

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**
**Федеральное государственное автономное образовательное учреждение
высшего образования**
«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий и интеллектуальных систем
Институт передовых образовательных технологий

ДПП ПП UI/UX и веб-дизайн для разработчиков

ОТЧЕТ
по практике

Слушатель Иванущенко В.А (ФИО)	ЕИ КФУ, е2216 (институт, группа)
Слушатель Никитина Н.О. (ФИО)	ЕИ КФУ, е2216 (институт, группа)
Слушатель Овчинникова К.А. (ФИО)	ЕИ КФУ, е2216 (институт, группа)
Слушатель Курицын К.Р. (ФИО)	ЕИ КФУ, е2216 (институт, группа)

Руководитель практики от компании

Лукьяничева Е.О.
(ФИО, должность)

Оценка за практику

Дата сдачи отчета

19.05.2025

Казань 2025 год

Оглавление

Введение	3
Дневник прохождение практики.....	4
Результат работы.....	5
Список использованных источников.....	10

Введение

Практика проходила на базе компании ООО «Б-Рейн Солюшенс». Компания специализируется на разработке продуктов и решений для бизнеса. Компания также предлагает услуги по оптимизации процессов при помощи машинного обучения и нейронных сетей.

Целью прохождения практики в ИТ-компании было знакомство с основными процессами, методологиями, подходами и технологиями разработки ИТ-продуктов в сфере управления данными, а также получение практического опыта разработки собственного решения.

В рамках практики были определены следующие задачи:

1. Выбор направления практической деятельности.
2. Изучение теоретического материала, соответствующего теме работы (в т.ч. учебников и предполагаемого учебными программами перечня лабораторных работ по физике).
3. Создание библиотеки для виртуальной имитации физических процессов.
4. Разработка программной базы для веб-лаборатории по физике для обучающихся.
5. Формирование интуитивно понятного интерфейса и создание соответствующего дизайна для проекта.

Дневник прохождения практики

Дата	Содержание выполненных работ
28.04. 2025 – 04.05. 2025	Знакомство с компанией и начало работы с руководителем практики. Обсуждение целей и задач проекта. Определение задания на практику. Знакомство с методами и технологиями, необходимыми для реализации проекта (HTML/CSS, JavaScript, Flexbox, Figma, Python, Flask, WSGI, gunicorn, NGinx, servicectl, Let's Encrypt, GitHub)
05.05. 2025 – 16.05. 2025	Разработка проекта: 1. Написание библиотеки визуализации физических процессов 2. Создание адаптивного дизайна платформы. 3. Доработка функционала, добавление дополнительных модулей проекта. 4. Тестирование проекта, внесение необходимых корректив.
17.05. 2025 – 19.05. 2025	Подготовка и оформление отчета по практике.

Результат работы

Практика для команды началась с проработки первичной функциональности физической библиотеки.

Был создан базовый класс физического объекта (`phys`) с функциональностью `drag&drop`, а также возможностью вращения объекта, что показано на рисунке 1. В рамках того же JavaScript класса был реализован дополнительный CSS класс - `phys-attachable`. Характерной чертой этого подкласса стала возможность создавать точки крепления для различных типов объектов, методы поиска пересечений, автоматического и мануального скрепления объектов в группы, а также метод открепления объектов. Были установлены методы, которые позволили отслеживать различные события, производящиеся пользователем, и реагировать на них при изменении состояния групп. Было также реализовано перемещение всей группы скрепленных объектов по рабочей области.

```

class PhysObject {
  constructor(element) {
    this.element = element;
    this.rotation = 0;
    this.isAttached = false;
    this.attachedObjects = new Set();
    this.attachmentPoints = [];
    this.attachedToPoint = null;
    this.parentObject = null;
    this.canRotate = true;

    this.initElement();
  }

  initElement() {
    const element = this.element;
    element.style.position = 'absolute';

    const style = window.getComputedStyle(element);
    const transform = style.transform || style.webkitTransform || '';

    let initialRotation = 0;
    const rotateMatch = transform.match(/rotate\((-?\d{1,2})deg\)/);
    if (rotateMatch) {
      initialRotation = parseFloat(rotateMatch[1]);
    } else if (transform.includes('matrix')) {
      const matrix = transform.match(/matrix(?:3d)?\(((^)+)\)/);
      if (matrix) {
        const values = matrix[1].split(',');
        if (values.length === 6) {
          const a = parseFloat(values[0]);
          const b = parseFloat(values[1]);
          initialRotation = Math.round(Math.atan2(b, a) * (180 / Math.PI));
        }
      }
    }

    this.rotation = initialRotation;
    element.style.transform = `rotate(${initialRotation}deg)`;

    element._physObject = this;
  }

  rotate(angleDelta) {
    if (this.isAttached) return;

    this.rotation += angleDelta;
  }
}

```

Рисунок 1 – Базовый класс физического объекта.

Поскольку любой физический эксперимент представляет собой четко определенную последовательность действий и событий, дополнительно были созданы классы для определения текущего этапа лабораторной работы и класс-итератор, хранящий в себе последовательность шагов. Каждый этап эксперимента хранит в себе информацию о разрешенных подключениях и разъединениях объектов, описание шага, статус выполнения и свой уникальный идентификатор. Таким образом, перед прикреплением или откреплением объекта обязательно вызывается один из двух методов проверки, определяющих допустимость подобного действия на данном этапе. Итератор, в свою очередь, имеет набор геттеров и сеттеров для шагов, несколько видов функций перемещения по шагам, а также методы-триггеры событий. Оба класса выведены на рисунке 2.

```
on(event, callback) {  
  this.eventListeners[event] && this.eventListeners[event].push(callback);  
  return this;  
}  
  
triggerEvent(event, data) {  
  this.eventListeners[event] && this.eventListeners[event].forEach(callback => callback(data));  
}
```

Рисунок 2 – Дополнительные классы по работе с этапами лабораторных работ.

Задав такую базовую функциональность, команда приступила к реализации нескольких первых пробных экспериментов, по мере проектирования которых была замечена недостаточность функционала для полного удобства пользования. Довольно быстро обнаружилось, что в эксперименте могут быть задействованы физические объекты, имеющие точки крепления, но не задуманные для перемещения (например, штатив). Для решения этой задачи функционал класса физических объектов был расширен, а в CSS введён новый селектор - `phys-fixed`, что сделало библиотеку более гибкой.

В дальнейшем, помимо массы незначительных доработок и введения вспомогательных методов, был также разработан функционал для дополнительного селектора `phys-connectors`, который дал возможность добавлять и использовать точки крепления проводов на физический объект, как показано на рисунке 3. Провода динамически отрисовываются при помощи `svg`, цвет проводов задается на этапе инициализации для каждого отдельного объекта. При взаимодействии пользователя с прибором автоматически находится ближайшая к месту клика незанятая точка крепления провода, в остальном же `phys-connectors` имеет поведение, во многом схожее с поведением `phys-attachable`.

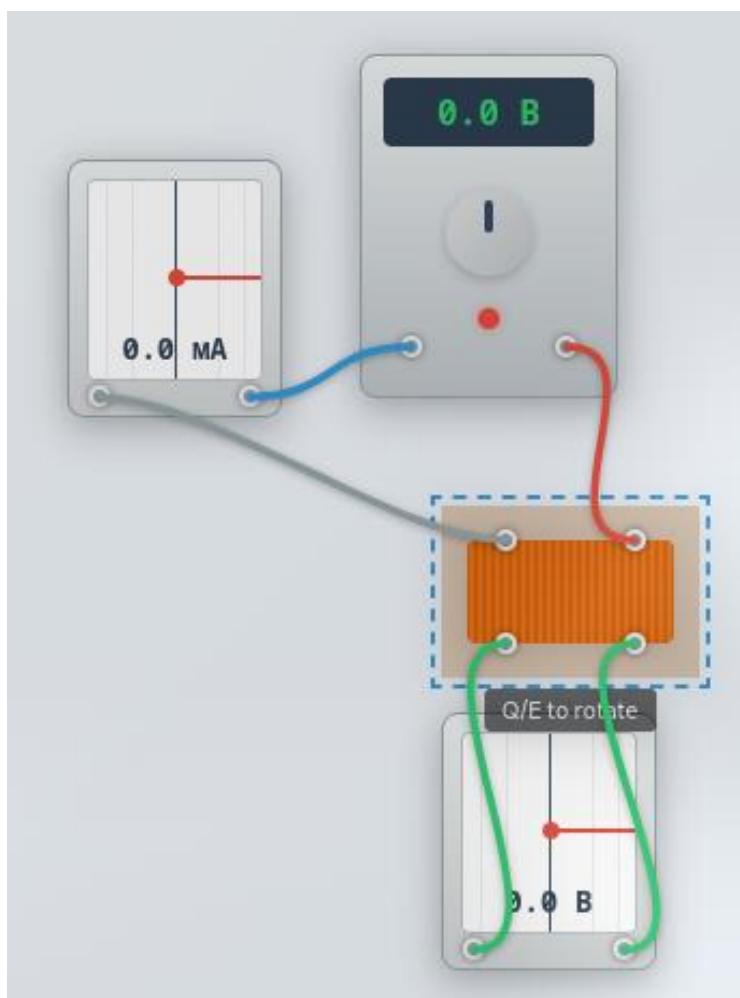


Рисунок 3 – Поведение селектора `phys-connectors`.

Параллельно с улучшением библиотеки велась разработка наглядных экспериментов из школьной программы по физике. Было принято решение для каждого отдельного блока лабораторной работы развести логику обработчиков и функции в различные файлы, что в купе с единообразной структурой работ должно было упростить понимание кода и ускорить процесс разработки. Помимо прочего, для поддержания единой кодовой базы общие стили интерфейса и описание стилей часто используемых физических объектов были вынесены в общие CSS файлы. Итоговый вид индивидуальной структуры каждой лабораторной работы представлен на рисунке 4.

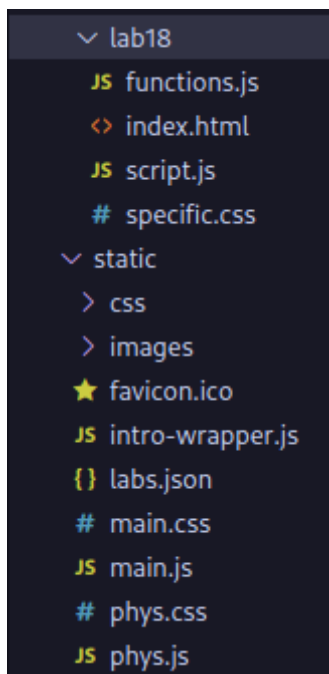


Рисунок 4 – Структура лабораторной работы (на примере lab18) и общих файлов (папка static).

Первоначальное проектирование лабораторных работ велось в формате отдельных микро-проектов, а появившаяся позднее главная страница была статична, подобно большинству элементов интерфейса внутри лабораторных. Однако, когда разработка физических моделей встала на поток и стала видна общая структура страниц, команда принялась за проектирование дизайн-макета итогового продукта и его перенос на готовую платформу. Помимо

этого, произвелась проверка адаптивности и ввод дополнительного функционала, не являющегося основной частью платформы, но заметно улучшающий UX.

В основе нашего решения лежало понимание того, что виртуальная лаборатория по физике должна максимально концентрировать внимание пользователя на сути эксперимента и не отвлекать его лишними деталями. С первых секунд, попадая на главную страницу, сдержанная голубая рамка и едва заметный градиент фона задают тон, так как все, что важно, расположено внутри: карточки с названиями опытов, кратким описанием и метками тем, без избыточных декоративных элементов. Такой минимализм выглядит современно, обеспечивает возможность быстро оценить доступные эксперименты, понятную визуальную иерархию и способствует комфортному чтению текста без утомления глаз. Итоговый вариант главной страницы вы можете рассмотреть на рисунке 5.

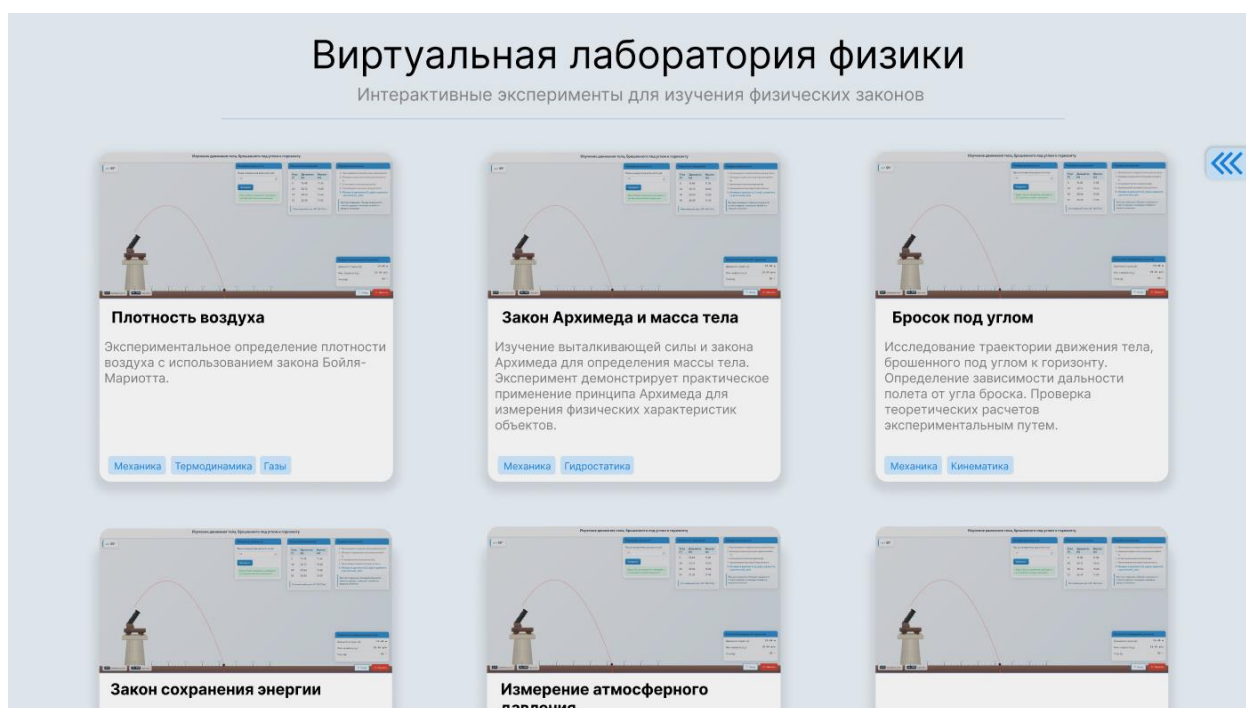


Рисунок 5 – Главная страница.

Боковая панель фильтров с тэгами, доступными в том числе в формате GET-параметров, демонстрирует легкость использования: достаточно одного

клика на стрелку, и список категорий раскрывается, сосредоточивая внимание пользователя на выборе. Отсутствуют сложные выпадающие списки и цветные ярлыки — только сдержанные текстовые кнопки и лаконичные иконки с четкой передачей назначения каждого элемента. Плавная анимация появления панели подчеркивает качество проработки взаимодействия и исключает резкие «скачки» интерфейса, что важно для непрерывности процесса обучения. Визуализация боковой панели представлена на рисунках 6-7.

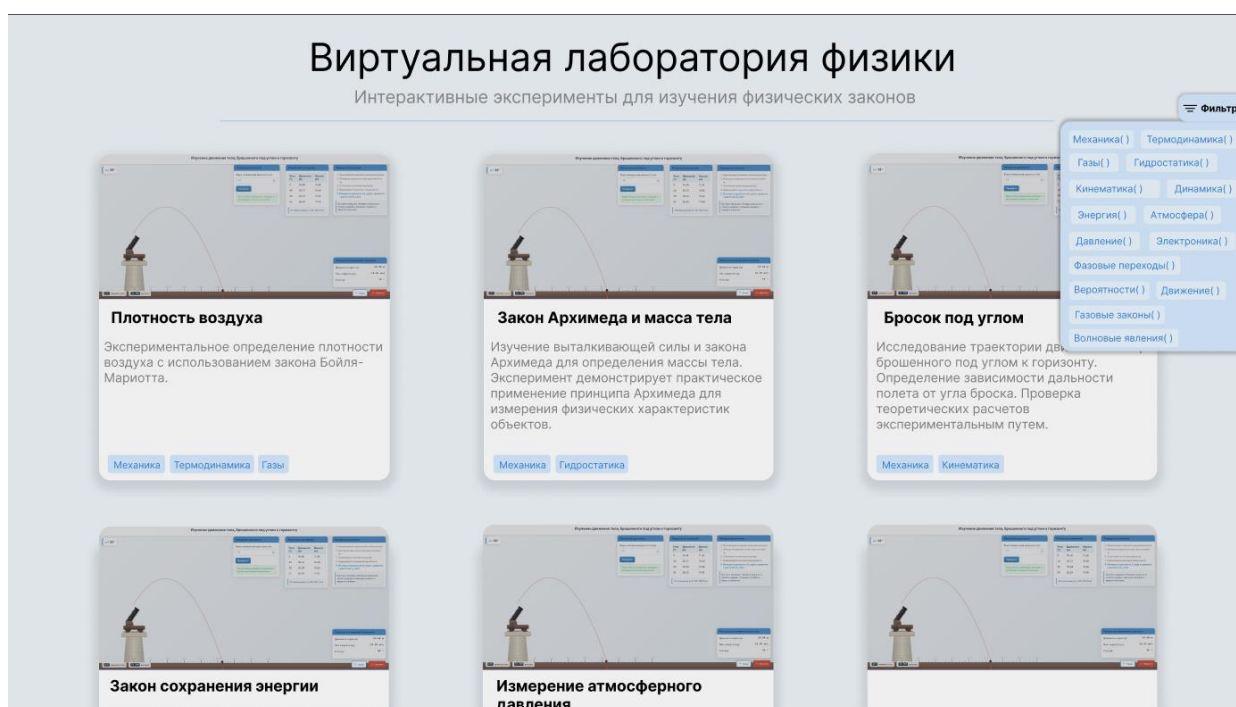


Рисунок 6 – Главная страница с боковой панелью.

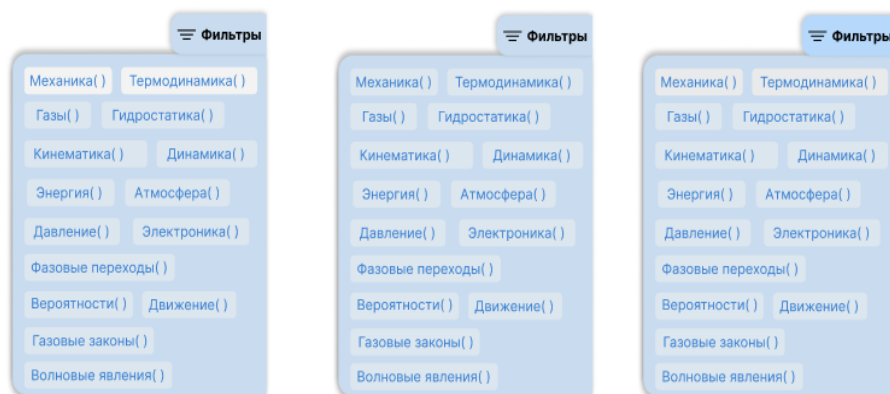


Рисунок 7 – Кнопка фильтрации в разных активностях (активная, неактивная, при наведении).

Общий вид самих лабораторных работ представлен на рисунке 8. Внутри рабочей области эксперимента сохраняется изначально заданный формат: фон ненавязчиво светлый, центральная часть зарезервирована под виртуальное оборудование, а по краям — сдержанные всплывающие подсказки и панель «Порядок выполнения» с несколькими иконками, предлагающими переключаться между инструкцией, таблицей замеров, графиками и результатами измерений. Сами иконки и их поведение показаны на рисунках 9-10. При каждом новом сценарии, будь то оценка плотности воздуха или построение траектории броска под углом, пользователь обнаружит знакомый макет, и единственное, что подвергнется изменениям, — это содержание (текстовые подсказки в облачках и динамически обновляющиеся данные на панелях). Такой подход позволяет с первых запусков «прочувствовать» логику системы без нужды в адаптации к сложной навигации.

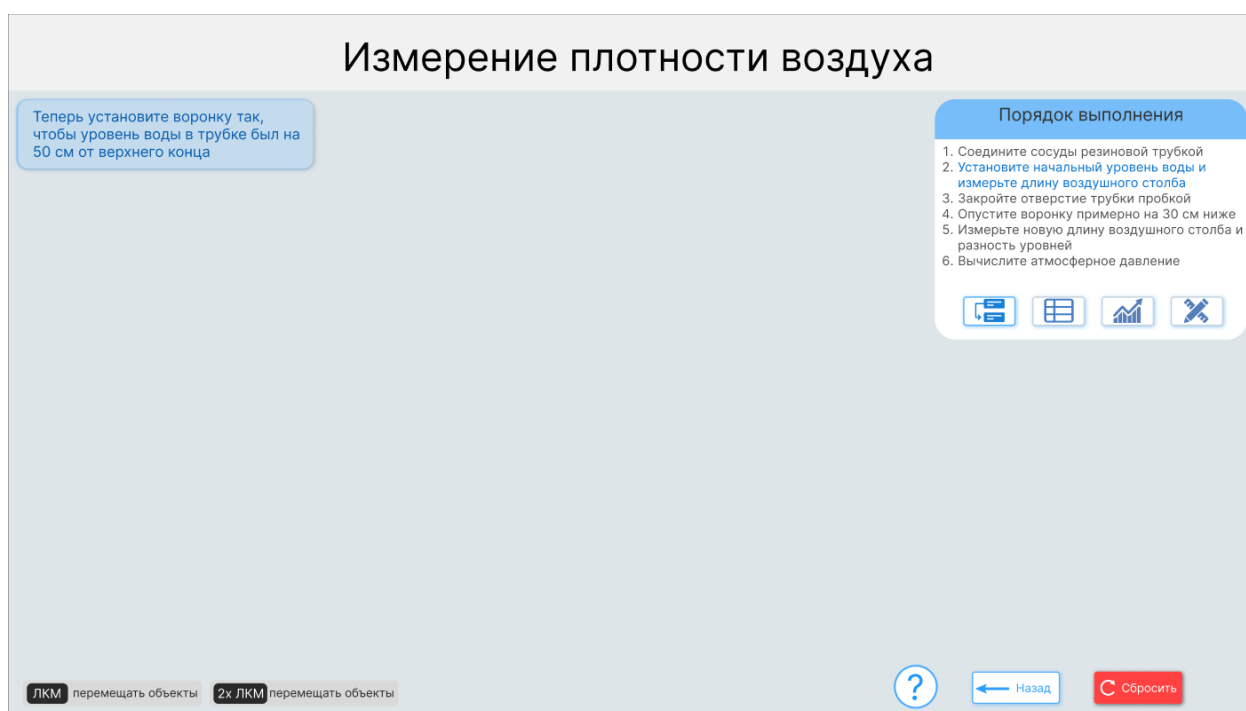


Рисунок 8 – Рабочая область эксперимента.



Рисунок 9 – Иконки рабочей области.



Рисунок 10 – Иконки в разных активностях (активный, неактивный, при наведении).

Особую заботу об удобстве демонстрирует встроенная система подсказок на рисунке 11: нажатие на знак вопроса внизу экрана активирует модальное окно с пошаговыми рекомендациями и иллюстрациями, поясняющими, где искать основные элементы управления и за что отвечают нижние кнопки, выведенные на рисунке 12. При этом сама подсказка сделана в том же минималистичном стиле — светлый прямоугольник на полупрозрачном затемнённом фоне, крупный понятный текст и единичная яркая кнопка «Далее», не отвлекающая от сути и способствующая плавному прохождению инструктажа.

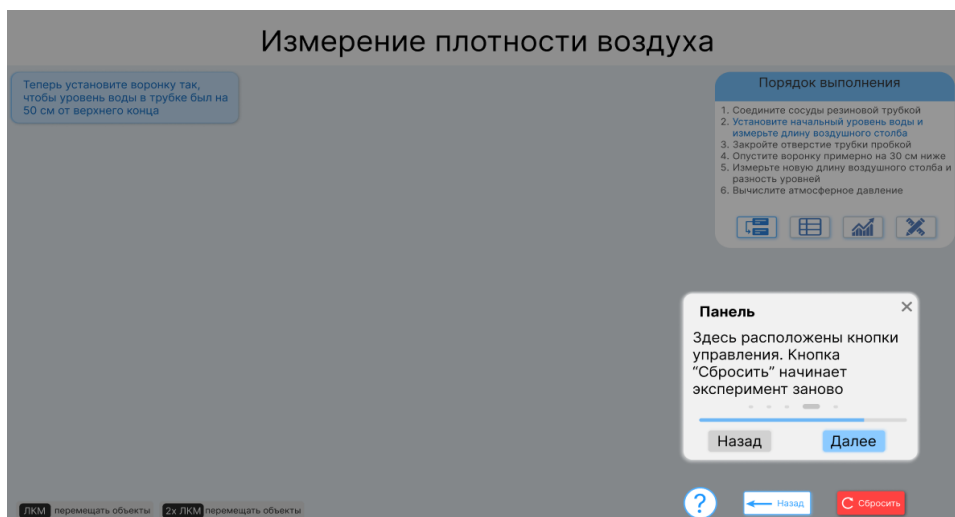


Рисунок 11 – Система подсказок.

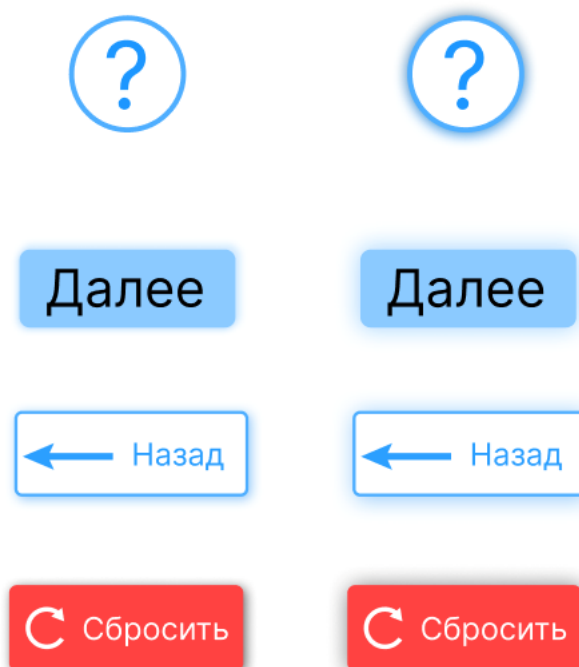


Рисунок 12 – Вспомогательные кнопки в разных активностях (активный, при наведении).

Внимание к деталям уделяется и в мобильной версии: если экран слишком мал, появляется предупреждающее сообщение, дополняемое милым персонажем-жабкой и советом перейти на другое устройство, как и показано на рисунке 13. Вместо неработающего интерфейса пользователь получает понятный и даже чуть шутливый текст, что сохраняет позитивное

впечатление от проекта, не бросая человека «в пустоту» при попытке открыть лабораторию на телефоне.

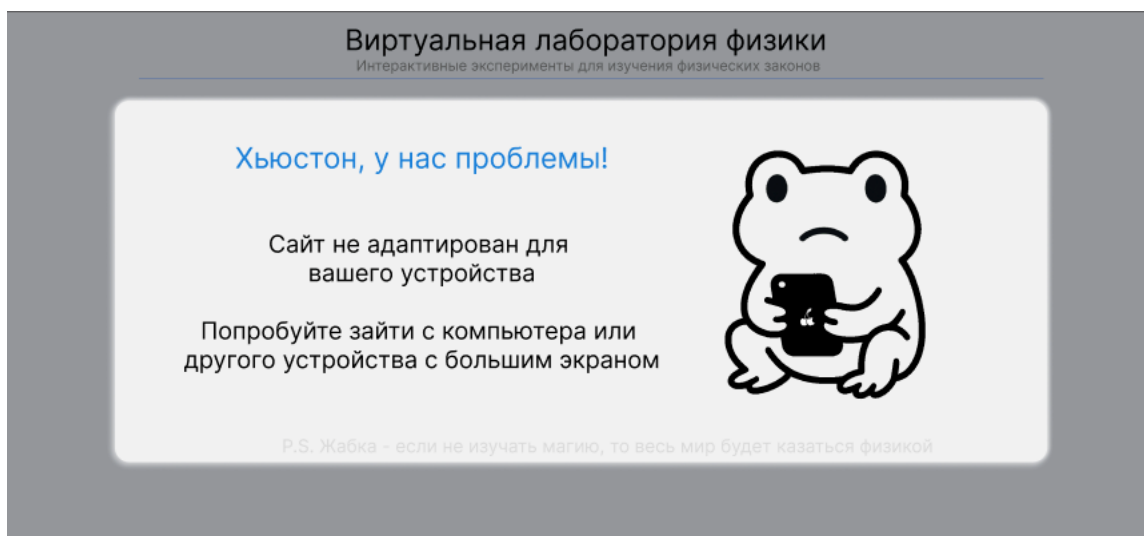


Рисунок 13 – Главная страница в мобильной версии при малых размерах экрана.

Единая палитра оттенков рисунка 14, выбор легко читаемого шрифта без засечек и ограниченный набор цветовых акцентов в кнопках и выделениях создают благоприятную атмосферу, снимающую напряжение и позволяющую сосредоточиться на задачах. Благодаря такой стилистике интерфейс остаётся универсальным для разных возрастных групп, от школьников до преподавателей, и не отвлекает от главного: понимания физических закономерностей на практике.



Рисунок 14 – Цветовая палитра проекта.

В общей сложности для демонстрации принципов работы разрабатываемой физической библиотеки командой были реализованы следующий перечень физических опытов:

1. Измерение плотности воздуха;
2. Изучение закона Архимеда;
3. Изучение баллистической траектории полета тела, брошенного под углом;
4. Изучение закона сохранения энергии;
5. Изучение атмосферного давления при помощи сообщающихся сосудов;
6. Изучение удельной теплоты плавления льда;
7. Изучение удельной температуры парообразования;
8. Изучение электрического сопротивления различных металлов;
9. Изучение поведения световой волны и определение ее длины;
10. Определение скорости света в различных веществах;
11. Измерение ЭДС и внутреннего сопротивления источника питания;
12. Изучение закона радиоактивного распада.

Все реализованные модели позволяют менять начальные параметры эксперимента динамически или в конфиге. Пример готовой лабораторной работы представлен на рисунке 15.

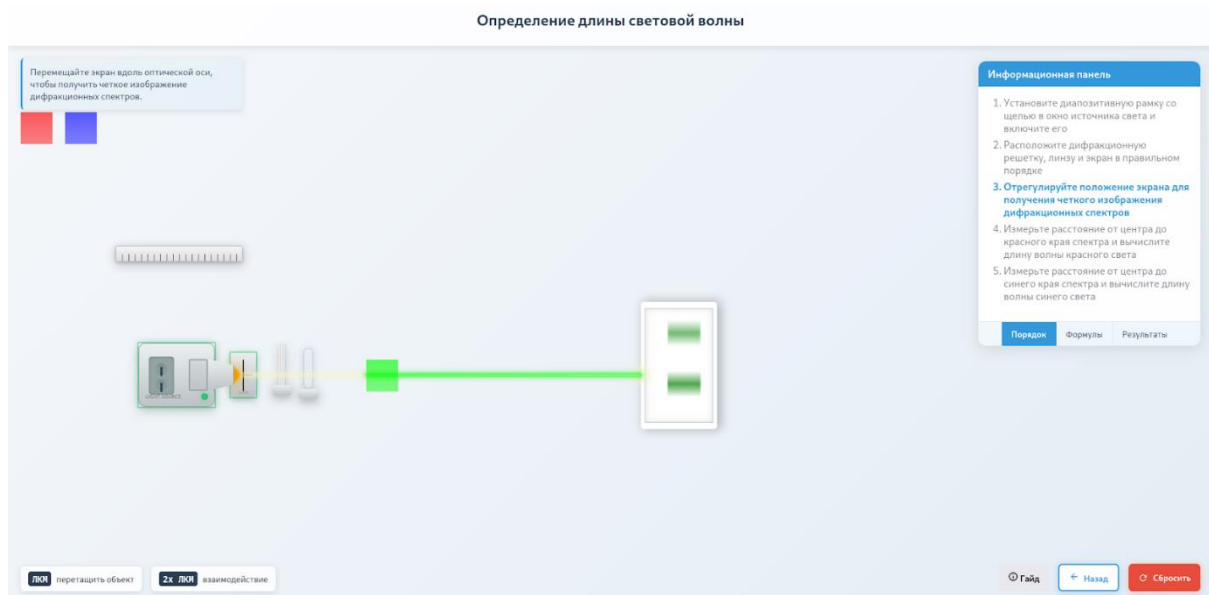


Рисунок 15 – Лабораторная работа «Определение длины световой волны».

С увеличением количества реализуемых экспериментов становилось проблематично каждый раз вручную добавлять их в макет главной страницы, поэтому нами было принято решение написать сервер, который бы помог быстрее форматировать блоки верстки на основе небольшого JSON файла со всеми данными. Для решения поставленной задачи нами был выбран язык программирования Python и фреймворк Flask. Такое решение позволило реализовать динамический поиск всех доступных для фильтрации тэгов и подсчитать количество вхождений этого тэга в списки загруженных экспериментов. На рисунке 16 показан код данной части проекта.

```

@app.route('/api/tags')
def get_tags():
    try:
        json_path = os.path.join(app.static_folder, 'labs.json')

        if not os.path.exists(json_path):
            return jsonify({"error": "Labs data not found"}), 404

        with open(json_path, 'r', encoding='utf-8') as file:
            labs = json.load(file)

        all_tags = []
        for lab in labs:
            if "tags" in lab and isinstance(lab["tags"], list):
                all_tags.extend(lab["tags"])

        unique_tags = []
        for tag in all_tags:
            if tag not in unique_tags:
                unique_tags.append(tag)

        tags_dict = {}
        for tag in all_tags:
            if tag in tags_dict:
                tags_dict[tag] += 1
            else:
                tags_dict[tag] = 1

        sorted_tags = dict(sorted(tags_dict.items(), key=lambda x: x[1], reverse=True))

        return jsonify({
            "tags": unique_tags,
            "tagsWithCount": sorted_tags,
            "total": len(unique_tags)
        })

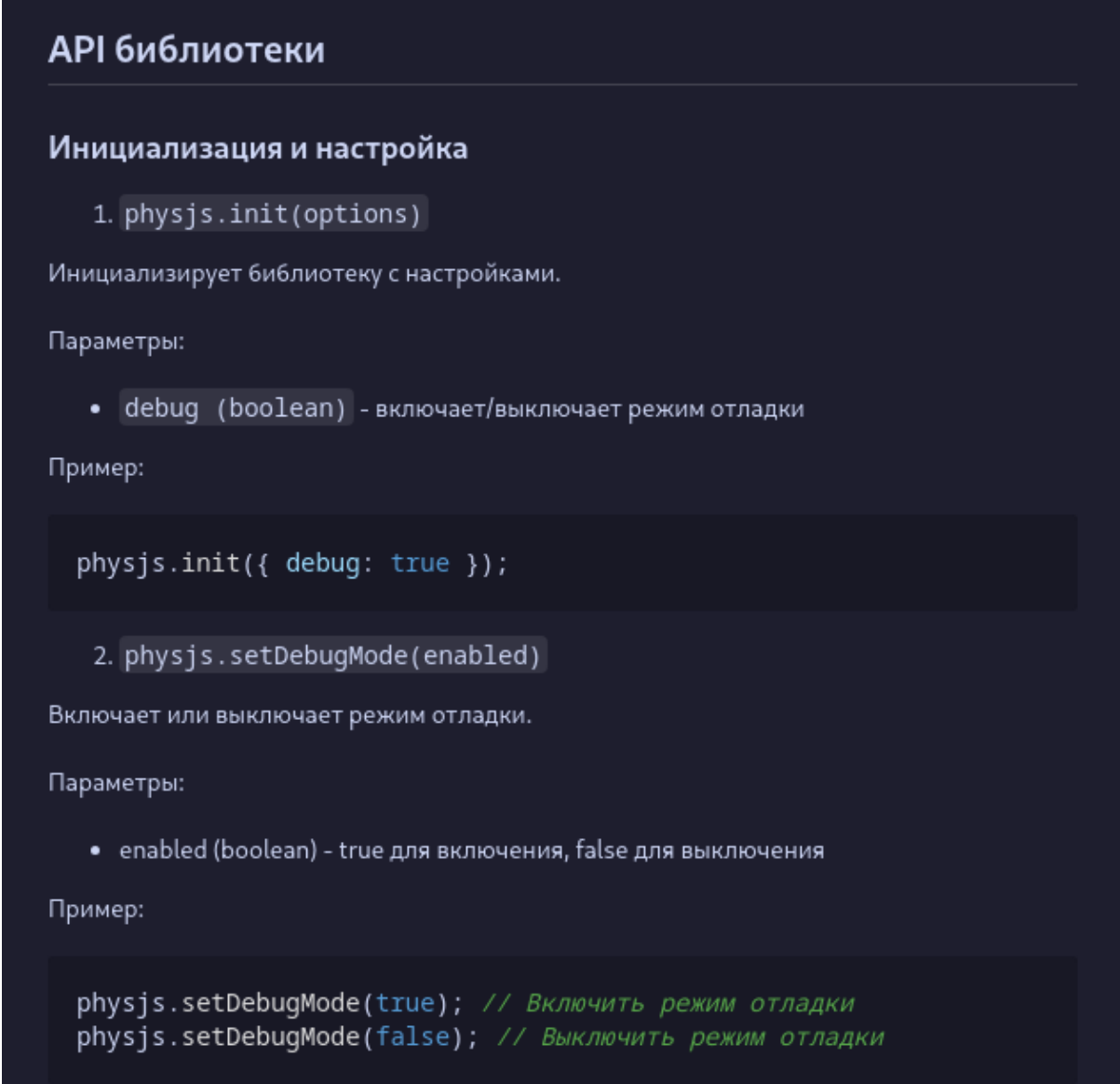
    except json.JSONDecodeError as e:
        return jsonify({"error": f"JSON parsing error: {str(e)}"}), 500
    except Exception as e:
        return jsonify({"error": f"Failed to get tags: {str(e)}"}), 500

```

Рисунок 16 – Применение фреймворка Flask для работы с тэгами.

Однако сам по себе прямой запуск сервера через Flask является лишь dev-решением, о чем и предупреждается пользователь при запуске приложения. Для полноценного запуска было необходимо подобрать соответствующий WSGI-сервер, и наш выбор пал на gunicorn. Применение этого сервера также позволило нам без проблем привязать приложение к сокету. Для поддержания постоянной работы сервера веб-приложение было запущено как системный сервис на VPS. Для вывода приложения в сеть нами использовался Nginx с плагином на автоматическую выдачу SSL сертификатов Let's Encrypt.

Завершающим штрихом в работе команды стало оформление документации к библиотеке на рисунке 17. В ней были подробно описаны все существующие методы и типы объектов, приведены примеры кода для каждого из них, а также разобраны все входные параметры, ожидаемые методом. Помимо прочего, мы начали реализовывать FAQ с решениями самых распространенных проблем, связанных с библиотекой и ее работой.



API библиотеки

Инициализация и настройка

1. `physjs.init(options)`

Инициализирует библиотеку с настройками.

Параметры:

- `debug (boolean)` - включает/выключает режим отладки

Пример:

```
physjs.init({ debug: true });
```

2. `physjs.setDebugMode(enabled)`

Включает или выключает режим отладки.

Параметры:

- `enabled (boolean)` - true для включения, false для выключения

Пример:

```
physjs.setDebugMode(true); // Включить режим отладки  
physjs.setDebugMode(false); // Выключить режим отладки
```

Рисунок 17 – Документация к библиотеке.

Список использованных источников

1. Лин М. В. Современный дизайн. Пошаговое руководство. – 2010.
2. Хвостенко Т. М., Велисар Д. С. Figma-перспективный инструмент современного веб-дизайнера //Вестник образовательного консорциума Среднерусский университет. Информационные технологии. – 2019. – №. 2. – С. 7-10.
3. Андреев Г. Дизайн-мышление. Создание инноваций. – Litres, 2025.
4. Фельке-Моррис Т. Большая книга веб-дизайна. – Litres, 2022.
5. Аббасов И. Б. и др. Дизайн-проекты: от идеи до воплощения. – 2021.
6. Figma Community – <https://www.figma.com/community>
7. MDN Web Docs – <https://developer.mozilla.org/en-US/>
8. Руководство для начинающих nginx – https://nginx.org/ru/docs/beginners_guide.html
9. Welcome to Flask – <https://flask.palletsprojects.com/en/stable/>
10. Введение в WSGI-серверы – <https://habr.com/ru/articles/426957/>
11. Документация по GitHub Pages – <https://docs.github.com/ru/pages>
12. Использование Systemctl для управления службами Systemd – <https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units-ru>
13. Яндекс Практикум – <https://practicum.yandex.ru/learn/uxui-design-for-developers/courses>