КОНКУРСНОЕ ЗАДАНИЕ КОМПЕТЕНЦИИ

«ВЕБ-ТЕХНОЛОГИИ»

2023 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](#_heading=h.26in1rg)

[1.1. ОБЩИЕ СВЕДЕНИЯ О ТРЕБОВАНИЯХ КОМПЕТЕНЦИИ 3](#_heading=h.lnxbz9)

[1.2. ПЕРЕЧЕНЬ ПРОФЕССИОНАЛЬНЫХ ЗАДАЧ СПЕЦИАЛИСТА ПО КОМПЕТЕНЦИИ «ВЕБ-ТЕХНОЛОГИИ» 3](#_heading=h.35nkun2)

[1.3. ТРЕБОВАНИЯ К СХЕМЕ ОЦЕНКИ 8](#_heading=h.1ksv4uv)

[1.4. СПЕЦИФИКАЦИЯ ОЦЕНКИ КОМПЕТЕНЦИИ 10](#_heading=h.44sinio)

[1.5. КОНКУРСНОЕ ЗАДАНИЕ 10](#_heading=h.2jxsxqh)

[1.5.1. Разработка/выбор конкурсного задания 10](#_heading=h.z337ya)

[1.5.2. Структура модулей конкурсного задания (инвариант/вариатив) 11](#_heading=h.3j2qqm3)

[Модуль А. Разработка интерфейса пользователя 11](#_heading=h.1y810tw)

[Модуль Б. Разработка Веб-приложения на стороне клиента 17](#_heading=h.qsh70q)

[Модуль В. Разработка Веб-приложения на стороне сервера 36](#_heading=h.4i7ojhp)

[2. СПЕЦИАЛЬНЫЕ ПРАВИЛА КОМПЕТЕНЦИИ 37](#_heading=h.1ci93xb)

[2.1. Личный инструмент конкурсанта 37](#_heading=h.3whwml4)

[2.2. Материалы, оборудование и инструменты, запрещенные на площадке 37](#_heading=h.17dp8vu)

[3. ПРИЛОЖЕНИЯ 38](#_heading=h.2bn6wsx)

**ИСПОЛЬЗУЕМЫЕ СОКРАЩЕНИЯ**

*1. ИР – Информационный ресурс*

*2. SSH - SSH (от англ. secure shell ― безопасная оболочка) ― это защищённый сетевой протокол для удалённого управления сервером через интернет.*

*3. FTP – (File Transfer Protocol), или «протокол передачи файлов» это набор процедур или правил, позволяющих электронным устройствам взаимодействовать между собой.*

*4. CMS – (Content Management System) ― это система создания и управления сайтом. Это визуально удобный интерфейс, с помощью которого можно добавлять и редактировать содержимое сайта.*

## 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»

*Таблица №1*

**Перечень профессиональных задач специалиста**

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Раздел** | **Важность в %** |
| 1 | **Тестирование информационных ресурсов** | 7 |
| Специалист должен знать и понимать:  − способы решения возникающих проблем, анализ проблемной ситуации возникшей в ходе решения профессиональных задач, пути их решения с учетом этических норм и правил, опираясь на профессиональную этику;  − принципы, лежащие в основе сбора и представления информации;  − основные приемы и методы визуального представления информации (черновое макетирование страниц, объектно-событийное моделирование, создание блок-схем и др.);  − английский язык в рамках чтения и понимания официальной технической документации по используемым технологиями и языкам программирования. |
| Специалист должен уметь:  − собирать, анализировать и оценивать информацию;  − использовать навыки грамотности для толкования стандартов и требований;  − составлять тестовую документации для тестирования новых функциональностей продукта  − проводить ручное тестирование новых функциональностей  − проводить регрессионное ручное тестирование  − вести баг-репорты  − составлять отчеты по итогам тестирования  − автоматизировать регрессионное тестирование  − общаться с заказчиком, командой разработки и тестирования |
| 2 | **Техническая поддержка и администрирование информационных ресурсов** | 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** |  | **A** | **Б** | **В** | **Г** |  |
| **1** | 2 | 1 | 2.5 | 0 | 5,5 |
| **2** | 2 | 2 | 2 | 0 | 6 |
| **3** | 20 | 1 |  | 0 | 21 |
| **4** |  | 23 |  | 0 | 23 |
| **5** |  |  | 22 | 0 | 22 |
| **6** |  |  |  | 0 | 0 |
| **Итого баллов за критерий/модуль** | | 24 | 27 | 26,5 | 0 | **77,5** |

### 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 клиента** | Проверка результата выполнения модуля после его завершения |
| **В** | **Разработка Веб-приложения на стороне сервера** | Проверка результата выполнения модуля после его завершения |
| **Г** | **Разработка ИР с использованием готовых решений** | Не используется |

### 1.5. КОНКУРСНОЕ ЗАДАНИЕ

Общая продолжительность Конкурсного задания[[1]](#footnote-1): 12 ч.

Количество конкурсных дней: 3 дня

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

Оценка знаний участника 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 4 модулей, включает обязательную к выполнению часть (инвариант) - 3 модуля, и вариативную часть - 1 модуль. Общее количество баллов конкурсного задания составляет 100.

Обязательная к выполнению часть (инвариант) выполняется всеми регионами без исключения на всех уровнях чемпионатов.

Количество модулей из вариативной части, выбирается регионом самостоятельно в зависимости от материальных возможностей площадки соревнований и потребностей работодателей региона в соответствующих специалистах. В случае если ни один из модулей вариативной части не подходит под запрос работодателя конкретного региона, то вариативный (е) модуль (и) формируется регионом самостоятельно под запрос работодателя. При этом, время на выполнение модуля (ей) и количество баллов в критериях оценки по аспектам не меняются.

*Таблица №4*

**Матрица конкурсного задания**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Обобщенная трудовая функция | Трудовая функция | Нормативный документ/ЗУН | Модуль | Константа/вариатив | ИЛ | КО |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Инструкция по заполнению матрицы конкурсного задания **(Приложение № 1)**

#### 1.5.2. Структура модулей конкурсного задания (инвариант/вариатив)

##### Модуль А. Разработка интерфейса пользователя

**Технологии этого модуля:** граф. дизайн, HTML5, CSS3

**Время на выполнение:** 6 часов

**Задания:** В современном мире с каждым днём появляется всё больше и больше новых технологий. В настоящее время сложно представить организацию, у которой нет веб-сайта.

К вам обратилась компания «Индекс Драйв» - новая и энергичная компания по аренде автомобилей. Главная цель компании – сделать любой автомобиль более доступным и практичным.

Основные принципы работы сервиса:

1. Выбрать любой удобный автомобиль

2. Возможность выбора места, времени, даты

3. Выбор тарифа

4. Удобная система оплаты

Призыв: На любой случай, любой автомобиль.

Вам необходимо использовать все имеющиеся навыки в дизайне и верстке чтобы сверстать Landing Page, а также все остальные страницы. Используйте анимацию для привлечения внимания посетителя к акцентам и основным объектам сервиса.

Заказчик хочет, чтобы сайт был современный и энергичный, а также удобный, простой и не менял свои качества при различных разрешениях экрана.

Заказчик отметил, что услугами аренды автомобиля, в основном, пользуются люди, от 23 лет и имеющие стаж вождения минимум 5 лет.

Компания не хочет разбираться со сторонними авторскими правами на материалы, поэтому вы можете использовать только то, что предоставляет заказчик в медиафайлах или ваши личные дизайнерские разработки.

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

Ваша задача – сверстать следующие страницы веб-сайта:

● Главная страница - Landing Page

● Страница входа в личный кабинет

● Страница регистрации в личном кабинете

● Страница личного кабинета

● Страница бронирования

● Страница история бронирования

● Страница филиалов

● Страница банковские карты

Главная страница (landing page)

Главная страница должна содержать следующие блоки:

● Шапка сайта

○ Логотип компании

○ Меню навигации

■ Акции

■ Бронирование

■ Личный кабинет

● Секция о доверии компании.

● Форма поиска. Должная содержать следующие поля:

○ Класс автомобиля (эконом, премиум)

○ Марка

○ Модель

○ Кнопка поиска автомобилей

● Новости (список новостей доступен в медиафайлах). Каждая новость должна содержать следующую информацию:

○ Изображение

○ Название новости

○ Краткое описание новости

○ Кнопка для просмотра новости

● Подвал сайта

○ Телефон “8 (800) 111-11-11”

○ Навигация по сайту

Страница бронирования

На этой странице необходимо отобразить форму для сбора данных клиентов и его авто, а именно:

● Данные о авто

○ Идентификатор машины

○ Марка авто

○ Название авто

○ Номер машины

○ Дата бронирования

○ Дата возврата

○ Стоимость бронирования за день

○ Филиал

● Данные о клиенте

○ Имя

○ Фамилия

○ Отчество

○ Номер телефона

○ Паспортные данные

○ Банковская карта (отображаются последние 4 цифры номера банковской карты)

○ Дата рождения

● Кнопка для бронирования машины и возможность для бронирования нескольких машин.

● Финальная стоимость

Формируется исходя из данных. Если бронируется машина срок больше 3-х дней, то клиенту формируется скидка в 5% и так за каждые 3 дня скидка прибавляется по 5% и может быть максимум 25%. Это необходимо визуализировать.

● Возможность выбора машин

● Возможность выбора банковской карты

● Кнопка для оформления бронирования

Страница регистрации в личном кабинете

На этой странице вам необходимо сделать форму со следующими полями:

● Имя

● Фамилия

● Отчество

● Паспортные данные

● Телефон

● Дата рождения

● Пароль

● Повтор пароля

● Кнопка для регистрации

Страница входа в личный кабинет

На этой странице вам необходимо сделать форму со следующими полями:

● Телефон

● Пароль

● Кнопка для входа

Страница истории бронирований

На этой странице необходимо отобразить всю информацию о истории бронирований, а именно:

● Информация о бронировании

○ Код бронирования

○ Стоимость бронирования

● Информация об аренде (для каждого заказа)

○ Номер заказа

○ Марка автомобиля

○ Модель автомобиля

○ Фото автомобиля

○ Дата начала аренды

○ Дата окончания аренды

○ Итоговое количество дней аренды

○ Наименование филиала, где производилась аренда

○ Итоговая стоимость аренды

● Кнопка для повторного бронирования

В случае если у пользователя нет бронирований, то будет сообщение: «Здесь пока ничего нет»

Страница личного кабинета

На этой странице необходимо отобразить информацию о пользователе, а именно:

● Имя

● Фамилия

● Отчество

● Баллы

● Количество бронирований

● Кнопка перехода на страницу банковские карты

● Кнопка выхода из личного кабинета

● Бронирования. Каждое бронирование должно отражать следующую информацию:

○ Идентификатор авто

○ Марка

○ Название

○ Картинка машины

○ Описание машины

○ Дата бронирования

○ Дата возврата

○ Выбранный филиал

Страница филиалов

На этой странице необходимо отобразить карту со всеми местонахождениями филиалов. Всего в городе 5 филиалов. При нажатии на филиал отображается список из 3-х машин, доступных к бронированию.

Страница банковские карты

На этой странице необходимо отобразить информацию о привязанных банковских картах клиента, а именно:

● Номер карты (номер карты должен быть не явным, отображаются последние 4 цифры)

● Тип карты (Visa, MasterCard, Мир): тип банковской карты определяется по первым цифрам номера карты (1111 - Visa, 2222 - MasterCard, 3333 – Мир)

● Активность (Активна/Закончился срок обслуживания карты)

● Кнопка удалить карту

● Кнопка добавить банковскую карту (модальное окно)

Модальное окно добавить банковскую карту

Данное окно содержит информацию о добавляемой банковской карте, а именно:

● Номер карты (длина номера карты 16 символов, разделенных пробелом по 4 символа)

● Имя и фамилия держателя банковской карты латиницей

● Дата окончания банковской карты

● Код CVV/CVC (длина кода 3 символа)

● Кнопка добавить (после нажатия кнопки банковская карта появляется на странице банковская карта)

Необходимо создать логотип компании в формате PNG.

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

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

Сохраните сверстанные страницы со следующими именами:

● Главная страница - Landing Page – index.html

● Страница входа в личный кабинет – login.html

● Страница регистрации в личном кабинете – register.html

● Страница личного кабинета – profile.html

● Страница бронирования – booking.html

● Страница история бронирования – history.html

● Страница филиалов – branches.html

● Страница банковские карты – cards.html

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

Проверяются только работы, загруженные на сервер! Страницы расположенные в других местах или с ошибками в названии проверяться не будут!

Ваш HTML/CSS должен быть валидным.

Оценка будет производиться в браузере Google Chrome.

Использование любых фреймворков и библиотек (bootstrap, например) запрещено.

##### Модуль Б. Разработка Веб-приложения на стороне сервера

**Технологии этого модуля:** REST API

**Время на выполнение**: 3 часа

**Задания:** К вам обратилась молодая компания «Tortotoro». В ее ближайших планах открыть кафе-кондитерскую в вашем городе и для управления им требуется информационная система. Пользователями системы являются сотрудники кафе. Основная задача системы состоит в учёте и управлении заказами, а также формировании отчётной документации.

Так как заказчик ограничен в деньгах, была достигнута договоренность о разработке MVP (minimum viable product). На данном этапе ваша задача состоит в разработке минимального функционала REST API для этой информационной системы.

Заказчик планирует дальнейшее развитие проекта, поэтому использование фреймворков будет плюсом.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

В медиафайлах вам предоставляется sql дамп с готовой базой данных. База данных уже содержит набор данных, которые НЕ должны быть изменены! Структуру БД менять также запрещается. Любое изменение или удаление предоставленных данных в БД будет влиять на вашу оценку.

ВНИМАНИЕ! Для упрощения последующей автоматической проверки не хешируйте пароли пользователей в БД!

Описание проекта и задач

Общее описание проекта

Все пользователи системы подразделяются на три группы:

1. Администраторы;

2. Официанты;

3. Повара.

Пользователи получают доступ к функциям ИС только после успешной авторизации.

Функционал администратора:

● Добавление новых пользователей в системе (с фото);

● Создание смены;

● Открытие и закрытие смены;

● Назначение работников на смены;

● Просмотр всех заказов определенной смены.

Функционал повара:

● Просмотр заказов, принятых от клиентов;

● Изменение статуса заказа на «готовится» и «готов».

Функционал официанта:

● Создание нового заказа;

● Добавление и удаление позиций из заказа;

● Просмотр всех принятых им заказов за смену;

● Изменение статуса заказа на «отменен» и «оплачен».

Ваша задача – реализовать REST API заданной структуры.

В примерах будет использоваться переменная {{host}} которая обозначает адрес http://xxxxxx-m2.wsr.ru/, где xxxxxx - логин участника.

Общие требования к API

API должно быть доступно с других доменов. Позаботьтесь о настройках правил CORS.

Все функции, кроме аутентификации доступны только авторизованным пользователям. Идентификацию пользователя организуйте посредством Bearer Token.

При попытке доступа к защищенным авторизацией функциям системы во всех запросах необходимо возвращать ответ следующего вида:

Status: 403

Content-Type: application/json

Body:

{

“error”: {

“code”: 403,

“message”: “Login failed”

}

}

При попытке доступа авторизованным пользователем к функциям недоступным для своей группы во всех запросах необходимо возвращать ответ следующего вида:

Status: 403

Content-Type: application/json

Body:

{

“error”: {

“code”: 403,

“message”: “Forbidden for you”,

}

}

В случае ошибок связанных с валидацией данных во всех запросах необходимо возвращать следующее тело ответа:

{

“error”: {

“code”: <code>,

“message”: <message>,

“errors”: {

<key>: [ <error message>]

}

}

}

Обратите внимание, что вместо <code> и <message> необходимо указывать соответствующее значение, определенное в описании ответа на соответствующий запрос. В свойстве error.errors необходимо перечислить те свойства, которые не прошли валидацию, а в их значениях указать массив с ошибками валидации.

Например, если отправить пустой запрос на сервер, где проверяется следующая валидация:

● phone – обязательно поле

● password – обязательное поле

то тело ответа будет следующим:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

phone: [ “field phone can not be blank” ],

password: [ “field password can not be blank” ]

}

}

}

Примите во внимание, что code и message могут быть определены иначе, если в запросе указано иное. В значениях свойств errors вы можете использовать любые сообщения об ошибках (если не указана конкретная ошибка), но они должны описывать возникшую проблему.

Специфические требования к API

Аутентификация

Запрос для аутентификации пользователя в системе. При отправке запроса необходимо передать объект с логином и паролем. Если клиент отправил корректные данные, то необходимо вернуть сгенерированный токен, а иначе сообщение об ошибке.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/login;  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  “login”: “admin“,  “password“: “admin“  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  “data”: {  “user\_token”: <сгенерированный token>  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Неправильные логин или пароль  **Status:** 401  **Content-Type:** application/json  **Body:**  {  “error”: {  “code”: 401,  “message”: “Authentication failed”  }  } |

Выход

Запрос предназначен для очистки значение токена пользователя.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/logout  **Method:** GET | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "message": "logout"  }  } |

Функционал администратора

Просмотр списка всех сотрудников

Запрос предназначен для получения списка всех сотрудников кафе.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/user  **Method:** GET | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "name": "Bell",  "login": "admin",  "status": "working",  "group": "Администратор"  },  {  "id": 2,  "name": "Shana",  "login": "waiter",  "status": "working",  "group": "Официант"  },  {  "id": 3,  "name": "Estrella",  "login": "cook",  "status": "working",  "group": "Повар"  }  ]  } |

Добавление сотрудника

Запрос для регистрации нового пользователя в системе. При отправке запроса необходимо передать FormData со следующими полями:

● name – обязательное поле, строка;

● surname – не обязательное поле, строка;

● patronymic – не обязательное поле, строка;

● login – обязательное и уникальное поле, строка;

● password – обязательное поле, строка;

● photo\_file – файл с фото нового сотрудника (загружается в папку photos), не обязательное поле, файл jpeg, png;

● role\_id – обязательное поле, число, роль с данным идентификатором должна быть.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/user  **Method:** POST    **Headers**  - Content-Type: multipart/form-data    **FormData:**  “name”: “Ivan”,  “login”: “Ivanov”,  “password”: “paSSword”  “role\_id”: 1 | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 6,  "status": "created"  }  }  Ошибки валидации полей  **Формат ответа из общих требований** |

Создание смены

Запрос предназначен для создания новой смены. При отправке запроса необходимо отправить объект со следующими полями:

● start – обязательное поле, дата и время начала смены, не должно быть раньше чем текущая дата и время, формат Y-m-d H:i;

● end – обязательное поле, дата и время начала смены, не должно быть раньше чем дата и время начала смены, формат Y-m-d H:i.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  "start": "2021-03-26 08:00",  "end": "2021-03-26 17:00"  } | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "id": 16,  "start": "2021-03-26 08:00",  "end": "2021-03-26 17:00"  }  Ошибки валидации полей  **Формат ответа из общих требований** |

Открытие смены

Запрос открывает смену. Открыть смену можно только если все смены закрыты.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift/{{id}}/open  **Method:** GET    **Headers**  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 18,  "start": "2021-03-28 08:00:00",  "end": "2021-03-29 17:00:00",  "active": true  }  }  Попытка открыть вторую смену  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. There are open shifts!"  }  } |

Закрытие смены

Запрос закрывает смену. Закрыть уже закрытую смену нельзя.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift/{{id}}/close  **Method:** GET    **Headers**  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 18,  "start": "2021-03-28 08:00:00",  "end": "2021-03-29 17:00:00",  "active": false  }  }  Попытка закрыть закрытую смену  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. The shift is already closed!"  }  } |

Добавление сотрудников на смену

Запрос позволяет добавить работников на смену. Два раза на одну и ту же смену пользователя добавить нельзя.

При отправке запроса необходимо отправить объект со следующими полями:

● user\_id - обязательное, пользователь с таким идентификатором должен быть среди работников и не быть уволенным.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift/{{id}}/user  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  "user\_id":4  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id\_user": 1,  "status": "added"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка два раза добавить работника  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. The worker is already on shift!"  }  } |

Просмотр заказов за конкретную смену

Запрос предназначен для просмотра заказов конкретной смены.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/work-shift/{{id}}/order  Method: GET    Headers  - Content-Type: application/json | Успех  Status: 200  Content-Type: application/json  Body:  {  "data": {  "id": 1,  "start": "2021-03-30 08:00:00",  "end": "2021-03-31 17:00:00",  "active": 1,  "orders": [  {  "id": 1,  "table": "Столик №1",  "shift\_workers": "Jaiden",  "create\_at": "2021-03-03T08:34:43.000000Z",  "status": "Готовится",  "price": 2934  },  {  "id": 2,  "table": "Столик №2",  "shift\_workers": "Trudie",  "create\_at": "2021-03-11T08:34:57.000000Z",  "status": "Готовится",  "price": 0  }  ],  "amount\_for\_all": 2934  }  } |

Функционал официанта

Создание заказа для определенного столика

Заказ предназначен для создания заказа официантом. Официант может создать заказ только на активную смену и только если он в ней работает. При отправке запроса необходимо отправить объект со следующими полями:

● work\_shift\_id - идентификатор смены, обязательное, смена должна существовать и быть активной, пользователь должен работать на смене;

● return parent::toArray($request);return parent::toArray($request); - идентификатор столика, должен существовать;

● count - количество персон, необязательное, целое.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/order  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  "work\_shift\_id":1,  "table\_id":2,  "number\_of\_person":2  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 10,  "table": "Столик №2",  "shift\_workers": "Trudie",  "create\_at": "2021-03-27T04:03:37.000000Z",  "status": "Принят",  "price": 0  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка добавить заказ в неактивную смену  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. The shift must be active!"  }  }  Попытка добавить заказ официантом не этой смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": Forbidden. You don't work this shift!"  }  } |

Просмотр конкретного принятого заказа

Запрос предназначен для просмотра деталей конкретного заказа принятого официантом. Если официант не принимал данный заказ, запрос не должен быть разрешен.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/order/{{id}}  Method: GET    Headers  - Content-Type: application/json | Успех  Status: 200  Content-Type: application/json  Body:  {  "data": {  "id": 1,  "table": "Столик №1",  "shift\_workers": "Allan",  "create\_at": null,  "status": "Принят",  "positions": [  {  "id": 1,  "count": 5,  "position": "Aut sit ut et reprehenderit sed cumque.",  "price": 8135  },  {  "id": 2,  "count": 1,  "position": "Ut similique dolorum eos culpa officiis.",  "price": 1993  }  ],  "price\_all": 10128  }  }  Попытка доступа к не своему заказу  Status: 403  Content-Type: application/json  Body:  {  "error": {  "code": 403,  "message": "Forbidden. You did not accept this order!"  }  } |

Просмотр всех принятых заказов за смену

Запрос предназначен для получения всех заказов, выданных всеми официантами за смену. Если официант не работал на смене, запрос не должен быть разрешен.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/work-shift/{{id}}/orders  Method: GET    Headers  - Content-Type: application/json | Успех  Формат как у соответствующей функции у администратора  Попытка получить заказы не своей смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. You did not accept this order!"  }  } |

Изменение статуса заказа

Запрос предназначен для смены статуса заказа активной смены, который он принимал. Для официанта доступна смена со статуса “принят” на “отменен”, а также со статуса “готов” на статус “оплачен”.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/order/{{id}}/change-status  **Method:** PATCH    **Headers**  - Content-Type: application/json    **Body:**  {  "status":"paid-up"  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "status": "paid-up"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка изменить на неразрешенный статус  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden! Can't change existing order status"  }  }  Попытка изменить не свой заказ  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden! You did not accept this order!"  }  }  Попытка изменить заказ закрытой смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "You cannot change the order status of a closed shift!"  }  } |

Функционал повара

Просмотр заказов текущей смены

Запрос предназначен для получения списка заказов активной смены со статусом “принят” или “готовится”.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/order/taken  Method: GET    Headers  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 7,  "table": "Столик №7",  "shift\_workers": "Palma",  "create\_at": "2021-03-28T02:52:52.000000Z",  "status": "Принят",  "price": 0  },  {  "id": 11,  "table": "Столик №7",  "shift\_workers": "Palma",  "create\_at": "2021-03-28T03:23:52.000000Z",  "status": "Готовится",  "price": 0  }  ]  } |

Изменение статуса заказа

Запрос предназначен для смены статуса заказа активной смены. Для повара доступна смена со статуса “принят” на “готовится”, а также со статуса “готовится” на статус “готов”.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/order/{{id}}/change-status  **Method:** PATCH    **Headers**  - Content-Type: application/json    **Body:**  {  "status":"preparing"  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "status": "preparing"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка изменить на неразрешенный статус  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden! Can't change existing order status"  }  }  Попытка изменить заказ закрытой смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "You cannot change the order status of a closed shift!"  }  } |

Инструкция для конкурсанта

Разработанное API должно быть доступно по адресу http://xxxxxx-m2.wsr.ru/api-cafe, где xxxxxx - логин участника (указан на индивидуальной карточке).

Форматы запросов и ответов, а также форматы дат и времени должен соответствовать примерам из задания.

##### Модуль В. Клиентское программирование

**Технологии этого модуля:** Клиентское программирование

**Время на выполнение**: 3 часа

**Задания:** К вам обратилась молодая компания «Tortotoro». В ее ближайших планах открыть кафе-кондитерскую в вашем городе и для управления им требуется информационная система. Пользователями системы являются сотрудники кафе. Основная задача системы состоит в учёте и управлении заказами, а также формировании отчётной документации.

Заказчик предоставляет вам готовую верстку со всеми страницами и рабочее API. Вам необходимо использовать все имеющиеся навыки в клиентской разработке для создания Single Page Application, далее SPA.

Заказчик планирует дальнейшее развитие проекта, поэтому использование фреймворков будет плюсом.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

Описание проекта и задач

Общее описание проекта

Все пользователи системы подразделяются на три группы:

1. Администраторы;

2. Официанты;

3. Повара.

Пользователи получают доступ к функциям ИС только после успешной авторизации.

Приложение должно обладать следующим функционалом:

Функционал администратора:

● Регистрация новых пользователей в системе;

● Просмотр всех пользователей

● Просмотр одного пользователя

● Увольнение сотрудника

● Создание смены

● Открытие смены

● Закрытие смены

● Просмотр всех смен

● Назначение работников на смены;

● Удаление сотрудника со смены

● Просмотр всех заказов определенной смены.

Функционал повара:

● Просмотр заказов, принятых от клиентов;

● Изменение статуса заказа на «готовится» и «готов».

Функционал официанта:

● Создание нового заказа;

● Отмена заказа

● Добавление и удаление позиций из заказа;

● Просмотр конкретного заказа

● Просмотр всех принятых им заказов за смену;

● Изменение статуса заказа на «оплачен».

Ваша задача – реализовать приложение заданной структуры, которое будет работать с уже разработанным API.

Для вашего удобства, во всех URL будет использоваться переменная {host} которая обозначает хост адрес API: http://server-m3.wsr.ru

Функционал администратора:

* просмотр списка всех сотрудников. На данном экране администратору должен предоставляться список активных сотрудников

Для получения списка всех сотрудников кафе следует использовать данный запрос:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/user  **Method:** GET | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 1,  "name": "Bell",  "login": "admin",  "status": "working",  "group": "Администратор"  },  {  "id": 2,  "name": "Shana",  "login": "waiter",  "status": "working",  "group": "Официант"  },  {  "id": 3,  "name": "Estrella",  "login": "cook",  "status": "working",  "group": "Повар"  }  ]  } |

Добавление сотрудника

Запрос для регистрации нового пользователя в системе. Неавторизованный пользователь должен иметь возможность зарегистрироваться в системе на странице с регистрацией.

Для регистрации пользователя предусмотрен следующий запрос:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/user  **Method:** POST    **Headers**  - Content-Type: multipart/form-data    **FormData:**  “name”: “Ivan”,  “login”: “Ivanov”,  “password”: “paSSword”  “role\_id”: 1 | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 6,  "status": "created"  }  }  Ошибки валидации полей  **Формат ответа из общих требований** |

Работа со сменами

На странице с работой по смене пользователь должен иметь возможность установить дату и время начала смены, дату и время конца смены, а также назначить сотрудников на данную смену.

На странице должна быть возможность просмотра закрытых заказов в выбранную смену.

Также должна быть возможность открытия и закрытия этой смены. Смена открывается при условии, что все другие смены закрыты. Пользователь должен закрыть текущую смену (если она есть) для создания новой.

Получить информацию по новой смене можно с помощью запроса:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  "start": "2021-03-26 08:00",  "end": "2021-03-26 17:00"  } | Успех  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "id": 16,  "start": "2021-03-26 08:00",  "end": "2021-03-26 17:00"  }  Ошибки валидации полей  **Формат ответа из общих требований** |

Запрос открывает смену:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift/{{id}}/open  **Method:** GET    **Headers**  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 18,  "start": "2021-03-28 08:00:00",  "end": "2021-03-29 17:00:00",  "active": true  }  }  Попытка открыть вторую смену  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. There are open shifts!"  }  } |

Запрос закрывает смену:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift/{{id}}/close  **Method:** GET    **Headers**  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 18,  "start": "2021-03-28 08:00:00",  "end": "2021-03-29 17:00:00",  "active": false  }  }  Попытка закрыть закрытую смену  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. The shift is already closed!"  }  } |

При отправке запроса на добавление сотрудника на смену необходимо отправить запрос:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/work-shift/{{id}}/user  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  "user\_id":4  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id\_user": 1,  "status": "added"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка два раза добавить работника  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. The worker is already on shift!"  }  } |

Просмотр заказов за конкретную смену

Запрос предназначен для просмотра заказов конкретной смены.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/work-shift/{{id}}/order  Method: GET    Headers  - Content-Type: application/json | Успех  Status: 200  Content-Type: application/json  Body:  {  "data": {  "id": 1,  "start": "2021-03-30 08:00:00",  "end": "2021-03-31 17:00:00",  "active": 1,  "orders": [  {  "id": 1,  "table": "Столик №1",  "shift\_workers": "Jaiden",  "create\_at": "2021-03-03T08:34:43.000000Z",  "status": "Готовится",  "price": 2934  },  {  "id": 2,  "table": "Столик №2",  "shift\_workers": "Trudie",  "create\_at": "2021-03-11T08:34:57.000000Z",  "status": "Готовится",  "price": 0  }  ],  "amount\_for\_all": 2934  }  } |

Страницы для официанта:

Работа с заказами:

Создание заказа на столик

Данная страница должна иметь поля для выбора действующей смены, карту столов / номер выбранного стола, возможность указать количество персон за столиком.  
  
В заказе также должна быть указана более подробная информация: статус заказа, позиция меню, количество данной позиции, а также стоимость.

Запрос для создания заказа на определенный столик:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/order  **Method:** POST    **Headers**  - Content-Type: application/json    **Body:**  {  "work\_shift\_id":1,  "table\_id":2,  "number\_of\_person":2  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 10,  "table": "Столик №2",  "shift\_workers": "Trudie",  "create\_at": "2021-03-27T04:03:37.000000Z",  "status": "Принят",  "price": 0  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка добавить заказ в неактивную смену  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. The shift must be active!"  }  }  Попытка добавить заказ официантом не этой смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": Forbidden. You don't work this shift!"  }  } |

Просмотр конкретного принятого заказа

Запрос предназначен для просмотра деталей конкретного заказа принятого официантом. Если официант не принимал данный заказ, запрос не должен быть разрешен.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/order/{{id}}  Method: GET    Headers  - Content-Type: application/json | Успех  Status: 200  Content-Type: application/json  Body:  {  "data": {  "id": 1,  "table": "Столик №1",  "shift\_workers": "Allan",  "create\_at": null,  "status": "Принят",  "positions": [  {  "id": 1,  "count": 5,  "position": "Aut sit ut et reprehenderit sed cumque.",  "price": 8135  },  {  "id": 2,  "count": 1,  "position": "Ut similique dolorum eos culpa officiis.",  "price": 1993  }  ],  "price\_all": 10128  }  }  Попытка доступа к не своему заказу  Status: 403  Content-Type: application/json  Body:  {  "error": {  "code": 403,  "message": "Forbidden. You did not accept this order!"  }  } |

Изменение статуса заказа

Запрос предназначен для смены статуса заказа активной смены, который он принимал. Для официанта доступна смена со статуса “принят” на “отменен”, а также со статуса “готов” на статус “оплачен”.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/order/{{id}}/change-status  **Method:** PATCH    **Headers**  - Content-Type: application/json    **Body:**  {  "status":"paid-up"  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "status": "paid-up"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка изменить на неразрешенный статус  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden! Can't change existing order status"  }  }  Попытка изменить не свой заказ  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden! You did not accept this order!"  }  }  Попытка изменить заказ закрытой смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "You cannot change the order status of a closed shift!"  }  } |

Просмотр всех принятых заказов за смену

На данной странице пользователь может посмотреть все заказы, выданные всеми официантами за текущую смену

Если официант не работал на смене, страница не может быть доступна.

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/work-shift/{{id}}/orders  Method: GET    Headers  - Content-Type: application/json | Успех  Формат как у соответствующей функции у администратора  Попытка получить заказы не своей смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden. You did not accept this order!"  }  } |

Интерфейс поваров

Просмотр заказов текущей смены. На данной странице пользователь может видеть все заказы активной смены. При необходимости можно «провалиться» в выбранный заказ и увидеть более подробную информацию по нему, а также изменить статус выбранного заказа.

Запрос предназначен для получения списка заказов активной смены со статусом “принят” или “готовится”:

|  |  |
| --- | --- |
| Request | Response |
| URL: {{host}}/api-tort/order/taken  Method: GET    Headers  - Content-Type: application/json | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": [  {  "id": 7,  "table": "Столик №7",  "shift\_workers": "Palma",  "create\_at": "2021-03-28T02:52:52.000000Z",  "status": "Принят",  "price": 0  },  {  "id": 11,  "table": "Столик №7",  "shift\_workers": "Palma",  "create\_at": "2021-03-28T03:23:52.000000Z",  "status": "Готовится",  "price": 0  }  ]  } |

Изменение статуса заказа

Запрос предназначен для смены статуса заказа активной смены. Для повара доступна смена со статуса “принят” на “готовится”, а также со статуса “готовится” на статус “готов”.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {{host}}/api-tort/order/{{id}}/change-status  **Method:** PATCH    **Headers**  - Content-Type: application/json    **Body:**  {  "status":"preparing"  } | Успех  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 1,  "status": "preparing"  }  }  Ошибки валидации полей  **Формат ответа из общих требований**  Попытка изменить на неразрешенный статус  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "Forbidden! Can't change existing order status"  }  }  Попытка изменить заказ закрытой смены  **Status:** 403  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 403,  "message": "You cannot change the order status of a closed shift!"  }  } |

Приложение должно быть доступно по адресу http://xxxxxx-m3.wsr.ru/, где xxxxxx - логин участника.

## 

## 2. СПЕЦИАЛЬНЫЕ ПРАВИЛА КОМПЕТЕНЦИИ*[[2]](#footnote-2)*

### 2.1. Личный инструмент конкурсанта

Конкурсанту разрешается использовать собственные:

● клавиатуру на любом языке. Если конкурсант пользуется своей

клавиатурой, и она выходит из строя, организатор предоставляет ему замену.;

● языковые файлы для клавиатуры;

● мышь;

● графический планшет;

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 Критерии оценки

Приложение №4 Инструкция по охране труда и технике безопасности по компетенции «Веб-технологии».

Приложение №5 Медиа файлы

1. *Указывается суммарное время на выполнение всех модулей КЗ одним конкурсантом.* [↑](#footnote-ref-1)
2. *Указываются особенности компетенции, которые относятся ко всем возрастным категориям и чемпионатным линейкам без исключения.* [↑](#footnote-ref-2)