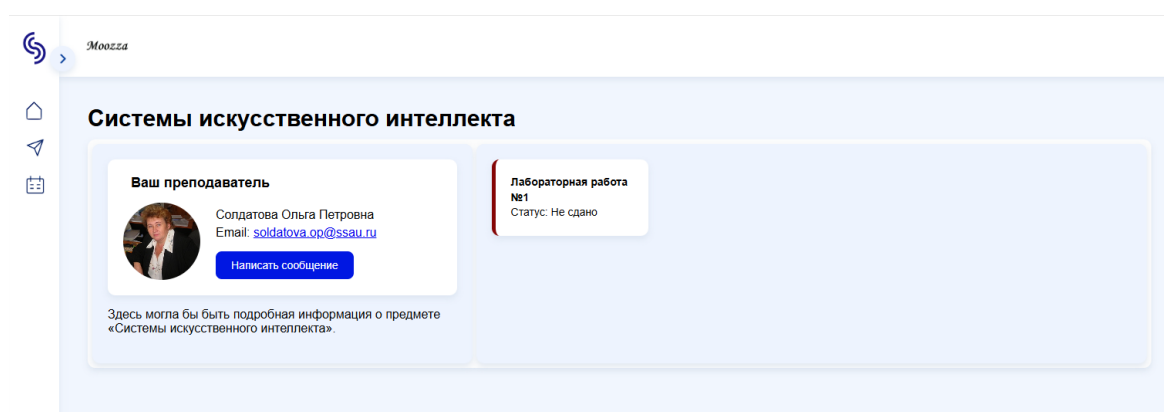


Рисунок 31 – Страница предмета

По нажатию на плитку лабораторной работы откроется подробная информация о задании на эту лабораторную работу, статусе выполнения и веб-интерфейс для компиляции кода (рисунок 32).

Ещё одной функцией системы, реализованной в рамках разработки системы, является список предметов и чатов по этим предметам отдельной страницей (рисунок 33). В этих чатах преподаватели могут оповещать студентов о каких-либо важных событиях или делиться информацией о лабораторных работах, а студенты могут спрашивать дополнительную информацию у преподавателя.

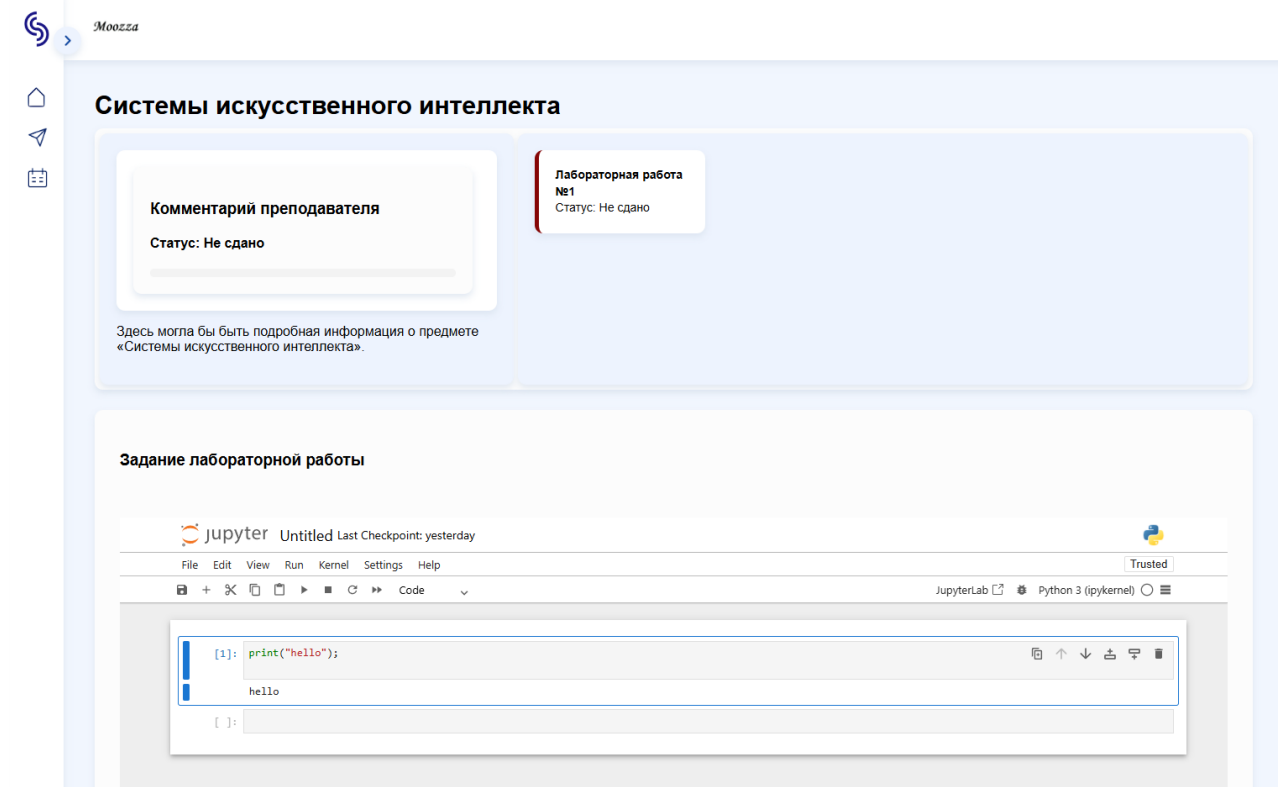


Рисунок 32 – Страница лабораторной работы

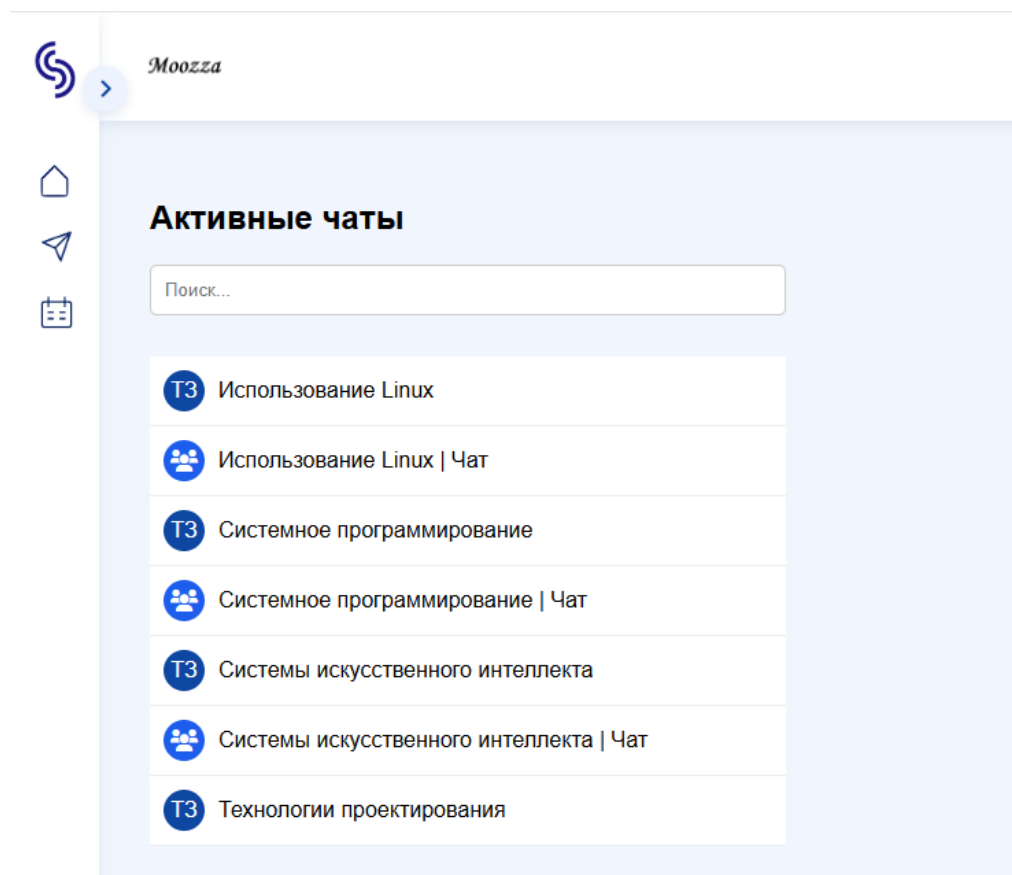


Рисунок 33 – Страница предметов и чатов

На странице реализован работающий поиск по чатам, поэтому даже в случае, если пользователь является участником большого количества чатов, он всегда сможет найти нужный (рисунок 34).

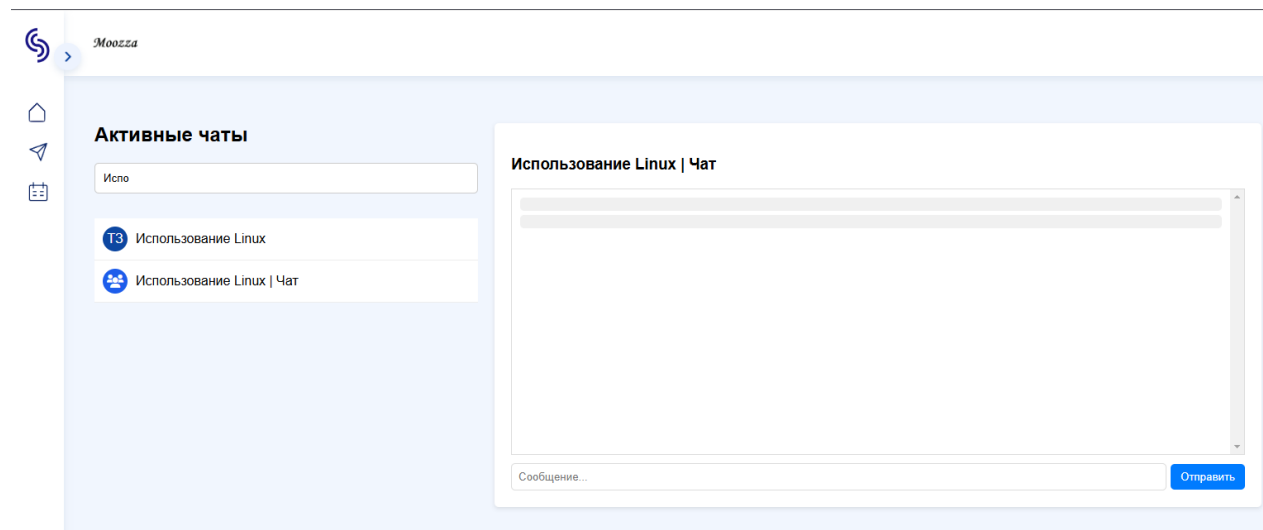


Рисунок 34 – Результат поиска

Таким образом, реализованная система выполнения практических и проектных заданий, ориентированная на использование в образовательных учреждениях, корпоративных обучающих системах и рабочих коллективах, работает исправно и реализует все поставленные задачи. Результаты тестирования функционала совпадают с ожидаемыми и позволяют пользователям эффективно пользоваться системой через интуитивно понятный интерфейс. Серверная часть приложения работает стабильно и позволяет легко модифицировать систему под нужды конкретной организации.

3 Охрана труда

Работа на персональном компьютере требует длительного нахождения в статическом положении и активного использования органов зрения, что может привести к развитию физического и психологического утомления. Даже при соблюдении всех норм эргономики рабочего места и правильной организации труда и отдыха оператор может испытывать зрительное и мышечное напряжение, а также эмоциональное перенапряжение, которые могут усиливаться, если не предпринимать никаких действий, связанных с минимизацией пагубных факторов. В связи с этим очень актуальными становятся профилактические меры, направленные на поддержание работоспособности и снижение утомляемости операторов, занятых на стадиях разработки и эксплуатации информационных систем.

В соответствии с ГОСТ 12.0.003-2015 «Система стандартов безопасности труда. Электронно-вычислительная техника. Общие требования безопасности», при эксплуатации ПВМ выделяются следующие основные риски:

- риски поражения электрическим током из-за: неисправной проводки; отсутствия заземления; попадания жидкости на технику; короткого замыкания при перегрузке сети.
- риски заболеваний опорно-двигательной системы из-за: неправильного положения тела; неправильной высоты стула и стола.
- риски заболеваний зрения из-за: использования ПК без перерывов для глаз; неправильной яркости экрана и расположения монитора.

В соответствии с обозначенными рисками формулируются требования к оборудованию рабочего места оператора и его освещённости, к пожарной, санитарной и электробезопасности, а также к комплексу профилактических мер, принимаемых работниками для снятия различных видов утомления и повышения работоспособности и двигательной активности.

Ключевым фактором профилактики утомляемости является правильная организация рабочего места с учётом эргономических требований и антропометрических особенностей работника. К основным требованиям, которые

должны соблюдаться при использовании оператором рабочего места, оборудованного ПЭВМ, относятся:

– рабочий стол должен обеспечивать размещение оборудования с учётом его размеров и характера выполняемой работы. Высота стола должна регулироваться в пределах 680-800 мм в зависимости от роста пользователя. При отсутствии регулировки она должна составлять 725 мм. Монитор должен располагаться ниже уровня глаз под углом 15-20°, а расстояние до глаз должно быть в районе 500-700 мм. Все элементы рабочего места должны быть расположены в зоне моторного поля оператора, исключая необходимость частых наклонов и поворотов тела. Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной не менее 500 мм и не менее 650 мм на уровне вытянутых ног;

– кресло должно обеспечивать поддержку спины, иметь регулируемую высоту сидения и углы наклона спинки, а также глубину сиденья в пределах 400-500 мм. Поверхность сиденья должна иметь закруглённый передний край. При выполнении работы в сидячем положении основная нагрузка падает на мышцы, поддерживающие позвоночник и голову, при этом основной вес тела передаётся на бёдра, что затрудняет кровоток в нижнюю часть туловища. Поэтому при длительном сидении оператору необходимо периодически менять свою позу. Для большинства людей комфортабельным может выступить рабочее место, которое позволяет приспособить его как минимум к двум позициям. Также внимание стоит уделить поясничному упору, который позволит расширить свободу движений в грудном сочленении и несколько снизить негативное влияние смены естественного спинно-поясничного прогиба на прогиб назад, приводящей к болям в пояснице. Для снижения статического напряжения мышц рук необходимо иметь стационарные или съёмные подлокотники длиной не менее 250 мм и имеющие регулировку по высоте над сиденьем. Поверхность сиденья, спинки и других элементов кресла должна быть полумягкой, с нескользящим, неэлектризующимся и воздухопроницаемым покрытием, обеспечивающим легкую очистку от загрязнений;

– клавиатуру рекомендуется размещать на отдельной, регулируемой по высоте поверхности или с использованием специальной подставки под запястья

					ПГУ 1.090301.03.001 ПЗ	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		

для уменьшения воздействия на мышцы рук и кистей, и обеспечения комфортных условий работы. Расстояние от переднего края стола до клавиатуры должно составлять 100-130 мм, чтобы запястья имели опору и находились в физиологически правильном положении. Клавиатура должна быть достаточно плоской и иметь опорное приспособление, позволяющее изменять угол наклона её поверхности в пределах 5-15°. Кресло и клавиатура должны быть размещены так, чтобы не приходилось тянуться. При изменении положения тела обязательно нужно сменить расположение клавиатуры.

Рабочее место должно обеспечивать удобное положение при работе за ПК в положении сидя. Должны выполняться следующие условия: прямая спина с упором на спинку кресла, расслабленные плечи, устойчивое положение ног на полу, прямое положение кистей и запястий. В процессе длительной работы в вынужденной позе может возникнуть позотоническое утомление. Для его предотвращения стоит применять комплекс специальных физических упражнений, смысл которых заключается в снятии излишнего напряжения путём попеременного сокращения и напряжения мышц, что в свою очередь приводит к перераспределению мышечных усилий, улучшению кровообращения, предотвращению застойных явлений и отёчности в нижних конечностях. Далее приведено несколько вариантов возможных упражнений для снятия костно-мышечного утомления с плечевого пояса, рук, ног и туловища. Исходное положение: сидя на стуле, отодвинутом от стола, ноги, согнутые в коленях, поставить на пол.

Первым упражнением выступит потягивание. Действие первое: руки – за голову, пальцы переплетены, прогнуться, голову наклонить назад, напрячь все мышцы тела – вдох. Действие второе: слегка наклониться вперёд и расслабить мышцы – выдох. Повторить 4-5 раз в среднем темпе.

Второе упражнение – поднимание согнутой ноги. Первым действием руки согнуть к плечам, ноги выпрямить и поставить пятками на пол – вдох. Вторым действием одну из ног, руками обхватить голень и подтянуть к груди, напрячь мышцы живота – выдох. Повторить 3-4 раза в медленном темпе.

					ПГУ 1.090301.03.001 ПЗ	Лист
						51
Изм.	Лист	№ докум.	Подпись	Дата		

Третьим упражнением выступит наклоны туловища с поворотом. Действие первое: руки – к плечам, наклониться вперёд, правым локтём коснуться колена левой ноги либо левым локтём коснуться колена правой ноги – выдох. Действие второе: вернуться в исходное положение – вдох. Повторить 4-5 раз в медленном темпе.

Четвёртое упражнение – перегруппировка веса тела. Первым действием перегруппировать вес тела на правую или левую часть – выдох. Вторым действием вернуться в исходное положение – вдох. Повторить 3-4 раза в медленном темпе и с глубоким дыханием.

Отдельно стоит упомянуть базовые упражнения любой гимнастики – круговые движения плечами назад и вперёд, повороты и наклоны головы, вертикальные и горизонтальные сведения и разведения рук. Все эти упражнения способствуют улучшению кровообращения, в том числе и мозговое, что позитивно сказывается на работоспособности и интенсивности действий оператора.

Для обеспечения здоровья органов зрения на рабочем месте должны соблюдаться стандарты освещения, которые, согласно санитарным правилам, составляют величину освещённости в не менее 400 люкс. Источники освещения должны обеспечивать равномерный рассеянный свет без бликов на мониторе. Недопустимо применение мерцающих источников света либо ярких нерассеянных источников света в поле зрения оператора. Цветовая температура источников должна составлять 4000-5000 К, что соответствует белому цвету.

Наиболее распространённым недомоганием у операторов ПЭВМ является зрительное утомление, которое при отсутствии надлежащих мер и при продолжении рабочей деятельности может проявляться в виде частого моргания, зуда, жжения или рези в глазах, обильного слёзотечения и др. Однако даже самое простое упражнение, не требующее ничего, кроме накрытия глаз ладонями, может дать глазам тот необходимый им отдых. Для его выполнения нужно пальцы рук, сложенные вместе, скрестить в центре лба, при этом ладони накроют глазные впадины, полностью исключая доступ света, и в то же время не будут давить на глазные яблоки, оставляя возможность свободно двигать веками. В процессе такого отдыха происходит химическое восстановление рецепторов глаз и

					ПГУ 1.090301.03.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		52

расслабление мышечных волокон. Подобное упражнение, называемое пальминг, является одним из лучших упражнений для глаз, значительно ускоряя процесс расслабления глазных мышц и улучшая кровообращение.

В качестве профилактических мер для снижения утомления глаз, улучшения кровоснабжения глазного яблока, релаксации глазодвигательных мышц рекомендуется проводить упражнение для глаз во время регламентированных перерывов вместе с другими комплексами физических упражнений каждые 20-30 минут работы за ПЭВМ. При появлении зрительного дискомфорта эти упражнения следует проводить индивидуально в соответствии со своими запросами по периодичности. При возникновении значительного зрительного дискомфорта в процессе работы, несмотря на хорошее качество монитора, освещения и соблюдение эргономических и режимных требований, следует ограничить время работы с дисплеем. Для этого требуется либо увеличить время перерыва для отдыха, либо сменить деятельность на другую, не связанную с работой с ЭВМ. Далее приведено несколько вариантов возможных упражнений для глаз, призванных снизить утомляемость и риски возможных заболеваний. Упражнения выполняются стоя или сидя, отвернувшись от монитора.

Первый комплекс упражнений заключается в следующей последовательности действий. Действие первое: закрыть глаза, сильно напрягая глазные мышцы на счёт от 1 до 4, далее раскрыть глаза, расслабив мышцы, и посмотреть вдаль на счёт от 1 до 6. Повторить 4-5 раз. Действие второе: посмотреть на переносицу и задержать взор на счёт от 1 до 4, затем открыть глаза и посмотреть вдаль на счёт от 1 до 6. Повторить 4-5 раз. Действие третье: не поворачивая головы посмотреть направо и зафиксировать взгляд на счёт от 1 до 4, затем посмотреть вдаль на счёт от 1 до 6. Аналогично выполняются действия для оставшихся 3 направлений движения глаз: вверх, налево и вниз. Повторить 3-4 раза.

Второй комплекс упражнений – следующая последовательность действий. Первым действием закрыть глаза, не напрягая глазные мышцы, на счёт от 1 до 4, потом раскрыть глаза и посмотреть вдаль на счёт от 1 до 6. Повторить 4-5 раз. Вторым действием посмотреть на кончик носа на счёт от 1 до 4 от 1 до 4, затем перевести взгляд вдаль на счёт от 1 до 6. Повторить 4-5 раз. Третьим действием,

					ПГУ 1.090301.03.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		53

не поворачивая головы, медленно делать круговые движения глазами вверх-влево-вниз-вправо и в обратную сторону, затем посмотреть вдаль на счёт от 1 до 6. Повторить 4-5 раз.

Одной из важных категорий рисков является пожарная безопасность. Они связаны с возможными перегревом оборудования, короткими замыканиями и возгоранием компонентов. Для предотвращения возгораний следует:

- регулярно проверять состояние электросети;
- располагать оборудование на расстоянии не менее 0,5 м от отопительных приборов;
- не загромождать проходы и выходы;
- оборудовать помещения средствами пожаротушения;
- установить автоматическую пожарную сигнализацию.

Санитарно-гигиенические параметры рабочего помещения регламентируются СанПиН 1.2.3685-21. Основные требования включают:

- температуру воздуха 22-25 °С;
- влажность около 60 %;
- скорость движения воздуха не более 0,1-0,2 м/с;
- уровень шума не выше 65 дБ;
- общую освещенность не менее 400 лк с равномерным, рассеянным светом.

В помещениях должны быть предусмотрены доступ к питьевой воде, аптечка первой помощи, огнетушители (1 шт. на 50 м³ помещения). Рабочие столы должны находиться не далее 6 м от окон, обеспечивающих естественное освещение.

Все сотрудники должны проходить инструктажи по технике безопасности не реже 1 раза в год.

Заключение

В процессе выполнения выпускной квалификационной работы была реализована цель по созданию информационной системы выполнения практических и проектных заданий, ориентированной на использование в образовательной и корпоративной среде. В ходе работы был произведён анализ требований к подобным системам, определена архитектура программного решения, а также разработан прототип веб-приложения, включающий в себя ключевые функциональные модули.

Были решены следующие задачи: спроектирован и реализован веб-интерфейс, обеспечивающий удобство взаимодействия пользователей с различным уровнем подготовки; создан механизм управления расписанием и сроками выполнения заданий; реализована возможность добавления и выполнения лабораторных и проектных заданий в интерактивной среде; разработана система ролевого доступа с разграничением прав и возможностей пользователей, внедрён модуль оценки выполненных заданий.

Разработанный прототип был протестирован на сценариях, имитирующих реальные процессы образовательных и корпоративных учреждений, что позволило выявить его применимость в различных условиях, включая возможность развёртывания в изолированной локальной сети предприятия. Также была произведена оценка масштабируемости системы и её потенциальной интеграции с внешними сервисами, подтверждая перспективы дальнейшего развития проекта.

В результате выполнения работы были получены практические навыки проектирования и реализации веб-систем, а также выработан подход к созданию универсального инструмента поддержки практической деятельности в условиях цифровизации. Разработанная система может служить основой для внедрения в учебный процесс образовательных учреждений и для автоматизации внутренних процессов в корпоративной среде. В дальнейшем возможна её доработка с целью расширения функциональности, повышения производительности и интеграции с современными платформами дистанционного обучения.

					ПГУ 1.090301.03.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		55

Список использованных источников

1. Система электронного обучения и тестирования Moodle: обзор возможностей. [Электронный ресурс] – URL: <https://www.ispring.ru/elearning-insights/moodle>. (дата обращения 25.04.2025).
2. Google Classroom: обзор возможностей. [Электронный ресурс] – URL: <https://www.ispring.ru/elearning-insights/platforma-onlain-obucheniya/google-classroom>. (дата обращения 23.04.2025).
3. Гугл Класс (Google Classroom): преимущества и возможности сервиса. [Электронный ресурс] – URL: <https://e-queo.com/blog/expertnie-stati/google-classroom/>. (дата обращения 23.04.2025).
4. Обзор возможностей Google Classroom. [Электронный ресурс] – URL: <https://skillspace.ru/blog/google-classroom-obzor-vozmozhnostej/>. (дата обращения 23.04.2025).
5. Stepik. [Электронный ресурс] – URL: <https://ru.ruwiki.ru/wiki/Stepik>. (дата обращения 21.04.2025).
6. Stepik — российская образовательная платформа. [Электронный ресурс] – URL: <https://infourok.ru/stepik-rossiyskaya-obrazovatel'naya-platforma-3771215.html>. (дата обращения 21.04.2025).
7. Курсы онлайн-школы Stepik. [Электронный ресурс] – URL: <https://education.pikabu.ru/schools/stepik>. (дата обращения 21.04.2025).
8. Что такое Trello? Подробный гайд по использованию сервиса. [Электронный ресурс] – URL: <https://skillbox.ru/media/management/chto-takoe-trello-podrobnyy-gayd-po-ispolzovaniyu-servisa/>. (дата обращения 17.04.2025).
9. Обзор Trello: возможности и функции. [Электронный ресурс] – URL: <https://sky.pro/wiki/profession/obzor-trello-vozmozhnosti-i-funkcii/>. (дата обращения 17.04.2025).
10. Что такое Trello? Подробный гайд по использованию сервиса. [Электронный ресурс] – URL: <https://skillbox.ru/media/management/chto-takoe-trello-podrobnyy-gayd-po-ispolzovaniyu-servisa/>. (дата обращения 17.04.2025).

11. Знакомство с Node-RED и потоковое программирование в Yandex IoT Core. [Электронный ресурс] – URL: <https://habr.com/ru/companies/yandex/articles/519600/>. (дата обращения 27.04.2025).

12. Начало работы с Node-RED на Raspberry Pi. [Электронный ресурс] – URL: <https://microkontroller.ru/raspberry-pi-projects/nachalo-raboty-s-node-red-na-raspberry-pi/>. (дата обращения 27.04.2025).

13. Что такое Apache NiFi. [Электронный ресурс] – URL: <https://gitverse.ru/blog/articles/architecture/248-chto-takoe-apache-nifi>. (дата обращения 29.04.2025).

14. Apache NiFi для новичков: обзор функций, принцип работы, полезные ссылки. [Электронный ресурс] – URL: <https://blog.skillfactory.ru/apache-nifi-dlya-novichkov-obzor/>. (дата обращения 29.04.2025).

15. Обзор n8n: конструктор бэкенда, автоматизаций и интеграций. [Электронный ресурс] – URL: <https://ya.zerocoder.ru/obzor-n8n-konstruktor-bekienda-avtomatizatsii-i-intieghratsii/>. (дата обращения 01.05.2025).

16. n8n. [Электронный ресурс] – URL: https://hostkey.ru/documentation/marketplace/business_apps/n8n/. (дата обращения 01.05.2025).

17. Мощный инструмент автоматизации рабочих процессов для вашего проекта. [Электронный ресурс] – URL: <https://cod-reg.ru/posts/n8n-moshniy-instrument-avtomatizacii-procesov/>. (дата обращения 01.05.2025).

18. Zapier Введение. [Электронный ресурс] – URL: <https://aipure.ai/ru/products/zapier/introduction>. (дата обращения 03.05.2025).

19. Zapier и его аналоги: что выбрать для автоматизации рутины. [Электронный ресурс] – URL: <https://vc.ru/services/144704-zapier-i-ego-analogi-chto-vybrat-dlya-avtomatizacii-rutiny>. (дата обращения 03.05.2025).

20. Tibco Flogo – An Ultra Light, web based, Tibco cloud Integration Platform. [Электронный ресурс] – URL: <https://www.devoteam.com/expert-view/tibco-flogo-an-ultra-light-web-based-tibco-cloud-integration-platform/>. (дата обращения 05.05.2025).

21. TIBCO Forays Into Edge And Serverless Computing With Project Flogo. [Электронный ресурс] – URL: <https://www.forbes.com/sites/janakirammsv/2017/11/13/tibco-forays-into-edge-and-serverless-computing-with-project-flogo/>. (дата обращения 05.05.2025).
22. TIBCO Introduces Project Flogo. [Электронный ресурс] – URL: <https://www.bigdatawire.com/this-just-in/tibco-introduces-project-flogo/>. (дата обращения 05.05.2025).
23. Running Node-RED locally. [Электронный ресурс] – URL: <https://nodered.org/docs/getting-started/local>. (дата обращения 19.04.2025).
24. P. Degreeef, D. Van Merode, G. Tabunshchuk. Low-Cost, Open-Source Automation System for Education, with Node-RED and Raspberry Pi // Smart Industry & Smart Education. REV 2018. – 2019. – Т. 47. – С. 458– 465.
25. F. Pallas, J. Legler, N. Amslgruber, E. Grünewald. RedCASTLE: Practically Applicable $\$k_s\$$ -Anonymity for IoT Streaming Data at the Edge in Node-RED // arXiv preprint arXiv:2110.15650. – 2021.
26. R. Sosa, C. Kiraly, J. D. Parra Rodriguez. Offloading Execution from Edge to Cloud: a Dynamic Node-RED Based Approach // arXiv preprint arXiv:1810.11287. – 2018.
27. A. Bröring, V. Charpenay, D. Anicic, S. Püech. NOVA: A Knowledge Base for the Node-RED IoT Ecosystem // The Semantic Web: ESWC 2019 Satellite Events. – 2019. – Т. 11762. – С. 257– 261.
28. node-red-nodes. [Электронный ресурс] – URL: <https://github.com/node-red/node-red-nodes>. (дата обращения 19.04.2025).
29. S. Schneider, K. Kashish, K. Tuma, R. Scandariato. In Specs we Trust? Conformance-Analysis of Implementation to Specifications in Node-RED and Associated Security Risks // arXiv preprint arXiv:2502.09117. – 2025.
30. N. Varghese, R. Sinha. Can Commercial Testing Automation Tools Work for IoT? A Case Study of Selenium and Node-Red // arXiv preprint arXiv:287.04246. – 2021.
31. D. Clerissi, M. Leotta, F. Ricca. Towards an approach for developing and testing Node-RED IoT systems // Proceedings of EnSEmble@ESEC/SIGSOFT 2018. – 2018. – С. 1– 8.

32. Node-RED Low-code programming for event-driven applications. [Электронный ресурс] – URL: <https://www.node-red.dev/>. (дата обращения 27.04.2025).

33. A. S. Thuluva et al. Semantic Node-RED for rapid development of interoperable industrial IoT applications // ResearchGate. – 2020.

34. J. M. Gutierrez-Guerrero, J. A. Holgado-Terriza. Supervisory Control and Data Acquisition Approach in Node-RED: Application and Discussions // Electronics. – 2019. – Т. 8, № 600.

35. T. Hagino. Practical Node-RED Programming. – Packt Publishing, 2021. – 326 с.

Приложение А

(обязательное)

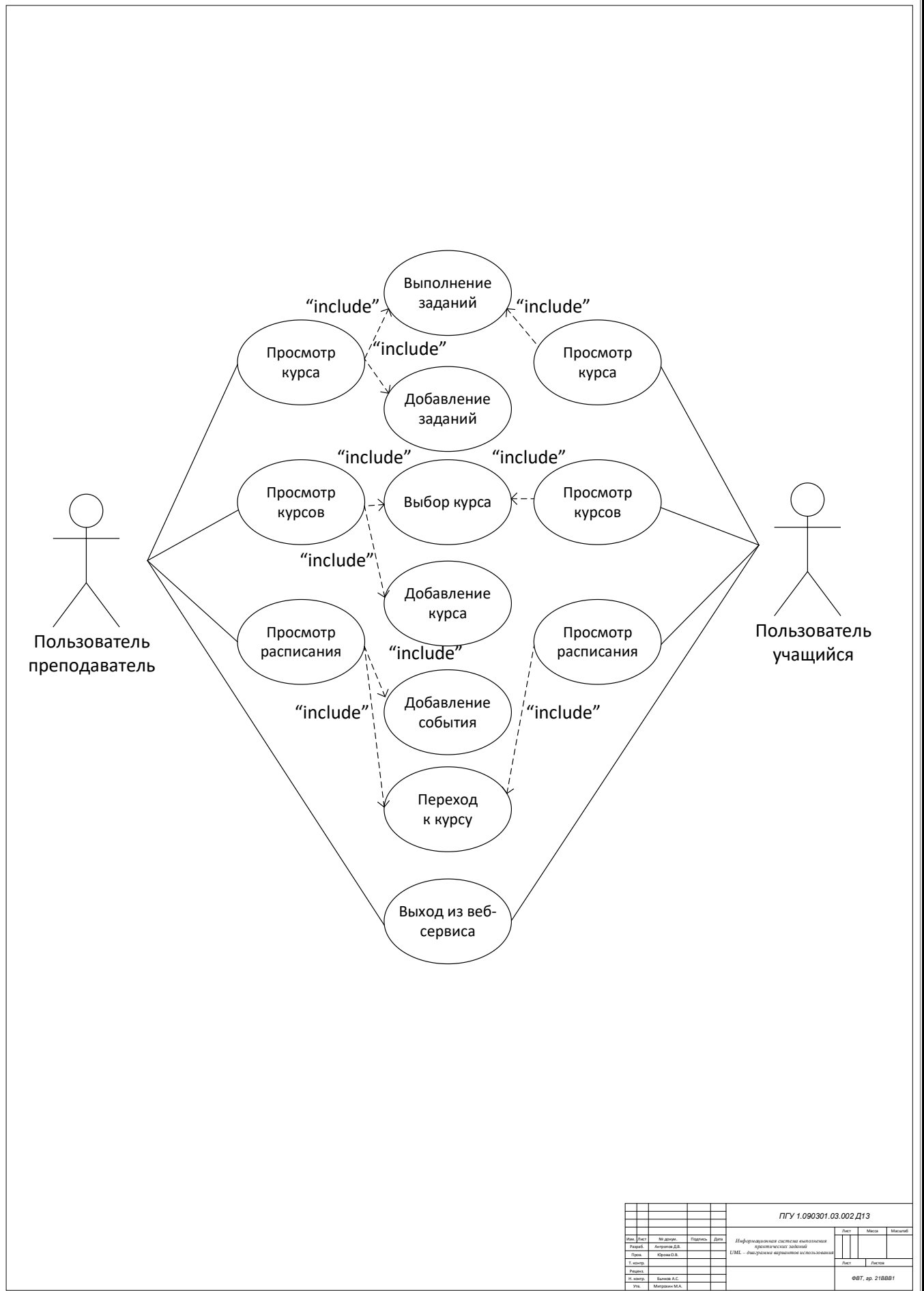
Программа моделирования интеллектуального управления дорожным

трафиком

UML – диаграммы

					ПГУ 1.090301.03.001 ПЗ	Лист
						60
Изм.	Лист	№ докум.	Подпись	Дата		

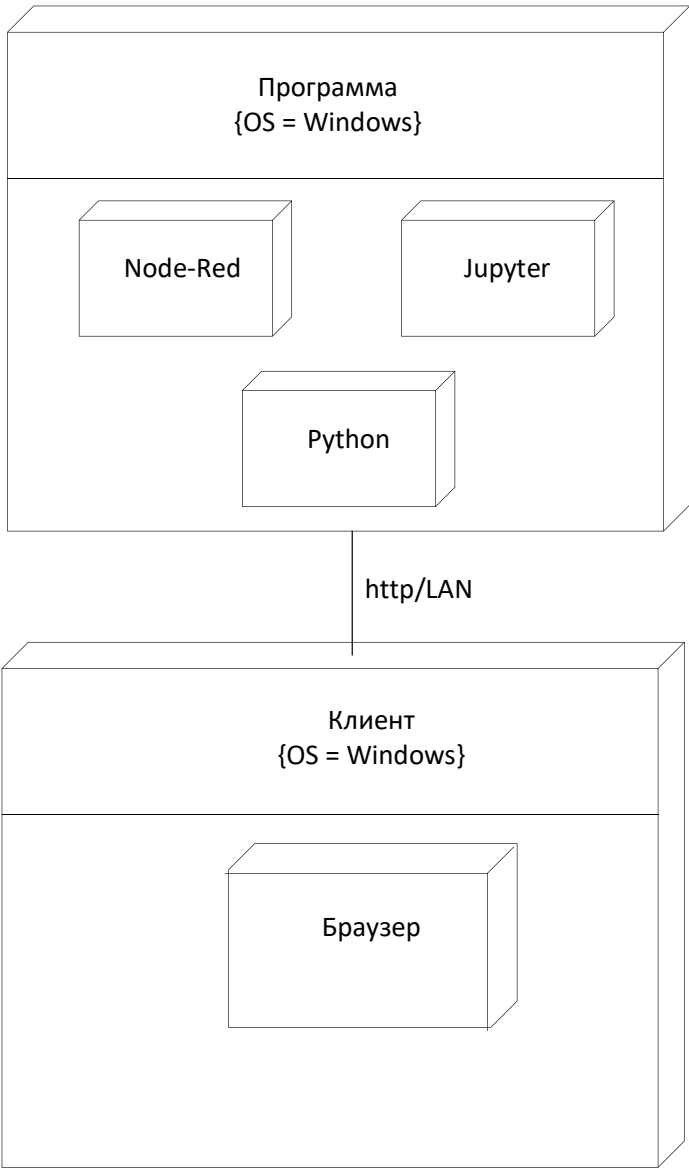
Приложение А.1 – Диаграмма вариантов использования



					ПГУ 1.090301.03.002.D13				
Имя	Лист	№ докум.	Подпись	Дата	Информационная система выполнения практических заданий ИМЛ – дистанция интерактивное использование				
Разработ.	Андреев Д.В.								
Тестиров.	Куркин О.В.								
Разработ.	Виноков А.С.								
И. номер	Уч.	Материал М.А.							

Изм.	Лист	№ докум.	Подпись	Дата

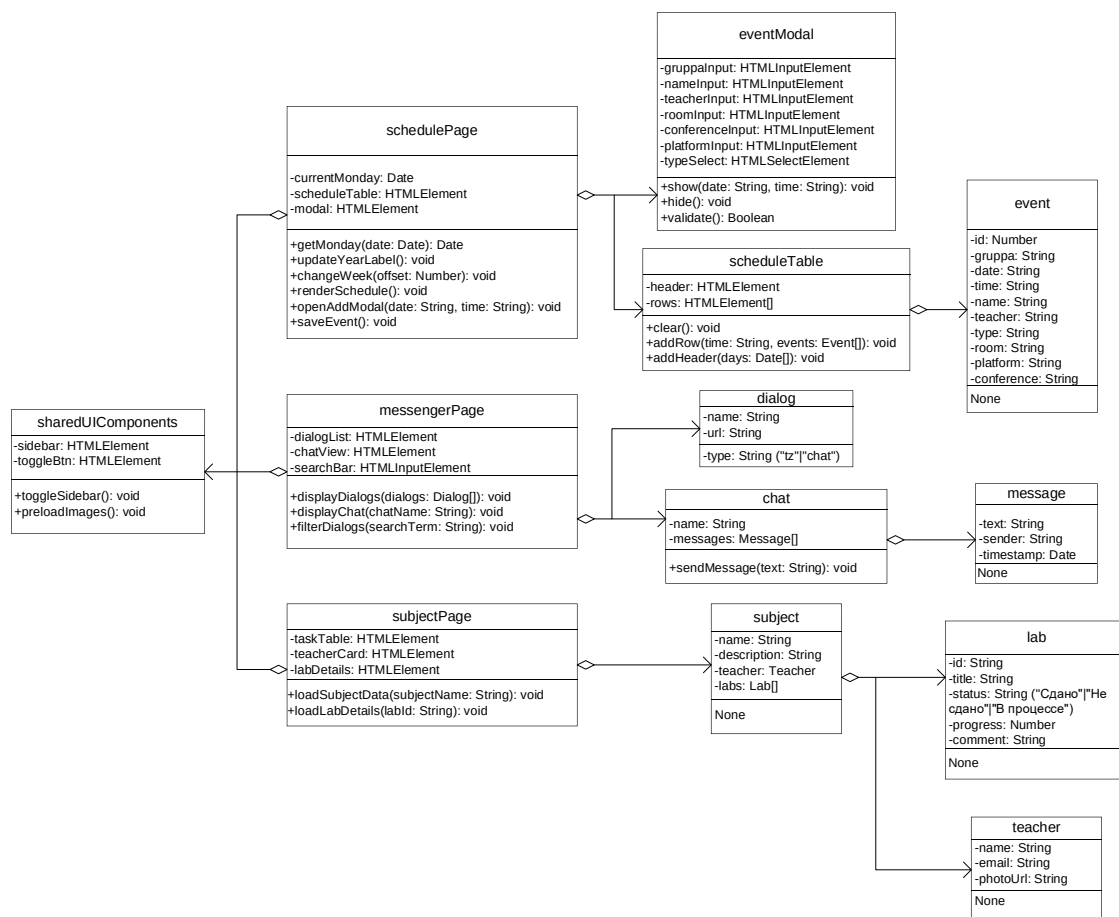
Приложение А.2 – Диаграмма развёртывания



					ПГУ 1.090301.03.002 Д19		
Изм.	Лист	№ докум.	Подпись	Дата	Информационная система выполнения практических заданий УМК «...Диагностика развёртывания»		
Разраб.	Андреев Д.М.						
Лист	Страна О.В.						
Т. номер							
Разраб.	Васильев А.С.						
И. номер					ФБГ, стр. 2188881		
Уч.	Масловский М.А.						

Изм.	Лист	№ докум.	Подпись	Дата

Приложение А.3 – Диаграмма классов



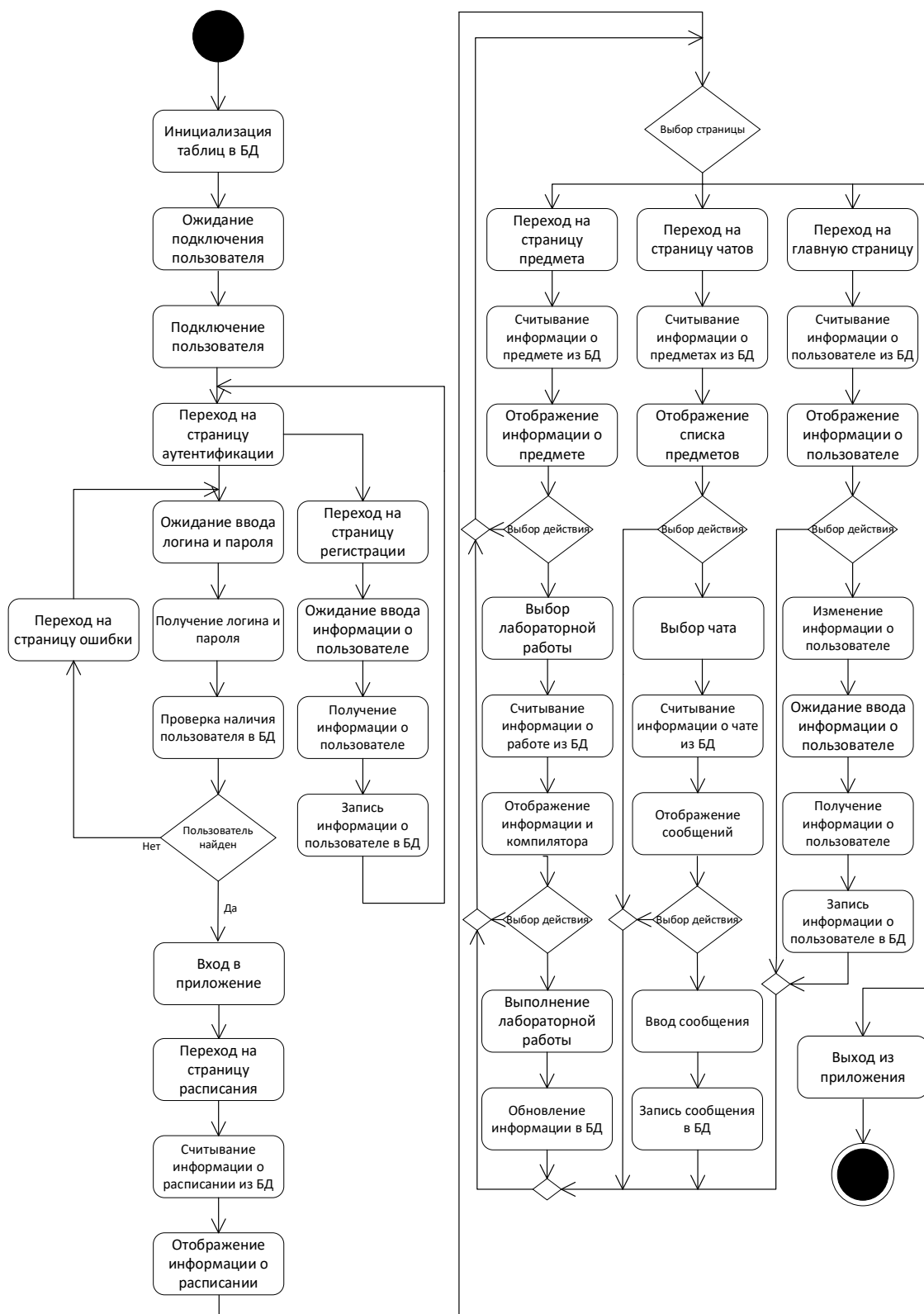
ПГУ 1.090301.03.002 Д11					
Имя	Лист	№ докум.	Подпись	Дата	Лист
Резаб.	Андреев Д.В.				Место
Тема	Курсовая Д.В.				Масштаб
Т. номер					Лист
Резаб.	Белов А.С.				Лист
И. номер	Михайлов М.А.				Лист
Уч.					Лист

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.03.001 ПЗ

Лист
63

Приложение А.4 – Диаграмма деятельности



						ПГУ 1.090301.03.002 Д17			
						Информационная система выполнения практических заданий УМБ - Дидактические материалы			
Имя	Лист	№ докум.	Подпись	Дата		Лист	Место	Масштаб	
Разработчик		Антонов Д.В.							
Проверен		Куркин С.В.							
Т. номер						Лист	Место	Масштаб	
Разработчик		Куркин С.В.							
И. номер		Бачинин А.С.							
		Бачинин А.С.							

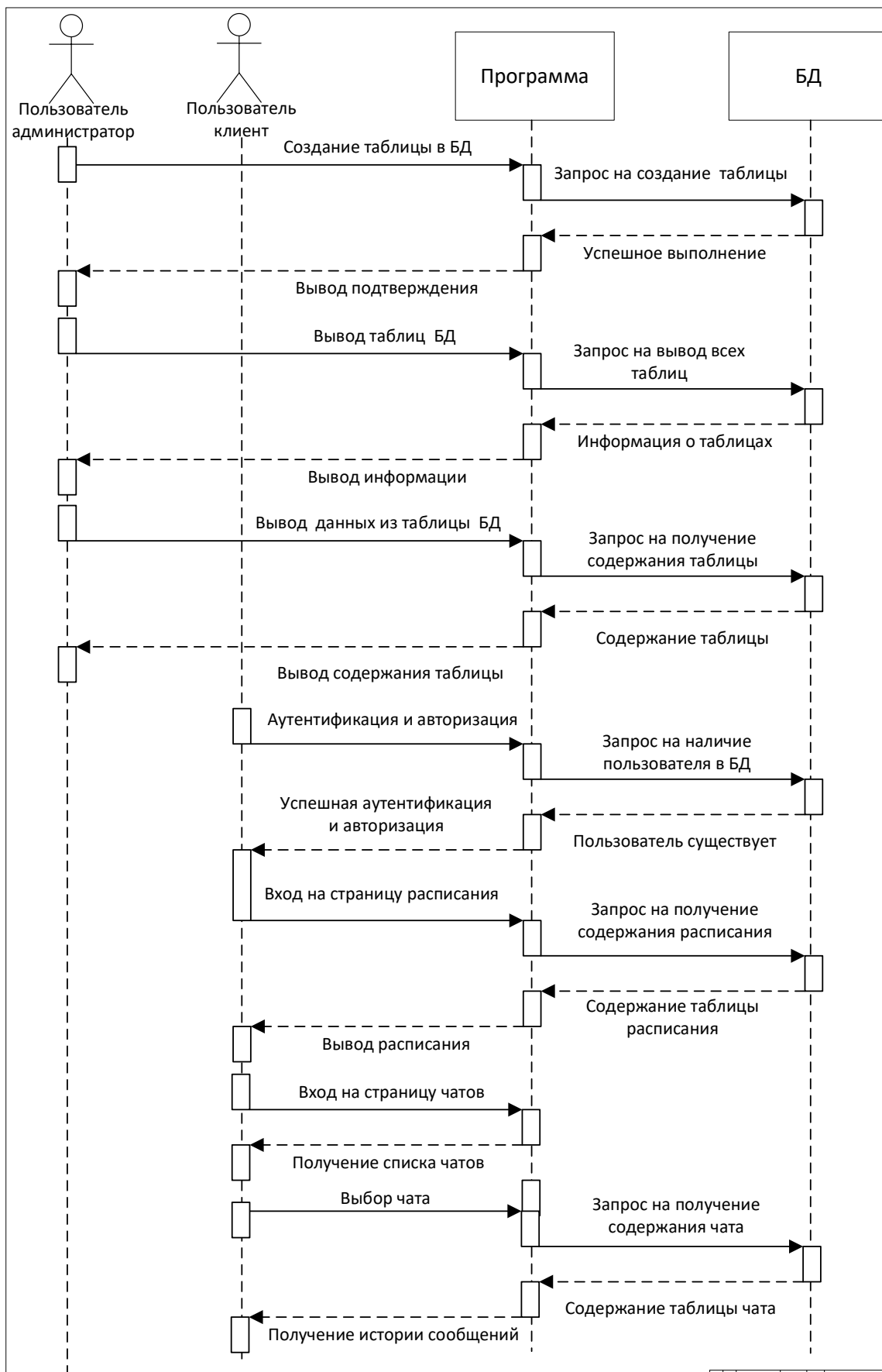
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.03.001 ПЗ

Лист

64

Приложение А.3 – Диаграмма последовательности

[illegible]

Приложение Б

(обязательное)

Программа выполнения практических заданий

Листинг программы

					ПГУ 1.090301.03.001 ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дата		

Файл «shedule.html»

```
<!DOCTYPE html>
<html lang="ru">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Система выполнения заданий</title>
  <style>
    {{{cssText}}}
  </style>
  <link
                                rel="icon"
                                type="image/png"
href="https://cabinet.ssau.ru/asset/favicon.ico">
</head>

<div class="modal" id="eventModal">
  <h3>Добавить событие</h3>
  <input type="text" id="eventGruppa" placeholder="Группа">
  <input type="text" id="eventName" placeholder="Предмет">
  <input
                                type="text"
                                id="eventTeacher"
placeholder="Преподаватель">
  <input type="text" id="eventRoom" placeholder="Аудитория">
  <input
                                type="text"
                                id="eventConference"
placeholder="Конференция">
  <input type="text" id="eventPlatform" placeholder="Платформа">
  <select id="eventType">
    <option value="Лекция">Лекция</option>
    <option value="Лабораторная">Лабораторная</option>
    <option value="Практика">Практика</option>
    <option value="Экзамен">Экзамен</option>
  </select>
  <button onclick="saveEvent()">OK</button>
  <button onclick="closeModal()">Отмена</button>
</div>
<body>
  <header class="topbar">
    <div
      class="divlogoname">
    </div>
  </header>

  <div class="sidebar" id="sidebar">
    <div class="logocenter" id="logocenter">

    </div>
  </div>
</body>
</html>
```

					ПГУ 1.090301.03.001 ПЗ	Лист
						67
Изм.	Лист	№ докум.	Подпись	Дата		

```

<button class="toggle-btn" id="toggleBtn">
  
</button>
  <ul class="menu" id="menu">
    <li>
      
      <span class="text">Главная</span>
    </li>
    <li
onclick="location.href='http://localhost:1880/messenger';"
style="cursor:pointer;">
      
      <span class="text">Мессенджер</span>
    </li>
    <li onclick="location.href='http://localhost:1880/ssau';"
style="cursor:pointer;">
      
      <span class="text">Расписание</span>
    </li>
    <li onclick="location.href='http://localhost:1880/logout';"
style="cursor:pointer;">
      <span class="logout">Выйти</span>
    </li>
  </ul>
</div>

<div class="main_content">
  <div>
    <div class="info-label"></div>
    <div class="info-rectangle">
      <div class="table" id="scheduleTable">
        <div class="row header-row" id="tableHeader">
          <div class="cell time">Время</div>
        </div>
      </div>
    </div>
  </div>
</div>

<script>
  const imagesToPreload = [
    "http://localhost:1880/logo_osnovnoy.png",

```

```

"https://raw.githubusercontent.com/Kulakov1Dima/SsauWorkspace/refs/heads/main/hahaton_exemple_fronted/logo_osnovnoy_small.png"
];

imagesToPreload.forEach(src => {
  const img = new Image();
  img.src = src;
});

const sidebar = document.getElementById('sidebar');
const toggleBtn = document.getElementById('toggleBtn');
const logo = document.getElementById('logo');
const logocenter = document.getElementById('logocenter');
const menu = document.getElementById('menu');

toggleBtn.addEventListener('click', () => {
  if (!toggleBtn.classList.contains('expanded1')) {
    toggleBtn.classList.add('expanded1');
    sidebar.classList.add('expanded1');
  }
  else{
    toggleBtn.classList.remove('expanded1');
    sidebar.classList.remove('expanded1');
  }
});

sidebar.addEventListener('mouseenter', () => {
  logo.src = 'http://localhost:1880/logo_osnovnoy.png';
  logo.classList.add('small');
  logocenter.classList.add('small');
  menu.classList.add('small');
});

sidebar.addEventListener('mouseleave', () => {
  if (!sidebar.classList.contains('expanded1')) {
logo.src =
'https://raw.githubusercontent.com/Kulakov1Dima/SsauWorkspace/refs/heads/main/hahaton_exemple_fronted/logo_osnovnoy_small.png';
    logo.classList.remove('small');
    logocenter.classList.remove('small');
    menu.classList.remove('small');
  }
});

let currentMonday = getMonday(new Date());

function getMonday(date) {
  const d = new Date(date);
  const day = d.getDay();
  const diff = d.getDate() - ((day + 6) % 7); // Преобразует:
  понедельник = 0, воскресенье = 6
  d.setDate(diff);
  d.setHours(0, 0, 0, 0); // Обнулили время

```

```

    return d;
}
function formatDate(date) {
    const year = date.getFullYear();
    const month = String(date.getMonth() + 1).padStart(2, '0');
    const day = String(date.getDate()).padStart(2, '0');
    return `${year}-${month}-${day}`;
}

function formatDisplayDate(date) {
    return date.toLocaleDateString('ru-RU', {
        day: '2-digit',
        month: '2-digit',
        year: '2-digit'
    });
}

function updateYearLabel() {
    const yearLabel = document.querySelector('.info-label');
    const year = currentMonday.getFullYear();
    const nextYear = currentMonday.getMonth() >= 8 ? year + 1 : year;
    yearLabel.innerHTML = `
        <button onclick="changeWeek(-1)">←</button>
        ${nextYear - 1} / ${nextYear} учебный год
        <button onclick="changeWeek(1)">→</button>
    `;
}

function changeWeek(offset) {
    currentMonday.setDate(currentMonday.getDate() + offset * 7);
    renderSchedule();
}

function closeModal() {
    document.getElementById('eventModal').style.display = 'none';
}

function renderSchedule() {
    updateYearLabel();

    const allTimes = [
        "08:00 - 09:35",
        "09:45 - 11:20",
        "11:30 - 13:05",
        "13:30 - 15:05",
        "15:15 - 16:50"
    ];

    fetch('http://localhost:1880/schedule')
        .then(response => response.json())
            .then(data => {
                const tableHeader = document.getElementById('tableHeader');
                const scheduleTable =
document.getElementById('scheduleTable');

```

```

        // Очистка таблицы
        scheduleTable.querySelectorAll('.row:not(.header-
row)').forEach(row => row.remove());
        tableHeader.innerHTML = '<div class="cell
time">Время</div>';

        // Дни недели (6 штук)
        const days = [];
        const dayNames = ["Понедельник", "Вторник", "Среда",
"Четверг", "Пятница", "Суббота"];
        for (let i = 0; i < 6; i++) {
            const date = new Date(currentMonday.getTime());
            date.setDate(date.getDate() + i);
            days.push(formatDate(date));
            const headerCell = document.createElement('div');
            headerCell.className = 'cell';
            headerCell.innerHTML = `
            ${dayNames[i]}<br>${formatDisplayDate(date)}`;
            tableHeader.appendChild(headerCell);
        }

        // Для каждого временного слота – строка
        allTimes.forEach(time => {
            const row = document.createElement('div');
            row.className = 'row';

            const timeCell = document.createElement('div');
            timeCell.className = 'cell time';
            timeCell.textContent = time;
            row.appendChild(timeCell);
            // Для каждого дня недели – одна ячейка
            for (let i = 0; i < 6; i++) {
                const dateObj = new Date(currentMonday.getTime());
                dateObj.setDate(dateObj.getDate() + i);
                const dateStr = formatDate(dateObj);
                const cell = document.createElement('div');
                cell.className = 'cell';

                const event = data.find(ev => ev.date === dateStr &&
ev.time === time);
                if (event) {
                    cell.classList.add('subject');
                    if (event.type === "Лабораторная")
cell.classList.add('purple');
                    else if (event.type === "Лекция")
cell.classList.add('green');
                    else if (event.type === "Практика")
cell.classList.add('blue');
                    else if (event.type === "Экзамен")
cell.classList.add('cyan');

```

```

        cell.innerHTML = `
            <b>${event.name}</b><br>
            <span>${event.teacher}</span><br>
            <span>${event.platform || event.room || ''}</span>
            ${event.conference ? '' : ''}<br><a
href="${event.conference}">Перейти в конференцию</a>` : ''
        `;
        cell.setAttribute('data-subject', event.name);
        cell.addEventListener('click', () => {
            window.location.href =
`subject?name=${encodeURIComponent(event.name)}`;
        });
    } else {
        // Добавим кнопку "+" в пустую ячейку
        cell.style.position = 'relative';
        const addButton = document.createElement('button');
        addButton.textContent = '+';
        addButton.className = 'add-button';
        addButton.title = 'Добавить событие';
        addButton.onclick = () => openAddModal(dateStr,
time);
        cell.appendChild(addButton);
    }

    row.appendChild(cell);
}

scheduleTable.appendChild(row);
});
})
.catch(error => {
    console.error("Ошибка получения данных расписания:", error);
});
}

function openAddModal(date, time) {
    const modal = document.getElementById('eventModal');
    modal.style.display = 'block';
    modal.setAttribute('data-date', date);
    modal.setAttribute('data-time', time);
}

function saveEvent() {
    const gruppa = document.getElementById('eventGruppa').value.trim();

    const name = document.getElementById('eventName').value.trim();
    const teacher =
document.getElementById('eventTeacher').value.trim();

    const room = document.getElementById('eventRoom').value.trim();
    const platform =
document.getElementById('eventPlatform').value.trim();
    const conference =

```

```

document.getElementById('eventConference').value.trim();
const type = document.getElementById('eventType').value;

const modal = document.getElementById('eventModal');
const date = modal.getAttribute('data-date');
const time = modal.getAttribute('data-time');

if (!name || !teacher || !date || !time || !gruppa) {
    alert("Пожалуйста, заполните все обязательные поля.");
    return;
}

const newEvent = {
    id: Math.floor(Date.now() / 1000),
    gruppa: gruppa,
    date: date,
    time: time,
    name: name,
    teacher: teacher,
    type: type,
    room: room,
    platform: platform,
    conference: conference
};

fetch('http://localhost:1880/schedule', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify(newEvent)
})
//.then(response => {
//    if (!response.ok) {
//        throw new Error("Ошибка сохранения события");
//    }
//    return response.json();
//})
.then(() => {
    closeModal();
    renderSchedule(); // перерисуем таблицу
})
.catch(error => {
    alert("Ошибка при сохранении события: " + error.message);
});
}
document.addEventListener('DOMContentLoaded', renderSchedule);

</script>
</body>
</html>

```

Файл «subject.html»

```

<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8" />
  <title id="subjectTitle">Информация о предмете</title>
  <style>
    {{{cssText}}}
  </style>
  <link rel="icon" type="image/png"
href="https://cabinet.ssau.ru/asset/favicon.ico">
</head>
<body>
  <header class="topbar">
    <div class="divlogoname"></div>
    </header>

    <div class="sidebar" id="sidebar">
      <div class="logocenter" id="logocenter">
        
        </div>
        <button class="toggle-btn" id="toggleBtn">
          
        </button>
        <ul class="menu" id="menu">
          <li>
            
            <span class="text">Главная</span>
          </li>
          <li>
            
            <span class="text">Мессенджер</span>
          </li>
          <li onclick="location.href='http://localhost:1880/ssau';"
style="cursor:pointer;">
            
            <span class="text">Расписание</span>
          </li>
        </ul>
      </div>
    </div>
  </body>
</html>

```

```

    </ul>
</div>

<div class="main_content">
  <div class="info-subject"></div>
  <div class="info-rectangle">
    <div class="task_info_table">
      <div class="row">
        <div class="cell task_state">

          <div class="lab-details hidden">
            <div class="info-rectangle">
              <div class="lab-comment">
                <h3>Комментарий преподавателя</h3>
                <p id="labComment"></p>
                <h4>Статус: <span id="labStatus"></span></h4>
                <div class="progress-container">
                  <div class="progress-bar" id="labProgressBar"
style="width: 0%></div>
                </div>
              </div>
            </div>
          </div>

          <div class="teacher-card">
            <h4 class="teacher-title">Ваш преподаватель</h4>
            <div class="teacher-info">
              
              <div class="teacher-details">
                <p id="teacherName"></p>
                <p>Email: <a href="mailto:ivanov@example.com"
id="teacherEmail"></a></p>
                <button class="msg-button">Написать сообщение</button>
              </div>
            </div>
          </div>

          <p id="subjectDescription"></p>

        </div>
        <div class="cell task_state">
          <div class="task_table" id="taskTable">
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

<script>

```

```

const imagesToPreload = [
  "http://localhost:1880/logo_osnovnoy.png",

  "https://raw.githubusercontent.com/Kulakov1Dima/SsauWorkspace/refs/heads/main/hahaton_exemple_fronted/logo_osnovnoy_small.png"
];

imagesToPreload.forEach(src => {
  const img = new Image();
  img.src = src;
});

const sidebar = document.getElementById('sidebar');
const toggleBtn = document.getElementById('toggleBtn');
const logo = document.getElementById('logo');
const logocenter = document.getElementById('logocenter');
const menu = document.getElementById('menu');

toggleBtn.addEventListener('click', () => {
  if (!toggleBtn.classList.contains('expanded1')) {
    toggleBtn.classList.add('expanded1');
    sidebar.classList.add('expanded1');
  }
  else{
    toggleBtn.classList.remove('expanded1');
    sidebar.classList.remove('expanded1');
  }
});

sidebar.addEventListener('mouseenter', () => {
  logo.src = 'http://localhost:1880/logo_osnovnoy.png';
  logo.classList.add('small');
  logocenter.classList.add('small');
  menu.classList.add('small');
});

sidebar.addEventListener('mouseleave', () => {
  if (!sidebar.classList.contains('expanded1')) {
    logo.src =
  "https://raw.githubusercontent.com/Kulakov1Dima/SsauWorkspace/refs/heads/main/hahaton_exemple_fronted/logo_osnovnoy_small.png";
    logo.classList.remove('small');
    logocenter.classList.remove('small');
    menu.classList.remove('small');
  }
});

document.addEventListener('DOMContentLoaded', () => {
  const tasks = document.querySelectorAll('.task');
  const teacherCard = document.querySelector('.teacher-card');
  const labDetails = document.querySelector('.lab-details');
  const labComment = document.getElementById('labComment');

```

```

const labStatus = document.getElementById('labStatus');
const labProgressBar = document.getElementById('labProgressBar');
const taskTable = document.getElementById('taskTable');

taskTable.addEventListener('click', async (event) => {
  const task = event.target.closest('.task');
  if (!task) return;

  const labId = task.dataset.labId;
  if (!labId) return;
  try {
    const response = await
fetch(`http://localhost:1880/lab/${labId}`);
    const labData = await response.json();

    labComment.textContent = labData.comment;
    labStatus.textContent = labData.status;
    labProgressBar.style.width = `${labData.progress}%`;

    labDetails.classList.remove('hidden');
    teacherCard.classList.add('hidden');

    let existingAssignment = document.querySelector('.assignment-
details');
    if (!existingAssignment) {
      existingAssignment = document.createElement('div');
      existingAssignment.className = 'assignment-details
main_content';
      document.body.appendChild(existingAssignment);
    }

    let taskWithConsole = labData.task;
    if (taskWithConsole.includes('[[CONSOLE]]')) {
      const consoleHTML = `<iframe src="https://mail.ru"
style="width:100%; height:480px; border:none; margin-
top:20px;"></iframe>`;
      taskWithConsole = taskWithConsole.replace('[[CONSOLE]]',
consoleHTML);
    }
    if (taskWithConsole.includes('[[PYTHON]]')) {
      const consoleHTML = `<div id="pythonLoader" class="loader-
container"><div class="loader"></div><p>Загрузка Jupyter
Notebook...</p></div><iframe id="pythonIframe"
src="http://127.0.0.1:8888/notebooks/Untitled.ipynb"
style="width:100%; height:1200px; border:none; margin-top:20px;
display:none;"
sandbox="allow-scripts allow-same-origin allow-popups allow-
forms"></iframe>`;
      taskWithConsole = taskWithConsole.replace('[[PYTHON]]',
consoleHTML);
      setTimeout(() => {
        const pythonIframe = document.getElementById('pythonIframe');

const pythonLoader = document.getElementById('pythonLoader');

```

					ПГУ 1.090301.03.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		77

```

        pythonIframe.onload = () => {
            pythonLoader.style.display = 'none';
            pythonIframe.style.display = 'block';
        };
    }, 0);
}

existingAssignment.innerHTML = `
<div class="assignment-details">
    <div class="info-rectangle">
        <div class="task_p">
            <h3>Задание лабораторной работы</h3>
            <p>${taskWithConsole}</p>
            <p>Статус: <span class="assignment-
status">${labData.status}</span></p>
            <div class="progress-container">
                <div class="progress-bar" style="width:
${labData.progress}%;"></div>
            </div>
        </div>
    </div>
</div>
`;
} catch (error) {
    console.error('Ошибка при загрузке данных:', error);
}
});

const urlParams = new URLSearchParams(window.location.search);
let subjectName = urlParams.get('name');

if (!subjectName || subjectName === "null") {
    window.location.href = 'error.html';
} else {
    const infoLabel = document.querySelector('.info-subject');
    infoLabel.textContent = subjectName;
    document.getElementById("subjectTitle").textContent = subjectName;
    document.getElementById("subjectDescription").textContent = `Здесь
могла бы быть подробная информация о предмете «${subjectName}».`;

    const defaultTeacher = {
        name: "Не найден",
        email: "null",
        photo: "https://cdn-icons-png.flaticon.com/512/4837/4837857.png"
    };

    fetch(`http://localhost:1880/teacher${encodeURIComponent(subjectName)}`)
    .then(response => response.json())
    .then(data => {

document.getElementById('teacherPhoto').src = data.photo ||
defaultTeacher.photo;

```

					ПГУ 1.090301.03.001 ПЗ	Лист
						78
Изм.	Лист	№ докум.	Подпись	Дата		

```

        document.getElementById('teacherName').textContent = data.name ||
defaultTeacher.name;
        document.getElementById('teacherEmail').href = `mailto:${data.email
|| defaultTeacher.email}`;
        document.getElementById('teacherEmail').textContent = data.email
|| defaultTeacher.email;
    })
    .catch(error => {
        console.error('Ошибка получения данных преподавателя:', error);
        document.getElementById('teacherPhoto').src =
defaultTeacher.photo;
        document.getElementById('teacherName').textContent =
defaultTeacher.name;
        document.getElementById('teacherEmail').href =
`mailto:${defaultTeacher.email}`;
        document.getElementById('teacherEmail').textContent =
defaultTeacher.email;
    });

fetch(`http://localhost:1880/labs/${encodeURIComponent(subjectName)}`)
    .then(response => response.json())
    .then(data => {
        data.forEach(lab => {
            const labDiv = document.createElement('div');
            labDiv.className = 'task subject';
            labDiv.dataset.labId = lab.id;

            if (lab.status === "Сдано") {
                labDiv.classList.add('green');
            } else if (lab.status === "Не сдано") {
                labDiv.classList.add('red');
            } else if (lab.status === "В процессе") {
                labDiv.classList.add('blue');
            }

            labDiv.innerHTML = `
                <b>${lab.title}</b><br>
                <span>Статус: ${lab.status}</span>
            `;
            taskTable.appendChild(labDiv);
        });
    })
    .catch(error => {
        console.error('Ошибка при загрузке данных:', error);
    });
}
});
</script>
</body>
</html>

```

Файл «messenger.html»

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Активные чаты</title>
  <link rel="icon" type="image/png"
href="https://cabinet.ssau.ru/asset/favicon.ico">
  <style>
    {{{cssText}}}
  </style>
  <style>
    /* Дополнительные стили для этой страницы */
    .main_content {
      margin-left: 100px; /* Уменьшаем отступ, чтобы боковое меню было
лучше видно */
      display: grid;
      grid-template-columns: 1fr 2fr; /* Две колонки: для списка и для
чата */
      gap: 20px; /* Отступ между колонками */
    }

    .dialog-list-container {
      /* Контейнер для списка диалогов */
    }

    .dialog-list {
      list-style: none;
      padding: 0;
      margin: 20px 0;
    }

    .dialog-item {
      display: flex;
      align-items: center;
      padding: 10px;
      border-bottom: 1px solid #eee;
      cursor: pointer;
      background-color: #fff;
    }

    .dialog-item:hover {
      background-color: #f9f9f9;
    }

    .dialog-icon {
      width: 30px;
      height: 30px;
      margin-right: 10px;
      border-radius: 50%;
      display: flex;
      align-items: center;
```

					ПГУ 1.090301.03.001 ПЗ	Лист
						80
Изм.	Лист	№ докум.	Подпись	Дата		

```

justify-content: center;
    color: white;
}

.dialog-icon.tz {
    background-color: #0D47A1; /* Цвет для иконки "ТЗ" */
}

.dialog-icon.chat {
    background-color: #1D5DEB; /* Цвет для иконки чата */
}

.dialog-text {
    flex-grow: 1;
    overflow: hidden;
    text-overflow: ellipsis;
    white-space: nowrap;
}

.chat-view {
    display: none;
    margin-top: 20px;
    background-color: #fff;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    border-radius: 5 px;
}

.chat-view.active {
    display: block;
}

#searchBar {
    width: 100%;
    padding: 10px;
    margin-bottom: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
    box-sizing: border-box;
}

.hidden {
    display: none;
}

.chat-messages {
    height: 300px;
    overflow-y: scroll;
    padding: 10px;
    border: 1px solid #ddd;
    margin-bottom: 10px;
}

```

```

.message {
  padding: 8px 12px;
  margin-bottom: 5px;
  border-radius: 5px;
  background-color: #f0f0f0;
}

.input-area {
  display: flex;
  align-items: center;
}

.input-area input {
  flex-grow: 1;
  padding: 8px;
  border: 1px solid #ddd;
  border-radius: 5px;
}

.input-area button {
  padding: 8px 12px;
  margin-left: 5px;
  border: none;
  border-radius: 5px;
  background-color: #007bff;
  color: white;
  cursor: pointer;
}

.input-area button:hover {
  background-color: #0056b3;
}
</style>
</head>
<body>

<header class="topbar">
  <div class="divlogoname"></div>
</header>

<div class="sidebar" id="sidebar">
  <div class="logocenter" id="logocenter">
    
  </div>
<button class="toggle-btn" id="toggleBtn">

```

```

        
    </button>
    <ul class="menu" id="menu">
        <li>
            
            <span class="text">Главная</span>
        </li>
        <li onclick="location.href='http://localhost:1880/messenger';"
style="cursor:pointer;">
            
            <span class="text">Мессенджер</span>
        </li>
        <li onclick="location.href='http://localhost:1880/ssau';"
style="cursor:pointer;">
            
            <span class="text">Расписание</span>
        </li>
    </ul>
</div>

<div class="main_content">

<div class="dialog-list-container">
<h2>Активные чаты</h2>
<input type="text" id="searchBar" placeholder="Поиск...">
<ul class="dialog-list" id="dialogList">
</ul>
</div>

<div id="chatView" class="chat-view">
<h3>Название чата</h3>
<div class="chat-messages">
    <!-- Сообщения чата будут здесь -->
    <div class="message"></div>
    <div class="message"></div>
</div>
<div class="input-area">
    <input type="text" placeholder="Сообщение...">
    <button>Отправить</button>
</div>
</div>

```

```

</div>

<script>
const imagesToPreload = [
  "http://localhost:1880/logo_osnovnoy.png",
  "https://raw.githubusercontent.com/Kulakov1Dima/SsauWorkspace/refs/heads/main/hahaton_exemple_fronted/logo_osnovnoy_small.png"
];

imagesToPreload.forEach(src => {
  const img = new Image();
  img.src = src;
});

const sidebar = document.getElementById('sidebar');
const toggleBtn = document.getElementById('toggleBtn');
const logo = document.getElementById('logo');
  const logocenter = document.getElementById('logocenter');
  const menu = document.getElementById('menu');

toggleBtn.addEventListener('click', () => {
  if (!toggleBtn.classList.contains('expanded1')) {
    toggleBtn.classList.add('expanded1');
    sidebar.classList.add('expanded1');
  } else {
    toggleBtn.classList.remove('expanded1');
    sidebar.classList.remove('expanded1');
  }
});

sidebar.addEventListener('mouseenter', () => {
  logo.src = 'http://localhost:1880/logo_osnovnoy.png';
  logo.classList.add('small');
  logocenter.classList.add('small');
  menu.classList.add('small');
});

sidebar.addEventListener('mouseleave', () => {
  if (!sidebar.classList.contains('expanded1')) {
    logo.src =
'https://raw.githubusercontent.com/Kulakov1Dima/SsauWorkspace/refs/heads/main/hahaton_exemple_fronted/logo_osnovnoy_small.png';
    logo.classList.remove('small');
    logocenter.classList.remove('small');
    menu.classList.remove('small');
  }
});

document.addEventListener('DOMContentLoaded', function () {
  const dialogList = document.getElementById('dialogList');
  const chatView = document.getElementById('chatView');
  const searchBar = document.getElementById('searchBar');

```

```

const dialogs = [
  {
    type: 'tz',
    name: 'Использование Linux',
    url:
'https://localhost:1880/subject?name=Использование%20Linux'
  },
  {
    type: 'chat',
    name: 'Использование Linux | Чат'
  },
  {
    type: 'tz',
    name: 'Системное программирование',
    url:
'https://localhost:1880/subject?name=Системное%20программирование'
  },
  {
    type: 'chat',
    name: 'Системное программирование | Чат'
  },
  {
    type: 'tz',
    name: 'Системы искусственного интеллекта',
    url:
'https://localhost:1880/subject?name=Системы%20искусственного%20интеллект'
  },
  {
    type: 'chat',
    name: 'Системы искусственного интеллекта | Чат'
  },
  {
    type: 'tz',
    name: 'Технологии проектирования',
    url:
'https://localhost:1880/subject?name=Технологии%20проектирования'
  }
];

function displayDialogs(dialogsToDisplay) {
  dialogList.innerHTML = '';

  dialogsToDisplay.forEach(dialog => {
    const li = document.createElement('li');
    li.classList.add('dialog-item');
    li.dataset.type = dialog.type;
    li.dataset.name = dialog.name; // Сохраняем название для
показа в чате

    let iconText = '';
    let iconClass = '';

```

```

        if (dialog.type === 'tz') {
iconText = 'T3';
            iconClass = 'tz';
            li.addEventListener('click', () => {
                window.location.href = dialog.url;
            });
        } else if (dialog.type === 'chat') {
            iconText = '<i class="fas fa-users"></i>'; // Используем
Font Awesome иконку
            iconClass = 'chat';
            li.addEventListener('click', () => {
                displayChat(dialog.name);
            });
        }

        li.innerHTML = `
            <div class="dialog-icon ${iconClass}">${iconText}</div>
            <div class="dialog-text">${dialog.name}</div>
        `;

        dialogList.appendChild(li);
    });
}

function displayChat(chatName) {
    const chatView = document.getElementById('chatView');
    chatView.querySelector('h3').textContent = chatName;
    chatView.classList.add('active');
}

searchBar.addEventListener('input', function (e) {
    const searchTerm = e.target.value.toLowerCase();
    const filteredDialogs = dialogs.filter(dialog => {
        return dialog.name.toLowerCase().includes(searchTerm);
    });
    displayDialogs(filteredDialogs);
});

// Инициализация отображения диалогов
displayDialogs(dialogs);
});

</script>
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.0.0/css/all.min.css" integrity="sha512-
9usAa10IRO0HhonyAIIVpjrylPvoDwiPUiKdWk5t3PyolY1cOd4DSE0Ga+ri4AuTroPR5a
QvXU9xC6qOPnzFeg==" crossorigin="anonymous" referrerpolicy="no-
referrer" />
</body>
</html>

```

Приложение В

(Обязательное)

Программа выполнения практических заданий

Презентация

					ПГУ 1.090301.03.001 ПЗ	Лист
						87
Изм.	Лист	№ докум.	Подпись	Дата		

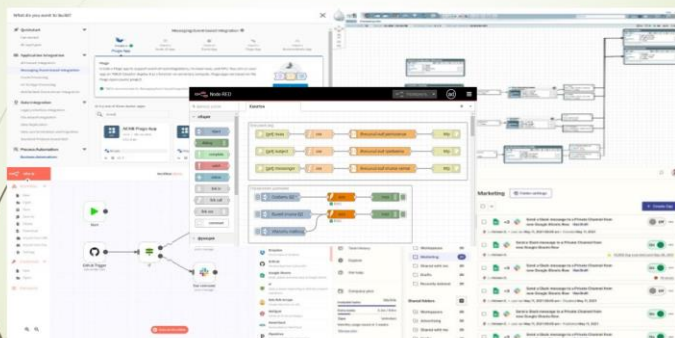
Докладчик: студент группы 21ВВВ1 Антропов Д.В.

Аналогичная система	Достоинства	Недостатки
Moode	Бесплатная и открытая код. Гибкость и настраиваемость. Поддержка различных форматов обучения.	Устаревший интерфейс. Сложная настройка и администрирование. Отсутствие возможности запуска кода.
Google Classroom	Бесплатность для образовательных учреждений. Интеграция с другими сервисами Google. Редактирование документов и комментирование работ. Интерактивный формат обучения.	Жесткая привязка к сервисам Google. Отсутствие инструментов для создания интерактивных курсов и встроенного тестирования. Отсутствие возможности запуска кода.
Stepik	Гибкий графический обучения. Поддержка геймификации.	Ограничение возможности для создания интерактивных курсов. Минимальная кастомизация.
Uralis	Простой и понятный интерфейс. Визуальное управление заданиями.	Не подходит для сложных проектов. Малая анфлексия. Большая часть функций в

Задачи, решаемые в рамках работы

- разработка веб-интерфейса, удобного и интуитивно понятного для пользователей с различным уровнем подготовки к работе с системой;
- механизм управления расписанием занятий, проектными этапами и сроками выполнения заданий;
- возможность добавления лабораторными проектными заданиями;
- реализация выполнения заданий в интегрированной среде;
- система ролевого доступа к системе и различным функционалам в зависимости от ролей;
- модуль оценки результатов ручной, автоматической и комбинированный;
- возможность развертывания системы как в интернете, так и в изолированной локальной сети предприятия;
- реализация прототипа системы, тестирование его на различных сценариях, представляющих собой образовательные корпоративные процессы;
- оценка перспектив масштабирования системы и возможности интеграции с внешними сервисами.

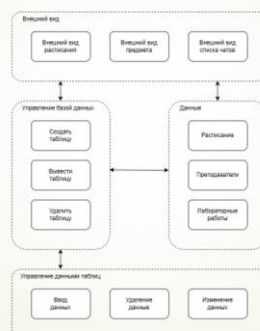
Выбор инструментов разработки



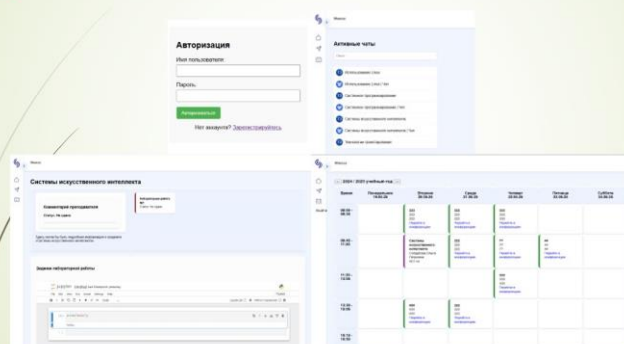
Результаты сравнения

Аналогичная система	Достоинства	Недостатки
Node-Red	Интуитивно понятный интерфейс. Множество примеров и готовых решений. Простота развертывания.	Сложность отладки. Отсутствие типизации и строгой структуры. Сложность в локализации ошибок.
Apache Nifi	Поддержка сбора данных из различных источников. Встроенный мониторинг потоков данных. Поддержка сложных маршрутизаций.	Высокий порог входа. Ограниченная документация. Требовательность к ресурсам. Необходимость ручного управления.
NSN	Сложные цепочки задач с условиями, циклами и ветвлениями. Удобный интерфейс и подробное логирование.	Нужны навыки администрирования серверов. Меньше готовых шаблонов, чем у конкурентов.
Zipper	Простота использования. Огромное количество интеграций. Облачный сервис.	Ограничения бесплатного тарифа. Стоимость различных тарифов. Сложность отладки.
Fligo (IISCO)	Легковесность и высокая производительность. Минимальные задержки.	Меньше документации и сообщество, чем у конкурентов. Меньше готовых конструкторов. Сложность отладки.

Архитектура программы



Итоговый внешний вид программы



Заключение

- В процессе выполнения выпускной квалификационной работы была реализована цель по созданию информационной системы выполнения практических и проектных заданий, ориентированной на использование в образовательной и корпоративной среде. В ходе работы был произведён анализ требований к подобным системам, определена архитектура программного решения, а также разработан прототип веб-приложения, включающий в себя ключевые функциональные модули.

Спасибо за внимание!

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.03.001 ПЗ

Лист

90