НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ «СИНЕРГИЯ»

Факультет Интернет-профессий
(наименование факультета/ института)

Направление подготовки /специальность: 09.03.02 Информационные системы и технологии
(код и наименование направления подготовки /специальности)

Профиль/специализация: Веб-разработка
(наименование профиля/специализации)

Форма обучения: Заочная
(очная, очно-заочная, заочная)

УТВЕРЖДАЮ
Декан факультета

Э.Р.Жданов
(ФИО)

Университет "Синергия" ФАКУЛЬТЕТ ИНТЕРНЕТ-ПРОФЕССИЙ

(Подпись)

«____» _____ 20___ г.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА УЧЕБНУЮ ПРАКТИКУ
(вид практики)

ОЗНАКОМИТЕЛЬНАЯ ПРАКТИКА
(тип практики)

обучающегося группы ОБИ-32309ЕЖЕвр (Шифр и № группы) Гришин Александр Андреевич (ФИО обучающегося)

Место прохождения практики:
»
(наименование Профильной организации)

Срок прохождения практики: с 29.06.2024 г. по 09.08.2024 г.

Сессию сдал!

Содержание индивидуального задания на практику:

№ п/п	Виды работ
1.	Инструктаж по соблюдению правил противопожарной безопасности, правил охраны труда, техники безопасности, санитарно-эпидемиологических правил и гигиенических нормативов.
2.	Выполнение определенных практических кейсов-задач, необходимых для оценки знаний, умений, навыков и (или) опыта деятельности по итогам учебной (ознакомительная практика) практики. (вид практики, тип практики)
2.1.	Кейс-задача № 1 Приведите и опишите подробный анализ по следующим вопросам: ➔ Рассмотрите и опишите какие языки программирования необходимо знать для работы в сфере веб-разработки? ➔ Какие принципы и паттерны программирования широко используются при создании веб-приложений?
2.2.	Кейс-задача № 2 Стилистическое преобразование чисел: ➔ Напишите программу, которая запрашивает у пользователя последовательно день его рождения, месяц и год; ➔ Напишите функцию, которая определяет какому дню недели соответствует эта дата? ➔ Напишите функцию, которая определяет - високосный это был год, или нет? ➔ Напишите функцию, которая определяет сколько сейчас лет пользователю; ➔ Реализуйте вывод в консоль даты рождения пользователя в формате дд мм гггг, где цифры прорисованы звёздочками (*), как на электронном табло. Ответом на задание будет ссылка на репозиторий GitHub, где хранится Ваша программа.
2.3.	Кейс-задача № 3 Создайте простой слайдер изображений. ➔ Создайте интерфейс с помощью HTML и CSS, состоящий из области отображения изображений и кнопок "вперед" и "назад"; ➔ Стилизуйте интерфейс, чтобы он был привлекательным: добавьте рамки, тени, выберите подходящие цвета и шрифты; ➔ Реализуйте функционал смены изображений с помощью ванильного JavaScript и слушателей событий. При нажатии на кнопку "вперед" должно отображаться следующее изображение, при нажатии на кнопку "назад" - предыдущее; ➔ Добавьте функционал, который будет обеспечивать зацикливание слайдера, то есть после последнего изображения снова отображается первое, а перед первым - последнее; ➔ Дополните интерфейс возможностью отображения текущего номера изображения (например, "Изображение 1 из 5") для удобства пользователя. Ответом на задание будет ссылка на репозиторий GitHub, где хранится Ваша программа.
2.4.	Кейс-задача № 4 Проведите анализ выполненной кейс-задачи №3, по следующим критериям: ➔ Определение требований: В этом шаге мы определяем основные требования к слайдеру изображений. Например, какие типы изображений он должен

№ п/п	Виды работ
	поддерживать, сколько изображений должно быть отображено одновременно, должна ли быть возможность прокрутки влево и вправо, должны ли быть кнопки навигации и т.д. ➔ Анализ существующих решений: На этом этапе мы исследуем уже существующие решения для создания слайдеров изображений. Мы можем рассмотреть различные библиотеки и фреймворки, такие как jQuery Slider, Swiper, Owl Carousel и другие. Это поможет нам понять, какие функции и возможности доступны в этих инструментах, и какие из них лучше всего подходят для наших требований. ➔ Выбор инструмента: После анализа существующих решений мы выбираем наиболее подходящий инструмент для нашей задачи. Мы учитываем такие факторы, как функциональность, удобство использования, поддержка сообщества, документация и лицензирование. ➔ Разработка прототипа: На этом этапе мы создаем прототип слайдера изображений, используя выбранный инструмент. Мы тестируем его на различных устройствах и браузерах, чтобы убедиться, что он работает корректно и соответствует нашим требованиям. ➔ Тестирование и отладка: После разработки прототипа мы проводим тестирование, чтобы выявить и исправить любые ошибки или проблемы. Мы также проверяем совместимость с различными версиями браузеров и устройствами. ➔ Документация и обучение: После того, как слайдер готов, мы документируем процесс его создания, чтобы другие члены команды могли легко его использовать и поддерживать. Мы также проводим обучение для команды, чтобы они могли эффективно использовать новый инструмент. ➔ Мониторинг и поддержка: После внедрения слайдера мы продолжаем мониторить его работу и собирать обратную связь от пользователей. Мы также регулярно обновляем его, чтобы он оставался совместимым с новыми версиями браузеров и устройствами. Этот план может быть адаптирован в зависимости от конкретных задач и требований.
2.5.	Кейс-задача № 5 На основе проведенного анализа в кейс-задаче №4 предложите способы решения выявленных проблем в ходе выполнения кейс-задачи №3.
3.	Систематизация собранного нормативного и фактического материала.
4.	Оформление отчета о прохождении практики.
5.	Защита отчета по практике.

Обучающийся индивидуальное задание получил

Гришин Александр Андреевич _____
(ФИО) (Подпись)

29.06.2024 г.

Сессию сдал!

НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ «СИНЕРГИЯ»

Факультет Интернет-профессий
(наименование факультета/ института)

Направление подготовки /специальность: 09.03.02 Информационные системы и технологии
(код и наименование направления подготовки /специальности)

Профиль/специализация: Веб-разработка
(наименование профиля/специализации)

Форма обучения: Заочная
(очная, очно-заочная, заочная)

ОТЧЕТ

ПО УЧЕБНОЙ ПРАКТИКЕ
(вид практики)

ОЗНАКОМИТЕЛЬНАЯ ПРАКТИКА
(тип практики)

Обучающийся Гришин Александр Андреевич _____
(ФИО) (подпись)

Ответственное лицо от Профильной организации
М.П. (при наличии) _____ _____
(ФИО) (подпись)

Москва 2024 г.

Практические кейсы-задачи, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности по итогам практики

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
Кейс-задача № 1	Основные языки программирования для веб-разработки: 1) HTML (HyperText Markup Language): Описание – язык разметки, используемый для создания структуры веб-страниц. Задачи – создание основного каркаса страницы, структурирование контента, использование тегов для текстов, изображений, ссылок и других элементов. 2) CSS (Cascading Style Sheets): Описание – язык стилей, используемый для описания внешнего вида веб-страниц. Задачи – определение оформления HTML-элементов, управление компоновкой, цветами, шрифтами, анимациями и адаптивным дизайном. 3) JavaScript: Описание – скриптовый язык программирования, который используется для создания динамического контента на веб-страницах. Задачи – валидация форм, создание интерактивных элементов, асинхронные запросы (AJAX), манипуляции с DOM, реализация клиентской логики. Серверные языки программирования: 1) PHP: Описание – скриптовый язык общего назначения, особенно подходящий для веб-разработки. Задачи – генерация динамического HTML-кода, обработка форм, работа с базами данных.

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	2) Python: Описание – высокоуровневый язык программирования, известный своей читаемостью и простотой. Задачи – создание серверной логики, API, работа с фреймворками Django и Flask. 3) Ruby: Описание – динамический, интерпретируемый язык программирования. Задачи – веб-разработка с использованием фреймворка Ruby on Rails, создание серверной логики, работа с базами данных. 4) Java: Описание – компилируемый язык программирования с поддержкой многопоточности. Задачи – создание корпоративных веб-приложений, серверная логика, работа с фреймворками Spring и JavaServer Faces (JSF). 5) Node.js: Описание – среда выполнения JavaScript вне браузера. Задачи – серверная логика на JavaScript, создание API, обработка асинхронных операций, использование фреймворка Express.js. Принципы и паттерны программирования в веб-разработке: 1) MVC (Model-View-Controller): Описание – паттерн проектирования, разделяющий приложение на три основные компоненты: модель (данные и бизнес-логика), вид (представление данных) и контроллер (обработка запросов и взаимодействие между моделью и видом). Применение – повышает модульность и тестируемость кода. 2) REST (Representational State Transfer):

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	Описание – архитектурный стиль для создания веб-сервисов. Принципы – использование HTTP-методов (GET, POST, PUT, DELETE), ресурсная ориентированность, stateless взаимодействие. Применение – создание API для взаимодействия между клиентом и сервером. 3) DRY (Don't Repeat Yourself): Описание – принцип программирования, призывающий избегать дублирования кода. Применение – повышение читаемости и поддержки кода, использование функций, модулей и библиотек. 4) KISS (Keep It Simple, Stupid): Описание – принцип, предлагающий сохранять решения простыми и избегать излишней сложности. Применение – упрощение кода и архитектуры, повышение понятности и легкости сопровождения. 5) SOLID: Описание – набор принципов объектно-ориентированного программирования. Принципы: - S (Single Responsibility Principle): каждый класс должен иметь одну обязанность; - O (Open/Closed Principle): классы должны быть открыты для расширения, но закрыты для модификации; - L (Liskov Substitution Principle): объекты должны быть заменяемы их подклассами без нарушения работы программы; - I (Interface Segregation Principle): клиенты не должны зависеть от интерфейсов, которые они не используют;

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	- D (Dependency Inversion Principle): модули верхнего уровня не должны зависеть от модулей нижнего уровня, оба должны зависеть от абстракций. Применение – улучшение структуры кода, повышение гибкости и расширяемости приложений. 6) CQRS (Command Query Responsibility Segregation): Описание – паттерн, разделяющий операции чтения и записи данных. Применение – повышение производительности и масштабируемости приложений, улучшение управления сложными данными. 7) Event-Driven Architecture (EDA): Описание – архитектурный паттерн, основанный на обработке событий. Применение – асинхронные приложения, системы реального времени, повышение отзывчивости и масштабируемости.

Кейс-задача № 2	Листинг:
	
	`from datetime import datetime, date`
	
	`# Функция для определения дня недели`
	`def get_weekday(day, month, year):`
	`week_days = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']`
	`birth_date = datetime(year, month, day)`
	`return week_days[birth_date.weekday()]`
	
	`# Функция для определения високосного года`
	`def is_leap_year(year):`
	`if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):`
	`return True`
	`return False`
	
	`# Функция для определения возраста пользователя`
	`def get_age(day, month, year):`
	`today = date.today()`

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи

```
        birth_date = date(year, month, day)
        age = today.year - birth_date.year - ((today.month, today.day) < (birth_date.month, birth_date.day))
        return age

    # Функция для вывода цифр звездочками
    def print_number_stars(number):
        digits = [
            [" *** ", "*   *", "*   *", "*   *", " *** "],  # 0
            ["  *  ", " **  ", "  *  ", "  *  ", "*****"],  # 1
            [" *** ", "*   *", "    * ", "  *  ", "*****"],  # 2
            [" *** ", "*   *", "  ** ", "*   *", " *** "],  # 3
            ["   * ", "  ** ", " * * ", "*****", "   * "],  # 4
            ["*****", "*    ", "**** ", "    *", "**** "],  # 5
            [" *** ", "*    ", "**** ", "*   *", " *** "],  # 6
            ["*****", "    *", "   * ", "  *  ", " *   "],  # 7
            [" *** ", "*   *", " *** ", "*   *", " *** "],  # 8
            [" *** ", "*   *", " ****", "    *", " *** "]   # 9
        ]
        for row in range(5):
            line = ''.join(digits[int(digit)][row] + ' ' for digit in number)
            print(line)

    # Запрос данных у пользователя
    day = int(input("Enter the day of your birth (DD): "))
    month = int(input("Enter the month of your birth (MM): "))
    year = int(input("Enter the year of your birth (YYYY): "))

    # Определение дня недели
    weekday = get_weekday(day, month, year)
    print(f"Day of the week: {weekday}")

    # Определение високосного года
    leap_year = is_leap_year(year)
    print(f"Leap year: {'Yes' if leap_year else 'No'}")

    # Определение возраста пользователя
    age = get_age(day, month, year)
```

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	print(f"Your age: {age} years") # Вывод даты рождения звездочками print("\nYour birth date in stars:") print_number_stars(f"{day:02}") print_number_stars(f"{month:02}") print_number_stars(f"{year}") Результат: Enter the day of your birth (DD): 19 Enter the month of your birth (MM): 03 Enter the year of your birth (YYYY): 2000 Day of the week: Sunday Leap year: Yes Your age: 24 years Your birth date in stars: * *** ** * * * **** * * ***** *** *** *** * * * * * * ** * * * * *** *** *** *** *** *** ***** *** *** ***
Кейс-задача № 3	Листинг: index.html: <!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Image Slider</title> <link rel="stylesheet" href="styles.css"> </head> <body> <img src="image1.jpg" alt="Image 1" class="slide"> <img src="image2.jpg" alt="Image 2" class="slide"> <img src="image3.jpg" alt="Image 3" class="slide">

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи

```
            <img src="image4.jpg" alt="Image 4" class="slide">
            <img src="image5.jpg" alt="Image 5" class="slide">
        </div>
        <button id="prev" class="nav-btn">←</button>
        <button id="next" class="nav-btn">→</button>
        <div class="caption">
            <span id="caption-text">Image 1 of 5</span>
        </div>
    </div>
    <script src="script.js"></script>
</body>
</html>
```

styles.css:

```
body {
    font-family: Arial, sans-serif;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    background-color: #f0f0f0;
    margin: 0;
}

.slider-container {
    position: relative;
    width: 600px;
    height: 400px;
    border: 2px solid #ccc;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
    overflow: hidden;
    background-color: #fff;
}

.slider {
    display: flex;
    transition: transform 0.5s ease-in-out;
}

.slide {
    min-width: 100%;
    height: 100%;
    object-fit: cover;
    display: none;
}

.slide.active {
```

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи

```
        display: block;
    }

    .nav-btn {
        position: absolute;
        top: 50%;
        transform: translateY(-50%);
        background-color: rgba(255, 255, 255, 0.8);
        border: none;
        font-size: 24px;
        cursor: pointer;
        padding: 10px;
        border-radius: 50%;
        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
        transition: background-color 0.3s;
    }

    #prev {
        left: 10px;
    }

    #next {
        right: 10px;
    }

    .nav-btn:hover {
        background-color: rgba(255, 255, 255, 1);
    }

    .caption {
        position: absolute;
        bottom: 10px;
        width: 100%;
        text-align: center;
        background-color: rgba(0, 0, 0, 0.5);
        color: #fff;
        padding: 10px 0;
    }
```

script.js:

```
    let currentIndex = 0;
    const slides = document.querySelectorAll('.slide');
    const captionText = document.getElementById('caption-text');

    document.getElementById('next').addEventListener('click',   ()
=> {
        slides[currentIndex].classList.remove('active');
```

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи

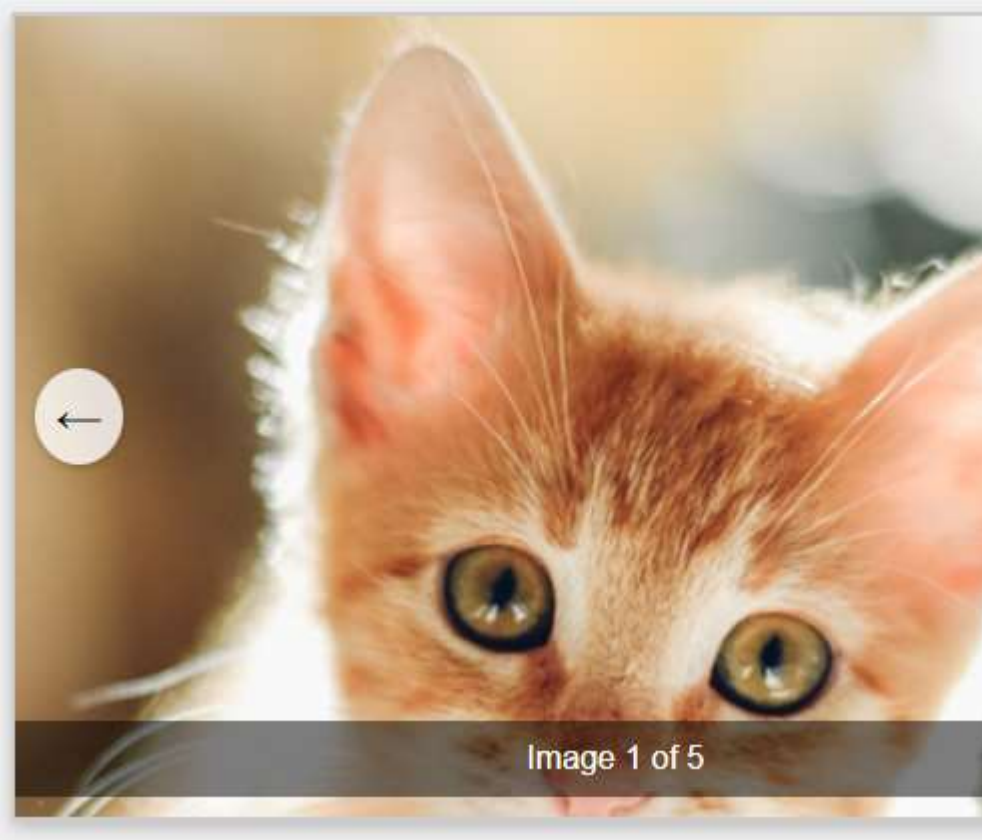
```
        currentIndex = (currentIndex + 1) % slides.length;
        slides[currentIndex].classList.add('active');
        updateCaption();
    });

    document.getElementById('prev').addEventListener('click', () => {
        slides[currentIndex].classList.remove('active');
        currentIndex = (currentIndex - 1 + slides.length) % slides.length;
        slides[currentIndex].classList.add('active');
        updateCaption();
    });

    function updateCaption() {
        captionText.textContent = `Image ${currentIndex + 1} of ${slides.length}`;
    }

    // Инициализация первого слайда
    slides[currentIndex].classList.add('active');
    updateCaption();
```

Результат:



№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
Кейс-задача № 4	Определение требований: 1) типы изображений – слайдер должен поддерживать стандартные форматы изображений (JPEG, PNG, GIF). Для нашего примера использовались изображения формата JPG; 2) количество изображений – в слайдере могут быть использованы любые количества изображений. В нашем примере их пять, но слайдер легко адаптируется к другим количествам; 3) отображение изображений – слайдер отображает по одному изображению за раз; 4) прокрутка – реализована возможность прокрутки влево и вправо с помощью кнопок; 5) кнопки навигации – включены кнопки "вперед" и "назад". Анализ существующих решений: 1) jQuery Slider – предлагает мощные возможности и плагины для создания слайдеров, однако требует зависимости от jQuery; 2) Swiper – популярный слайдер с множеством функций, таких как поддержка мобильных устройств и адаптивность.; 3) Owl Carousel – легкий в использовании и хорошо документированный слайдер с поддержкой множества функций. Каждое из этих решений обладает своими преимуществами и недостатками. В нашем случае, для простоты и демонстрации базовых возможностей, был выбран подход с использованием ванильного JavaScript, что устраняет зависимости и обеспечивает базовую функциональность. Выбор инструмента: 1) функциональность – основная функциональность включала отображение изображений, кнопки для навигации, зацикливание слайдера и отображение текущего изображения;

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	2) удобство использования – ванильный JavaScript был выбран для легкости интеграции и использования; 3) поддержка сообщества и документация – ванильный JavaScript имеет обширную поддержку и документацию; 4) лицензирование – использование ванильного JavaScript не требует специальной лицензии. Разработка прототипа: 1) прототип был создан с использованием HTML для структуры, CSS для стилей и JavaScript для функциональности; 2) было проведено тестирование на различных устройствах и браузерах для обеспечения кроссбраузерной совместимости. Тестирование и отладка: 1) тестирование – прототип был протестирован на различных устройствах (мобильные телефоны, планшеты, настольные компьютеры) и в различных браузерах (Chrome, Firefox, Safari, Edge); 2) отладка – были устранены ошибки, связанные с навигацией и отображением изображений, обеспечена корректная работа кнопок "вперед" и "назад". Документация и обучение: 1) документация – код был написан с комментариями для легкости понимания. Можно было бы создать дополнительную документацию с описанием каждой функции и ее назначения. 2) обучение – проведение обучения для команды было бы необходимо в случае использования более сложного инструмента. Для текущего простого решения с ванильным JavaScript обучение не требуется. Мониторинг и поддержка: 1) мониторинг – после внедрения слайдера следует периодически проверять его работу, особенно после обновлений браузеров;

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	2) обратная связь – сбор отзывов от пользователей поможет выявить возможные улучшения и ошибки; 3) обновления – регулярное обновление слайдера для обеспечения совместимости с новыми версиями браузеров и устройствами.
Кейс-задача № 5	На основе проведенного анализа кейс-задачи №4, можно предложить следующие способы решения выявленных проблем и улучшения реализации слайдера изображений из кейс-задачи №3: 1) улучшение адаптивности и кроссбраузерной совместимости: Проблема – возможные проблемы с отображением на различных устройствах и в разных браузерах. Решение – использовать CSS Flexbox или Grid для более гибкого и адаптивного размещения изображений и элементов управления. Провести дополнительные тесты на различных устройствах и браузерах, используя инструменты для тестирования адаптивности, такие как BrowserStack или Sauce Labs. 2) повышение производительности: Проблема – медленная загрузка изображений может ухудшить пользовательский опыт. Решение – использовать lazy loading (ленивую загрузку) для изображений, чтобы загружать их по мере необходимости. Можно использовать атрибут loading="lazy" для тегов <img>, чтобы браузеры автоматически загружали изображения, когда они становятся видимыми. 3) расширение функциональности: Проблема – ограниченная функциональность (например, нет автоматической прокрутки, отсутствует поддержка жестов для мобильных устройств). Решение – добавить автоматическую прокрутку слайдов с возможностью включения/выключения. Реализовать поддержку жестов

№ п/п	Подробные ответы обучающегося на практические кейсы-задачи
	для мобильных устройств с помощью библиотек, таких как Hammer.js. 4) улучшение пользовательского интерфейса: Проблема – простой дизайн интерфейса. Решение – добавить анимацию переходов между слайдами с помощью CSS-трансформаций и анимаций. Использовать более привлекательные кнопки навигации и индикаторы текущего слайда. 5) обработка ошибок и уведомления: Проблема – нет обработки ошибок при загрузке изображений. Решение – добавить обработку ошибок для изображений (например, замена на placeholder-изображение при ошибке загрузки) и уведомления для пользователя о возможных проблемах. 6) улучшение кода и структуры проекта: Проблема – код может быть улучшен для повышения читаемости и расширяемости. Решение – организовать код в виде модулей. Использовать ES6+ синтаксис, такие как классы и модули. Добавить комментарии и улучшить документацию.

Дата: 09.08.2024 _____ Гришин Александр Андреевич

(подпись) (ФИО обучающегося)

Сессию сдал!