ВСЕРОССИЙСКОЕ
ЧЕМПИОНАТНОЕ
ДВИЖЕНИЕ
ПО ПРОФЕССИОНАЛЬНОМУ
МАСТЕРСТВУ

КОНКУРСНОЕ ЗАДАНИЕ КОМПЕТЕНЦИИ

«Веб-технологии»

Регионального этапа **чемпионата**

по профессиональному мастерству

«Профессионалы» в 2026 **г.**

в Республике Калмыкия

2025 г

Конкурсное задание разработано экспертным сообществом и утверждено Менеджером компетенции, в котором установлены нижеследующие правила и необходимые требования владения профессиональными навыками для участия в соревнованиях по профессиональному мастерству.

Конкурсное задание включает в себя следующие разделы:

1. ОСНОВНЫЕ ТРЕБОВАНИЯ КОМПЕТЕНЦИИ 3
1.1. ОБЩИЕ СВЕДЕНИЯ О ТРЕБОВАНИЯХ КОМПЕТЕНЦИИ 3
1.2. ПЕРЕЧЕНЬ ПРОФЕССИОНАЛЬНЫХ ЗАДАЧ СПЕЦИАЛИСТА ПО КОМПЕТЕНЦИИ «ВЕБ-ТЕХНОЛОГИИ» 3
1.3. ТРЕБОВАНИЯ К СХЕМЕ ОЦЕНКИ 9
1.4. СПЕЦИФИКАЦИЯ ОЦЕНКИ КОМПЕТЕНЦИИ 9
1.5. КОНКУРСНОЕ ЗАДАНИЕ 10
1.5.1. Разработка/выбор конкурсного задания
1.5.2. Структура модулей конкурсного задания (инвариант/вариатив) 11
2. СПЕЦИАЛЬНЫЕ ПРАВИЛА КОМПЕТЕНЦИИ 40
2.1. Личный инструмент конкурсанта 40
2.2. Материалы, оборудование и инструменты, запрещенные на площадке 41

ИСПОЛЬЗУЕМЫЕ СОКРАЩЕНИЯ

1. ФГОС – Федеральный государственный образовательный стандарт
2. ПС – Профессиональный стандарт
3. КЗ – Конкурсное задание
4. ИЛ – Инфраструктурный лист

1. ОСНОВНЫЕ ТРЕБОВАНИЯ КОМПЕТЕНЦИИ

1.1. Общие сведения о требованиях компетенции

Требования компетенции (ТК) «Веб-технологии» определяют знания, умения, навыки и трудовые функции, которые лежат в основе наиболее актуальных требований работодателей отрасли.

Целью соревнований по компетенции является демонстрация лучших практик и высокого уровня выполнения работы по соответствующей рабочей специальности или профессии.

Требования компетенции являются руководством для подготовки конкурентоспособных, высококвалифицированных специалистов / рабочих и участия их в конкурсах профессионального мастерства.

В соревнованиях по компетенции проверка знаний, умений, навыков и трудовых функций осуществляется посредством оценки выполнения практической работы.

Требования компетенции разделены на четкие разделы с номерами и заголовками, каждому разделу назначен процент относительной важности, сумма которых составляет 100.

1.2. Перечень профессиональных задач специалиста по компетенции «Веб-технологии»

Перечень видов профессиональной деятельности, умений, знаний и профессиональных трудовых функций специалиста (*из ФГОС/ПС/ЕТКС*) базируется на требованиях современного рынка труда к данному специалисту.

Таблица №1

Перечень профессиональных задач специалиста

№ п/п	Раздел	Важность в %
1	**Охрана труда**	3
	Специалист должен знать и понимать: − нормативно-правовую базу; − Опасные и вредные производственные факторы на	

	своем рабочем месте; − Требования к организации рабочего места; − Способы и средства защиты. − Действия в чрезвычайных ситуациях	
	Специалист должен уметь: − Практически организовывать свое рабочее место в соответствии с эргономическими требованиями. − Настраивать программное. − Применять на практике приемы и методы снятия напряжения. − Соблюдать режим труда и отдыха. − Безопасно эксплуатировать электрооборудование	
2	**Бережливое производство**	7
	Специалист должен знать и понимать: − принципы и практики, которые позволяют продуктивно работать, в том числе в команде; − аспекты систем, которые позволяют повысить продуктивность и выработать оптимальную стратегию; − основные принципы выбора технологий и инструментария для решения поставленных задач (проектов); - основные подходы к планированию и документированию проекта.	
	Специалист должен уметь: − формировать архитектуру проекта (программного продукта) в соответствии с последними отраслевыми решениями; − выбирать технологии и инструменты для решения поставленных задач; − планировать график рабочего дня с учетом требований; − планировать задачи, учитывать временные ограничения и сроки; − решать распространенные задачи веб-дизайна и разработки кода; − формировать тестовые наборы, применять инструменты автоматического тестирования; − производить отладку кода программ и находить ошибки; − оптимально использовать компьютерное оборудование и программное обеспечение для повышения эффективности своей работы; − использовать менеджеры пакетов при разработке проекта; - использовать систему контроля версий.	
3	**Разработка интерфейса пользователя**	26
	Специалист должен знать и понимать: − структуру и общепринятые элементы веб-страниц	

	различных видов и назначений; − основные принципы организации контента веб-приложения; − основные правила выбора цвета, работы с типографикой и композицией; − принципы и методы создания и адаптации графики для использования ее на веб-сайтах; − ограничения, которые накладывают мобильные устройства и разрешения экранов при использовании их для просмотра веб-сайтов; − принципы построения эстетичного и креативного дизайна; − методы обеспечения доступа к страницам веб-сайтов аудитории с ограниченными возможностями; − World Wide Web Consortium (W3C) стандарты HTML и CSS; − методы верстки веб-сайтов и их стандартную структуру; − Web accessibility initiative (WAI) стандарт доступности активных Интернет-приложений для людей с ограниченными возможностями; − основные принципы применения соответствующих CSS правил и селекторов для получения ожидаемого результата; − лучшие практики для Search Engine Optimization (SEO) и интернет-маркетинга;	
	Специалист должен уметь: − создавать, использовать и оптимизировать изображения для веб-сайтов; − выбирать дизайнерское решение, которое будет наиболее подходящим для целевого рынка; − принимать во внимание влияние каждого элемента, который добавляется в проект во время разработки дизайна; − использовать все требуемые элементы при разработке дизайна; − создавать «отзывчивый» дизайн, который будет отображаться корректно на различных устройствах и при разных разрешениях; − создавать html-страницы сайта на основе предоставленных графических макетов их дизайна; − корректно использовать CSS для обеспечения единого дизайна в разных браузерах; − создавать адаптивные веб-страницы, которые способны оставаться функциональными на различных устройствах при разных разрешениях; − создавать веб-сайты полностью соответствующие текущим стандартам W3C (http://www.w3.org);	

	− создавать и модифицировать веб-интерфейсы с учетом принципов Search Engine Optimization; − использовать препроцессоры.	
4	**Разработка на стороне клиента**	25
	Специалист должен знать и понимать: − основные принципы паттерновой разработки веб-приложений; − ECMAScript (JavaScript); − принципы, особенности и способы использования открытых фреймворков; − принципы разработка кода с использованием открытых библиотек; − различные интерфейсы взаимодействия с объектами браузера	
	Специалист должен уметь: − создавать и модифицировать JavaScript код для улучшения функциональности и интерактивности сайта; − манипулировать элементами страницы веб-приложения; − разрабатывать анимацию для повышения доступности и визуальной привлекательности веб-приложения; − применять открытые библиотеки и фреймворки; − тестировать веб-приложение.	
5	**Разработка веб приложения на стороне сервера**	25
	Специалист должен знать и понимать: − процедурные и объектно-ориентированные языки PHP, Python, Node.js; − основные принципы и правила использования открытых библиотек и фреймворков; − распространенные модели организации и хранения данных; − основные принципы создания баз данных; − основные принципы обмена данными между клиентом и сервером; − методы работы с протоколами SSH/(s)FTP при подключении к серверам; − способы разработки программного кода в соответствии с паттернами проектирования; − основные принципы обеспечения безопасности веб-приложения.	
	Специалист должен уметь: − разрабатывать процедурный и объектно-ориентированный программный код; − разрабатывать веб-сервисы с применением PHP, Python, Node.js в соответствии с техническим заданием; − создавать библиотеки и модули для выполнения	

	повторяющихся задач; − разрабатывать веб-приложения с доступом к различным базам данных; − создавать SQL (Structured Query Language) запросы и конструкции; − обеспечивать безопасность (устойчивость веб-приложения к атакам и взломам); − интегрировать существующий и создавать новый программный код с API (Application Programming Interfaces); − использовать открытые библиотеки и фреймворки;	
6	**Разработка информационных ресурсов с использованием готовых решений**	10
	Специалист должен знать и понимать: − преимущества и ограничения системы управления контентом с открытым исходным кодом; − методы работы с плагинами/модулями; − способы реализации функциональных возможностей CMS; − основные принципы организации контента веб-приложения; − понимать необходимость поддержания и обновления для плагинов CMS и соответствующих модулей для безопасности системы; основные принципы интеграции с внешними веб-приложениями.	
	Специалист должен уметь: − устанавливать, настраивать и модифицировать систему управления контентом; − устанавливать, настраивать и обновлять плагины/модули CMS; − создать пользовательские темы/шаблоны для системы управления контентом; − создавать пользовательские плагины/модули и шаблоны/темы; − использовать встроенные методы и средства CMS при разработке веб-приложения.	

1.3. Требования к схеме оценки

Сумма баллов, присуждаемых по каждому аспекту, должна попадать в диапазон баллов, определенных для каждого раздела компетенции, обозначенных в требованиях и указанных в таблице №2.

Таблица №2

Матрица пересчета требований компетенции в критерии оценки

Критерий/Модуль					Итого баллов за раздел ТРЕБОВАНИЙ КОМПЕТЕНЦИИ
Разделы ТРЕБОВАНИЙ КОМПЕТЕНЦИИ		А	Б	В	
	1	1	1	1	**3**
	2	2	2	3	**7**
	3	24		2	**26**
	4			25	**25**
	5		25		**25**
	6	6	5	3	**14**
Итого баллов за критерий/модуль		**33**	**33**	**34**	**100**

1.4. Спецификация оценки компетенции

Оценка Конкурсного задания будет основываться на критериях, указанных в таблице №3:

Таблица №3

Оценка конкурсного задания

Критерий		Методика проверки навыков в критерии
А	**Разработка интерфейса пользователя**	Проверка результата выполнения модуля после его завершения
Б	**Разработка Веб-приложения на стороне сервера**	Проверка результата выполнения модуля после его завершения
В	**Разработка Веб-приложения на стороне клиента**	Проверка результата выполнения модуля после его завершения

1.5. СОДЕРЖАНИЕ КОНКУРСНОГО ЗАДАНИЯ

Общая продолжительность Конкурсного задания: 18 ч.

Количество конкурсных дней: 3 дня

Вне зависимости от количества модулей, КЗ должно включать оценку по каждому из разделов требований компетенции.

Оценка знаний конкурсанта должна проводиться через практическое выполнение Конкурсного задания. В дополнение могут учитываться требования работодателей для проверки теоретических знаний / оценки квалификации.

1.5.1. Разработка/выбор конкурсного задания

Конкурсное задание состоит из 3 модулей, включает обязательную к выполнению часть (инвариант) - 3 модуля. Общее количество баллов конкурсного задания составляет 100.

Обязательная к выполнению часть (инвариант) выполняется всеми регионами без исключения на всех уровнях чемпионатов. Вариативная часть может подвергаться изменениям, в зависимости от потребностей региона в технологиях и специалистах.

В случае если ни один из модулей вариативной части не подходит под запрос работодателя конкретного региона, то вариативный(е) модуль(и) формируется(ются) регионом самостоятельно под запрос работодателя. Исключать вариативную часть из конкурсного задания запрещается. Допускается объединение вариативных модулей, однако общее время, отведенное на выполнение вариативного(ых) модуля(ей) и количество баллов в критериях оценки по аспектам не изменяются (Приложение 2 Матрица конкурсного задания).

1.5.2. Структура модулей конкурсного задания (инвариант/вариатив)

Модуль А. Разработка интерфейса пользователя (Инвариант)

Время на выполнение модуля: 6 часов

Стек: HTML, CSS

Задания:

Концепция сервиса

“Тюльпан” — это компания, предоставляющая услуги пассажирских перевозок. Сервис объединяет:

- выбор удобной даты поездки в период проведения фестиваля;
- выбор удобного и комфортабельного вид транспорта для поездки на фестиваль;

Целевая аудитория

- молодые люди всех возрастов.

Требования к проекту

1. Адаптивность
   1.1. Сайт ориентирован для пользователей мобильных устройств, но должен корректно отображаться на всех устройствах.
   1.2. Минимум 4 структурных изменения при переходе от мобильной (360px) к планшетной (768px), и 4 структурных изменения — от планшетной к десктопной версии (1920px) для главной страницы. Все страницы должны корректно отображаться на всех устройствах.
2. Запрет на фреймворки
   2.1. Запрещено использование Bootstrap, Tailwind и других CSS-фреймворков
3. Система контроля версий
   3.1. Обязательное использование Git
   3.2. Минимум 4 осмысленных коммита
4. Для главной страницы должны быть разработаны светлая и темная темы.
   4.1. По умолчанию используется темная тема
   4.2. Минимум 4 изменения в дизайне страницы

Структура проекта

Ваша задача – сверстать следующие страницы веб-сайта:

1. Главная страница

2. Личный кабинет
3. Страница входа
4. Страница регистрации
5. Страница личного кабинета
6. Страница с результатами поиска
7. Страница бронирования
8. Страница выбора мест в выбранном транспорте.

Детализация страниц

1. Главная страница — Landing Page

Главная страница должна содержать следующие блоки:

1.1.2. Логотип (должен быть разработан в векторе с использованием представленных цветов в медиа).

1.1.3. Меню навигации:

1.3.1. Поиск (якорь на раздел главной страницы)

1.3.2. Кнопка регистрации/авторизации («Войти»)

1.3.3. Личный кабинет

1.3.4. Выбор транспорта

1.3.5. Выбор даты бронирования

1.3.6. Акции

В мобильной и планшетной версии меню навигации должно быть представлено в виде бургера, в десктопной — отображаются все элементы.

1.1.4. Слайдер с новинками игр, количество слайдов — не менее 3. Каждый слайд должен содержать изображение, название игры, стоимость, кнопка-ссылка «Подробнее» (контент представлен в медиа). Слайдер должен содержать индикацию активного слайда.

1.1.5. Быстрый поиск по жанрам. Предусматривает наличие одного поля ввода с подсказками и кнопки для поиска.

1.1.6. Инфографика о преимуществах маркетплейса:

1.6.1. Привлекательные цены

1.6.2. Ежемесячные акции

1.6.3. Круглосуточная поддержка

1.6.4. Быстрая покупка

Около каждого из пунктов заказчик желает, чтобы отображалось соответствующее графическое изображение.

1.1.7. Карточки с популярными играми (не менее 5). В каждой карточке отображается изображение игры, название игры, жанр, компания-разработчик, рейтинг в виде количества заполненных звездочек, стоимость, ссылка-кнопка для перехода на страницу с каталогом игр. Также пользователь может переходить в каталог по клику на карточку.

1.1.8. Наши партнеры (не менее 3 логотипов партнеров, представленных из медиа).

1.1.9. Подвал сайта:

1.9.1. Телефон (8-800-100-10-10)

1.9.2. e-mail: tulpan@.ru

1.9.3. Клиентам

1.9.4. Партнерам

1.9.5. О нас

1.9.6. Контакты

1.9.7. Навигация по сайту:

1.9.7.1. Главная

1.9.7.2. Регистрация

1.9.7.3. Авторизация

1.9.7.4. Личный кабинет

1.9.7.5. Поиск (якорь на раздел главной страницы)

2. Страница входа в личный кабинет

На данной странице вам необходимо создать форму со следующими полями:

1.2.1. E-mail

1.2.2. Пароль

1.2.3. Кнопка для входа

1.2.4. Кнопка-ссылка для восстановления пароля

3. Страница регистрации

На данной странице вам необходимо создать форму со следующими полями, обязательными для заполнения:

3.1.2. E-mail

3.1.3. Фамилия

3.1.4. Имя

3.1.5. Дата рождения

3.1.6. Номер телефона

3.1.7. Пароль

3.1.8. Повтор пароля

3.1.9. Элемент, подтверждающий согласие на обработку персональных данных

3.1.10. Кнопка для регистрации

4. Личный кабинет

4.1. Информация о пользователе (E-mail, фамилия, имя, дата рождения, номер телефона — контент представлен в медиа)

4.2. История покупок:

4.2.2. Дата покупки

4.2.3. Стоимость

4.3. Настройки профиля

4.3.2. Кнопка для смены пароля

4.3.3. Элемент управления для выбора темы (темная/светлая)

5. Страница с каталогом игры

Страница должна содержать игры в виде карточек. Каждая карточка игры должна содержать изображение игры, название, жанр, компания-

разработчик, стоимость, рейтинг в виде количества заполненных звездочек, ссылка-кнопка для перехода на страницу с описанием игры.

Мобильная и планшетная версии страницы должна содержать элемент, обозначающий открытие фильтра, а на десктопной версии должен быть предусмотрен элемент для фильтрации игр по жанрам.

6. Административная панель

   Страница должна содержать форму добавления информации о новой поездке:

   6.1. Даты бронирования

   6.2. Выбор транспорта

   6.3. Акции

   6.4. Стоимость

Технические требования

Общие требования

- Адаптивность (минимум 4 между переходами к каждой точке брейкпоинта)
- Валидный HTML/CSS
- Кроссбраузерная совместимость
- Оптимизированные изображения

Дизайн-принципы

- Представленная цветовая гамма из медиа
- Минималистичный стиль
- Интуитивно понятный интерфейс
- Четкая структура контента
- Анимации и переходы

Требования к коду

- Семантическая вёрстка
- БЭМ методология
- Комментарии к коду

Требования к названию файлов

- index.html (главная)
- login.html (вход)
- register.html (регистрация)
- profile.html (личный кабинет)
- catalog.html (каталог)
- admin.html (админка)
- index_light.html (светлая тема)

Для оформления текста должны использоваться как минимум 2 шрифта, представленных из медиа.

Сверстанный веб-сайт должен быть доступен по адресу http://xxxxxx-m1.web.ru, где xxxxxx - логин конкурсанта (указан на индивидуальной карточке).

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

Все страницы указанные выше должны быть доступны к просмотру по соответствующим адресам: http://xxxxxx-m1.web.ru/index.html, http://xxxxxx-m1.web.ru/login.html и т.д.

Работы, выполненные с использованием CSS-фреймворков, проверяться не будут!

Модуль Б. Разработка Веб-приложения на стороне сервера (Инвариант)

Время на выполнение модуля: 6 часов Стек: PHP (Python), MySQL

Задание

Вам необходимо разработать серверную часть платформы онлайн-обучения. Задание включает в себя:

- разработку административной панели по предоставленному шаблону;
- реализацию REST API по предоставленной спецификации.

Админ-панель

Панель должна быть реализована на основе предоставленного HTML-шаблона.

Функциональность доступна только авторизованному администратору. При работе с формами необходимо реализовать валидацию:

- Поля с ошибками должны быть визуально выделены.
- Под каждым некорректным полем должно отображаться текстовое сообщение об ошибке.

Экран авторизации

Экран авторизации содержит поля для ввода email и пароля, а также кнопку отправки формы.

Оба поля обязательны для заполнения. При вводе некорректных данных отображаются соответствующие ошибки.

После успешной авторизации пользователь перенаправляется на список курсов.

Курсы

Администратор должен иметь возможность управлять курсами:

- создавать, редактировать, удалять курсы;
- просматривать список курсов с пагинацией (по 5 на страницу).

Поля курса:

- **Название курса** — обязательно, максимум 30 символов;
- **Описание курса** — необязательно, максимум 100 символов;
- **Продолжительность (в часах)** — обязательно, целое число не больше 10;
- **Цена** — обязательное, формат «xx.xx», число не менее 100;
- **Дата начала и дата окончание** — обязательное, формат «дд-мм-гггг»;
- **Обложка курса (изображение)** — обязательно при создании, JPG (JPEG), максимум 2000 Кб.

Загружаемая обложка должна автоматически конвертироваться в миниатюру с максимальным размером 300×300 пикселей (с сохранением пропорций, как CSS). И название файла должно начинаться с mpic* с сохранием формата.

Уроки

Каждый курс может содержать не более 5 уроков. Администратор может:

- добавлять, редактировать и удалять уроки;
- каждый урок содержит:
  - **Заголовок** (обязательно, максимум 50 символов),
  - **Текстовое содержание** (обязательно),
  - **Видеоссылку SuperTube** (необязательно, валидная ссылка, https://super-tube.cc/video/v23189),
  - **Длительность** (обязательное, целое число, не более 4 часов).

Удаление уроков возможно только при отсутствии активных записей студентов на курс.

Студенты и записи

В админ-панели также должен отображаться список студентов, записанных на курсы, с фильтрацией по курсу. Для каждого студента отображаются:

- email,
- имя,
- курс,
- дата записи,
- статус оплаты: *ожидает оплаты*, *оплачено, ошибка оплаты*.

После завершения курса администратор может рапеспечатать сертификат студента на «фирменной» бумаге, используя специальный шаблон для бланков в которые будут подставляться все данные пользователя, согласно шаблону (6 первых символа номера сертификата формируется ответом от сервера сертификации). Всё формирование номера сертификата происходит на стороне сервера.

Формирование номера сертификата

Запрос	Ответ

URL: {service_host}/create-sertificate
Method: POST
Headers:
ClientId: {login},
Accept: application/json,
Content-Type: application/json
Body:
{
student_id: {id},
course_id: {id}
}
	Status code: 200
	{
	"course_number": "xxxxxx"
	}

Номер сертификата должен состоять из 6 симвлов (возвращаемых сервисом) и 6 цифр фомируемых разрабатываемой системой (12 символов). Последней цифрой должна быть 1 (действительный).

REST API

API должно предоставлять следующий функционал:

- регистрация и авторизация студентов;
- просмотр списка курсов;
- просмотр уроков курса;
- запись и покупка курса;
- просмотр своих курсов;
- отмена записи на курс;
- проверка сертификата.

Общие требования

- Все запросы и ответы — в формате JSON.
- Авторизация — через заголовок Authorization: Bearer {token}.
- В случае ошибки валидации возвращать:
  Status code: 422

```
{
 "message": "Invalid fields",
  "errors": {
   "field_name": [
     "Error message"
   ]
  }
}
```

- При попытке доступа без авторизации:
  Status code: 403
  ```
  {
   "message": "Forbidden for you"
  }
  ```

Регистрация

Запрос	Ответ

Method: POST
Link: {{api-host}}/registr
Body:
{
“email”: “name@example.com”,
“password”: “password”
}
	Успешный
	Status code: 201
	Body:
	{
	“success”: true
	}

Валидация полей:

- email - обязательное, валидный email, уникальный
- password - обязательное, минимум 3 символа, с содержанием минимум одного символа верхнего и нижнего регистра, одной цифры и один из спецсимволов «_», «#», «!», «%».

Авторизация

Запрос	Пример ответа

Method: POST
Link: {{api-host}}/auth
Body:
{
“email”:
“name@example.com”,
“password”: “password”
}
	Успешный
	Status code: 200
	Body:
	{
	“token”: “{токен авторизации}”
	}
	
	Некорректные данные:
	Status code: 422
	{
	“message”: “Invalid data”,
	“errors”: {
	“email”: [
	“Invalid data”
	]
	}
	}

Валидация полей:

- email - обязательное

- password - обязательное

Получения списка курсов

Запрос	Пример ответа

Method: GET
Headers:
Authorization: Bearer {token}
Link: {{api-host}}/courses
	Успешный
	Status code: 200
	Body:
	{
	“data”: [
	{
	“id”: {идентификатор},
	“name”: {название},
	“description”: {описание},
	“hours”: {количество_часов},
	“img”: {ссылка_на_изображение_курса},
	“start_date”: {дата_по_формату},
	“end_date”: {дата_по_формату},
	“price”: {цена_по_формату}
	},
	… {остальные курсы},
	],
	“pagination”: {
	“total”: {всего_страниц},
	“current”: {текущая_страница},
	“per_page”: {количество_курсов_на_странице}
	}
	}

Курсы должны выводиться с пагинацией

Получения списка уроков

Запрос	Пример ответа
Method: GET Headers: Authorization: Bearer {token} Link: {{api-host}}/courses/{course_id}	**Успешный** Status code: 200 Body: { “data”: [{ “id”: {идентификатор}, “name”: {название}, “description”: {описание},

	“video_link”: {ссылка_на_видео}, “hours”: {количество_часов} }, … {остальные уроки}] }

Запись на курс

Запрос	Пример ответа

Method: POST
Headers:
Authorization: Bearer {token}
Link: {{api-host}}/courses/{course_id}/buy
	Успешный
	Status code: 200
	Body:
	{
	“pay_url”: “{ссылка на оплату курса}”
	}

Запись на курс должна производиться с учётом его дат проведения, то есть нельзя записаться на курс, который уже начался или закончился.

Webhook для платежной системы

Запрос	Пример ответа

Method: POST
Link: {{api-host}}/payment-webhook
Body:
{
“order_id”: {идентификатор заказа},
“status”: “{success | failed}”
}
	Любой
	Status code: 204
	Body: отсутствует

На данный webhook будет отправлен запрос от платежной системы, когда пользователь перейдет по ссылке и вводит реквизиты карты. Статус будет зависеть от того, какие реквизиты введет пользователь. order_id будет тот, который платежная система отправила в ответ на запрос получения ссылки на оплату.

Реквизиты для успешной покупки: 8888 0000 0000 1111

Реквизиты для безуспешной покупки: 8888 0000 0000 2222

Просмотр списка курсов студента

Запрос	Пример ответа

Method: GET
Headers:
Authorization: Bearer {token}
Link: {{api-host}}/orders
	Успешный
	Status code: 200
	Body:
	{
	“data”: [
	{
	“id”: {идентификатор},
	“payment_status”: {pending(ожидает оплаты) | success(оплачен) | failed(ошибка оплаты)},
	“course”: {
	“id”: {идентификатор},
	“name”: {название},
	“description”: {описание},
	“hours”: {количество_часов},
	“img”: {ссылка_на_изображение_курса},
	“start_date”: {дата_по_формату},
	“end_date”: {дата_по_формату},
	“price”: {цена_по_формату}
	}
	},
	{остальные_курсы}
	]
	}

Отмена записи на курс

Запрос	Пример ответа

Method: GET
Headers:
Authorization: Bearer {token}
Link: {{api-host}}/orders/{id}
	Успешный
	Status code: 200
	Body:
	{
	“status”: “success”
	}
	ошибка

	Status code: 418
	Body:
	{
	“status”: “was payed”
	}

Отписаться от курса можно только со статусами «ожидает оплаты» или «ошибка оплаты»

Проверка сертификата

Запрос	Ответ

URL: {service_host}/check-sertificate
Method: POST
Headers:
Accept: application/json,
Content-Type: application/json
Body:
{
sertikate_number: {номер_сертификата},
}
	Status code: 200
	{
	"status": "success | failed"
	}

Номер сертификата, заканчивающийся цифрой 1 — действительный, а 2 - не действительный.

Инструкция

- Админ-панель должна быть доступна по адресу {{site-url}}/course-admin
- API — по адресу {{api-host}} = {{site-url}}/school-api
- {service_host} — предоставлен в media файлах
- Учетные данные администратора:
  - email: admin@edu.com
  - пароль: course2025
- В системе должен использоваться MySQL с предоставленными реквизитами.

- Разрешенные фреймворки: Laravel (10–12), Django (5.1.7), Yii2 (последние версии).
- Работа должна быть развернута на сервере. Проверяться будут только загруженные проекты.

Модуль B. Разработка Веб-приложения на стороне клиента

Время на выполнение модуля: 6 часов
Стек: JavaScript, Vue, React

Задание

Вам необходимо разработать одностраничное приложение (SPA) для создания и совместного использования интерактивных досок. Пользователь должен иметь возможность:

- зарегистрироваться и авторизоваться;
- создавать доски;
- предоставлять доступ (с правом редактирования) к доске другим пользователям по email;
- просматривать доски, к которым у пользователя есть доступ (с правом редактирования);
- совместно с приглашенными пользователями в реальном времени редактировать, взаимодействовать и работать над доской;
- размещать на доске различные объекты: текст, изображение, прямоугольник, круг, линию, с возможностью вращения и управления их размерами (сохраняя соотношение сторон у изображения);
- видеть в реальном времени изменения, совершаемые другими пользователями: захват объектов фокусом (данный объект редактируется другим пользователем, изменения будут видны вам только после снятия фокуса);
- видеть имя пользователя, который взял объект в фокус;
- получать и отправлять данные в формате JSON при изменениях на доске;
- синхронизировать изменения через WebSocket в режиме реального времени;
- генерировать публичную ссылку на просмотр доски, доступную в том числе без авторизации;
- просматривать список публичных досок;

- ставить «лайк» понравившимся доскам;
- фильтровать все публичные доски по количеству «лайков».

Необходимо использовать предоставленный шаблон. Шаблон должен отображаться корректно.

Документация к API и WebSocket будет предоставлена в медиафайлах.

Общие требования

- Приложение должно быть реализовано как SPA.
- Все взаимодействия с сервером осуществляется через REST API и WebSocket.
- При всех действиях должна быть реализована валидация на стороне сервера. Ошибки отображаются рядом с соответствующими полями, поля подсвечиваются.
- Должна быть реализована базовая навигация (например, с использованием React Router или Vue Router).
- Необходимо использовать предоставленный шаблон.
- Авторизация должна использовать токен, передаваемый в заголовке Authorization.

Функциональность приложения

Авторизация и регистрация

Пользователь должен иметь возможность зарегистрироваться и авторизоваться.

Поля регистрации:

- email (валидный);
- имя (только латиница);

- пароль (от 8 символов, должен содержать цифры и спецсимволы).

Поля авторизации:

- email;
- пароль.

После успешной авторизации пользователь попадает на страницу списка публичных досок. После регистрации — на страницу входа.

Управление досками

Пользователь может:

- создавать доски;
- предоставлять доступ (с правом редактирования) к доске другим пользователям по email;
- просматривать список досок, к которым у пользователя есть доступ (с правом редактирования);
- генерировать публичную ссылку на просмотр доски (доступ по /board/{hash} без авторизации);
- просматривать список публичных досок;
- ставить «лайк» понравившимся доскам;
- фильтровать все публичные доски по количеству «лайков».

Работа с доской

На доске пользователь может:

- добавлять объекты: текст, изображение, прямоугольник, круг, линию;
- вращать и перемещать объекты;

- изменять размеры объектов (сохраняя соотношение сторон у изображений);

- редактировать только те объекты, на которые у него фокус;

- получать и отправлять изменения в формате JSON;

- видеть все изменения в реальном времени благодаря WebSocket;

- видеть изменения, совершённые другими (приглашенными) пользователями.

Работа с холстом

- Каждая доска должна содержать холст фиксированного размера **1600 x 900 пикселей**.

- Все объекты (текст, изображения, фигуры) должны размещаться **только внутри границ холста**.

- Пользователю должно быть **запрещено перемещать или масштабировать объекты за пределы холста**.

- Объект в фокусе должен визуально выделяться от других, должно быть понятно кто его редактирует в данный момент (подпись имени пользователя).

- Невозможно установить фокус на объект, который в фокусе у другого пользователя.

- Изменения объекта на холсте применяются только после снятия фокуса с объекта (клик в любой области вне объекта).

- Визуально границы холста должны быть чётко обозначены на экране.

Общий доступ и публичные доски

Если доска сделана публичной, на сервере генерируется hash, доступ к доске осуществляется по адресу вида /board/{hash}. Все пользователи (включая гостей) могут наблюдать изменения в реальном времени через WebSocket.

Инструкция

- Приложение должно быть доступно по адресу {{host}}

- Приложение должно быть собрано и опубликовано на сервере без необходимости дополнительных настроек (билд на сервере).

- Все данные хранятся на сервере. В локальном хранилище браузера должен храниться только токен авторизации.

- Исходный код необходимо сохранить в папке source на сервере.

- ClientId — это ваш логин, в любых запросах вы должны указать заголовок ClientId, который равен вашему логину. Например:

GET /board/{hash}

ClientId: <ClientId>

РАБОТА ДОЛЖНА БЫТЬ ОФОРМЛЕНА В ФОРМАТЕ GIT-РЕПОЗИТОРИЯ С МИНИМУМ ДВУМЯ ОСМЫСЛЕННЫМИ КОММИТАМИ

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

2. СПЕЦИАЛЬНЫЕ ПРАВИЛА КОМПЕТЕНЦИИ

2.1. Личный инструмент конкурсанта

Конкурсанту разрешается использовать собственные:

- Клавиатуру. Если конкурсант пользуется своей клавиатурой, и она выходит из строя, организатор предоставляет ему замену;
- Мышь;
- Графический планшет;
- Наушники;
- аудиофайлы с музыкальными композициями (не более 30 файлов в формате mp3). Файлы предоставляются на флеш-носителях в день С-1 техническому эксперту на проверку.

Все оборудование не должно содержать встроенной памяти.

2.2. Материалы, оборудование и инструменты, запрещенные на площадке

Всё оборудование, принесенное конкурсантами, может быть проверено экспертами на наличие внутренних запоминающих устройств. В случае обнаружения материалы будут изыматься.

Экспертам допускается использовать персональные компьютеры, но в специальной зоне. В помещениях для проведения оценки использование любых электронных устройств запрещено, кроме специально организованных для оценки.

Также запрещено приносить:

- дополнительные программы и библиотеки, не предусмотренные
- инфраструктурным листом;
- мобильные телефоны;
- фото/видео устройства;
- карты памяти и другие носители информации;
- внутренние устройства памяти в собственном оборудовании.

3. ПРИЛОЖЕНИЯ

Приложение 1. Инструкция по заполнению матрицы конкурсного задания

Приложение 2. Матрица конкурсного задания

Приложение 3. Инструкция по охране труда

Приложение 4. Чек-лист компетенции

Приложение № 5 Медиафайлы