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](#_Toc126846333)

[1.1. ОБЩИЕ СВЕДЕНИЯ О ТРЕБОВАНИЯХ КОМПЕТЕНЦИИ 3](#_Toc126846334)

[1.2. ПЕРЕЧЕНЬ ПРОФЕССИОНАЛЬНЫХ ЗАДАЧ СПЕЦИАЛИСТА ПО КОМПЕТЕНЦИИ «ВЕБ-ТЕХНОЛОГИИ» 3](#_Toc126846335)

[1.3. ТРЕБОВАНИЯ К СХЕМЕ ОЦЕНКИ 8](#_Toc126846336)

[1.4. СПЕЦИФИКАЦИЯ ОЦЕНКИ КОМПЕТЕНЦИИ 10](#_Toc126846337)

[1.5. КОНКУРСНОЕ ЗАДАНИЕ 10](#_Toc126846338)

[1.5.1. Разработка/выбор конкурсного задания (https://disk.yandex.ru/i/XJfQr1jBmZ2rew) 10](#_Toc126846339)

[1.5.2. Структура модулей конкурсного задания (инвариант/вариатив) 11](#_Toc126846340)

[Модуль А. Разработка интерфейса пользователя 11](#_Toc126846341)

[Модуль Б. Разработка Веб-приложения на стороне клиента 13](#_Toc126846342)

[Модуль В. Разработка Веб-приложения на стороне сервера 18](#_Toc126846343)

[Модуль Г. Разработка ИР с использованием готовых решений 37](#_Toc126846344)

[2. СПЕЦИАЛЬНЫЕ ПРАВИЛА КОМПЕТЕНЦИИ 40](#_Toc126846345)

[2.1. Личный инструмент конкурсанта 40](#_Toc126846346)

[2.2. Материалы, оборудование и инструменты, запрещенные на площадке 40](#_Toc126846347)

[3. ПРИЛОЖЕНИЯ 41](#_Toc126846348)

**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 | 1,5 | 7 |
| **2** | 2 | 2 | 2 | 1 | 7 |
| **3** | 20 | 1 |  | 5 | 26 |
| **4** |  | 23 |  | 2 | 25 |
| **5** |  |  | 22 | 3 | 25 |
| **6** |  |  |  | 10 | 10 |
| **Итого баллов за критерий/модуль** | | 24 | 27 | 26,5 | 22,5 | **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 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** | Проверка результата выполнения модуля после его завершения |

### 1.5. КОНКУРСНОЕ ЗАДАНИЕ

Общая продолжительность Конкурсного задания[[1]](#footnote-1): 15 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 (https://disk.yandex.ru/i/XJfQr1jBmZ2rew)

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

**Задание:**

Рынок совместного использования автомобилей (каршеринг) — быстро растущая отрасль мировой экономики. Согласно отчету Allied Market Research, в следующие несколько лет этот рынок будет прибавлять по 12% в год, согласно отчету Zion Market Research — 13,5% в год. Растущая популярность каршеринга также означает рост популярности площадок, которые предлагают эту услугу.

Каршеринг – вид проката автомобиля, предназначенный для аренды на небольшой отрезок времени (несколько часов) и оплачиваемый только за длительность аренды и пройденное расстояние.

К вам обратилась компания «Take Your Car» - новая и энергичная компания, предоставляющая услуги аренды автомобиля. Главная цель компании - повысить активность уже зарегистрированных в их приложениях пользователей и привлечь как можно больше новых.

Основные принципы работы сервиса:

1. Выбрать подходящий автомобиль.

2. Накопить бонусные баллы.

3. Получить снижение стоимости аренды.

Призыв: Владей поездкой, а не автомобилем!

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.

Целевая аудитория каршеринговых сервисов – это жители мегаполисов молодого и среднего возраста, которые предпочитают самостоятельное вождение службам такси. При этом они стремятся сэкономить на поездках. В основном каршеринговым сервисом пользуются мужчины 25–35 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 – создать следующие страницы веб-сайта:

● Главная страница - Landing Page

● Страница входа в личный кабинет

● Страница регистрации в личном кабинете

● Страница личного кабинета

● Страница с результатами поиска

● Страница бронирования

● Страница управления бронированием

● Страница выбора автомобиля

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

Сохраните дизайны и верстку страниц со следующими именами:

● Главная страница - Landing Page – index.png, index.html

● Страница входа в личный кабинет – login.png, login.html

● Страница регистрации в личном кабинете – register.png, register.html

● Страница личного кабинета – profile.png, profile.html

● Страница с результатами поиска – search.png, search.html

● Страница бронирования – booking.png, booking.html

● Страница управления бронированием – booking\_management.png,

booking\_management.html

● Страница выбора автомобиля – seat.png, seat.html

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

Оценка будет производиться в браузере Google Chrome.

Для работы вам будут доступны следующие фреймворки: Bootstrap 5, TailwindCSS

Главная страница (landing page)

Главная страница должна содержать следующие блоки:

● Шапка сайта

○ Логотип компании

○ Меню навигации

■ Акции

■ Поиск

■ Регистрация

■ Личный кабинет

● Секция(и) описания принципа накопления бронирований для осуществления заказа автомобиля с целью снижения его стоимости. Три принципа необходимо визуализировать при помощи анимированной инфографики на основе вводимых демо данных посетителем. Например, выбор условной даты, выбор автомобиля, демонстрация изменения стоимости и вероятность заказа.

● Секция о доверии компании.

● Форма поиска. Должна содержать следующие поля ввода:

○ Откуда – город отправления

○ Дата – дата бронирования

○ Марка автомобиля

○ Кнопка Найти

● Акции (список акций доступен в медиафайлах). Каждая акция должна содержать следующую информацию:

○ Изображение

○ Название акции

○ Краткое описание

○ Кнопка Подробнее

● Форма для подписки на закрытые акции. Должна содержать следующие поля:

○ Поле для ввода Email

○ Кнопка Подписаться

● Подвал сайта

○ Телефон 8 (800) 555-35-35 >

○ Навигация по сайту (список ссылок для навигации будет предоставлен в медиафайлах)

Страница с результатами поиска

Попасть на эту страницу можно с главной (из формы поиска автомобилей). На этой странице необходимо отобразить все найденные автомобили, а именно:

● Марка автомобиля

● Название автомобиля

● Дата бронирования

● Время бронирования

● Стоимость

● Вероятность бронирования

Вероятность бронирования - это визуальное отображение зависящие от спроса автомобилей в указанную дату.

Страница бронирования

Данная страница отображается сразу после страницы с результатами поиска. На этой странице необходимо отобразить форму для сбора данных пассажиров, а именно:

● Данные о водителе

○ Имя

○ Фамилия

○ Дата рождения

○ Номер водительского удостоверения

● Финальная стоимость бронирования

● Кнопка Оформить бронирование

Страница управления бронированием

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

● Информация о бронировании

○ Код бронирования

○ Стоимость бронирования

● Информация об автомобиле

○ Марка автомобиля

○ Название автомобиля

○ Дата бронирования

○ Время бронирования

○ Вероятность бронирования

● Информация о водителе

○ Имя

○ Фамилия

○ Дата рождения

○ Номер водительского удостоверения

● Кнопка Выбора автомобиля

Страница выбора автомобиля

На этой странице необходимо отобразить схему парковки автомобилей с возможностью забронировать автомобиль.

Страница регистрации в личном кабинете

На этой странице вам необходимо сделать форму со следующими полями:

● Имя

● Фамилия

● Номер водительского удостоверения

● Телефон

● Пароль

● Повтор пароля

● Кнопка Зарегистрироваться

Страница входа в личный кабинет

На этой странице вам необходимо сделать форму со следующими полями:

● Телефон

● Пароль

● Кнопка Войти

Страница личного кабинета

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

● Имя

● Фамилия

● Количество бронирований

● Количество бонусных баллов

● Кнопка Выход

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.

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

**Технологии этого модуля:** Клиентское программирование

**Время на выполнение**: 3 часа

**Задание:**

К вам обратилась компания «ICANFLY» - новая авиакомпания, предоставляющая услуги пассажирских авиаперевозок. Заказчик предоставляет вам полностью 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 хочет, чтобы api можно было легко поддерживать, поэтому использование JavaScript фреймворков будет плюсом.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

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

Для вашего удобства, во всех URL будет использоваться переменная {host} которая обозначает хост адрес API: http://ndntyjmx-m2.wsr.ru

Ваше SPA должно состоять из следующих экранов:

● Главная

● Вход в личный кабинет

● Регистрация в личном кабинете

● Личный кабинет

● Результаты поиска

● Бронирование

● Управление бронированием

● Выбор места в салоне воздушного судна

Приложение должно обладать следующим функционалом:

1. Домашний экран – на данном экране располагается статическая информация и форма для поиска подходящих рейсов со следующими полями:

a. Откуда (From where) – город или аэропорт вылета

b. Куда (To where) – город или аэропорт прилета

c. Туда (Departing) – дата вылета

d. Обратно (Returning) – дата возвращения обратно

e. Количество пассажиров (Passengers) – от 1 до 8

При отправке формы пользователю должен отобразиться экран с найденными рейсами.

Получить список аэропортов по введенном запросу можно с помощью следующего запроса:

Request Response

URL: {host}/api/airport

Method: GET

Query string (GET parameters):

- query ------------------------- Successful --------------------------

Status: 200

Content-Type: application/json

Body:

{

“data”: {

“items”: [

{

“name”: “Sheremetyevo”,

“iata”: “SVO”

}

]

}

}

---------------------------- No results -----------------------

Status: 200

Content-Type: application/json

Body:

{

“data”: {

“items”: []

}

}

2. Экран с найденными рейсами – на данном экране пользователю должен предоставляться список найденных рейсов в зависимости от указанных данных, а также город отправления и прибытия. Каждый рейс должен содержать:

a. Номер рейса (Flight)

b. Самолет (Aircraft)

c. Дата и время отправления (Date and time of departure)

d. Время прибытия (Arrival time)

e. Время в полете (Flight time)

f. Общую цену, учитывая количество пассажиров (Cost)

Если пользователь не указал дату возвращения обратно (Returning), то список рейсов должен включать в себя только перелеты из точки А в точку В. Если пользователь указал дату возвращения обратно (Returning), то список рейсов должен включать в себя и перелеты из точки А в В и из В в А на указанные даты.

Пользователь должен выбрать 1 рейс туда и 1 обратно (если была указана обратная дата) , которые ему подходят и нажать на кнопку “Go to booking” для перехода на экран с оформлением бронирования.

На данном экране должна быть предусмотрена фильтрация рейсов следующими способами:

● Фильтрация по времени вылета – должны отображаться рейсы время вылета которых находиться внутри выбранного диапазона.

Также должна работать сортировка по следующим параметрам:

● Самый быстрый — рейсы должны отсортироваться по времени в пути (наиболее быстрые в начале)

● Самый дешевый – рейсы должны отсортироваться по стоимости (наиболее дешевые в начале)

Пользователь не должен иметь возможность выбрать рейс, на который не осталось мест.

Получить список рейсов можно используя следующий запрос:

Request Response

URL: {host}/api/flight

Method: GET

Query string (GET parameters):

- from (SVO)

- to (KZN)

- date1 (2020-10-01)

- date2 (2020-10-13)

- passengers (1)

------------------------- Successful --------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"flights\_to": [

{

"flight\_id": 2,

"flight\_code": "FP1200",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "12:00"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "13:35"

},

"cost": 9500,

"availability": 156

},

{

"flight\_id": 14,

"flight\_code": "FP 1201",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "08:35"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "10:05"

},

"cost": 10500,

"availability": 156

}

],

"flights\_back": [

{

"flight\_id": 1,

"flight\_code": "FP 2100",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-10",

"time": "08:35"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-10",

"time": "10:05"

},

"cost": 10500,

"availability": 156

},

{

"flight\_id": 13,

"flight\_code": "FP 2101",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-10",

"time": "12:00"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-10",

"time": "13:35"

},

"cost": 12500,

"availability": 156

}

]

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

3. Экран оформления бронирования – на данном экране пользователь должен видеть информацию о выбранных рейсах, а также иметь возможность добавить пассажиров.

Если пользователь авторизован, то в бронировании изначально должен быть указан авторизованный пассажир.

Добавить пассажира можно нажав на кнопку “Add passenger”. Для каждого пассажира необходимо заполнить следующие поля:

a. Имя (First name)

b. Фамилия (Last name)

c. Дата рождения (Date of Birth)

d. Номер документа (Document number)

Также должна быть возможность удалить пассажира, но нельзя удалить пассажира, если он единственный.

Для подтверждения бронирования необходимо нажать на кнопку “Confirm”. После этого пользователь должен быть перенаправлен на страницу управления бронированием.

Для оформления бронирования используйте следующий запрос:

Request Response

URL: {host}/api/booking

Method: POST

Body:

{

"flight\_from": {

"id": 1,

"date": "2020-09-20"

},

"flight\_back": {

"id": 2,

"date": "2020-09-30"

},

"passengers": [

{

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890"

},

{

"first\_name": "Ivan",

"last\_name": "Gorbunov",

"birth\_date": "1990-03-20",

"document\_number": "1224567890"

}

]

} --------------------- Successful ----------------------

Status: 201

Content-Type: application/json

Body:

{

“data”: {

“code”: “QSASE”

}

}

----------------- Validation error -------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

4. Экран управления бронированием – на данном экране необходимо отобразить всю имеющуюся информацию о бронировании:

a. Информация о бронировании

i. Код бронирования

ii. Стоимость бронирования

b. Информация о перелете (для каждого рейса) – обратите внимание, что если бронирование оформлено “туда-обратно”, то рейса должно быть два (туда и обратно):

i. Номер рейса (Flight)

ii. Самолет (Aircraft)

iii. Откуда (From where)

iv. Дата и время отправления (Date and time of departure)

v. Куда (To where)

vi. Время прибытия (Arrival time)

vii. Время в полете (Flight time)

c. Информация о пассажирах

i. Имя (First name)

ii. Фамилия (Last name)

iii. Дата рождения (Date of Birth)

iv. Номер документа (Document number)

v. Место (Seat)

d. Также на этом экране присутствует кнопка для выбора места “Select seats”, при нажатии на которую пользователь должен перейти на экран с выбором места для пассажиров в бронировании.

Получить информацию о бронировании можно используя следующий запроса:

Request Response

URL: {host}/api/booking/{code}

Method: GET Status: 200

Content-Type: application/json

Body:

{

"data": {

"code": "АKIJF",

"cost": 40000,

"flights": [

{

"flight\_id": 1,

"flight\_code": "FP2100",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "08:35"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "10:05"

},

"cost": 10500,

"availability": 56

},

{

"flight\_id": 2,

"flight\_code": "FP1200",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-12",

"time": "12:00"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-12",

"time": "13:35"

},

"cost": 9500,

"availability": 56

}

],

"passengers": [

{

"id": 1,

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890",

"place\_from": "7B",

"place\_back": null

},

{

"id": 2,

"first\_name": "Ivan",

"last\_name": "Larin",

"birth\_date": "1990-03-20",

"document\_number": "1224567890",

"place\_from": null,

"place\_back": null

}

]

}

}

5. Экран с выбором места – на данном экране должна быть представлена схема самолета и список пассажиров. Должна быть возможность выбрать пассажира, а после для него место. При нажатии на кнопку “Back” экран должен смениться на предыдущий без сохранения выбранных мест. При нажатии на кнопку “Confirm” информация о выбранных местах должна быть сохранена и пользователь должен быть перенаправлен на предыдущую страницу.

Для сохранения выбранного места нужно использовать следующий запрос:

Request Response

URL: {host}/api/booking/{code}/seat

Method: PATCH

Headers

- Content-Type: application/json

Body:

{

"passenger": 1,

"seat": "7B",

"type": "from/back"

}

------------------------ Success --------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"id": 1,

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890",

"place\_from": "7B",

"place\_back": null

}

}

------------------------ Seat is occupied --------------------------

Status: 422

Content-Type: application/json

Body:

{

"error": {

"code": 422,

"message": "Seat is occupied",

}

}

------------------------ Forbidden --------------------------

Status: 403

Content-Type: application/json

Body:

{

"error": {

"code": 403,

"message": "Passenger does not apply to booking"

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

6. Регистрация – неавторизованный пользователь должен иметь возможность зарегистрироваться в системе на странице с регистрацией.

Для регистрации пользователя предусмотрен следующий запрос:

Request Response

URL: {host}/api/register

Method: POST

Headers

- Content-Type: application/json

Body:

{

“first\_name”: “Ivan”,

“last\_name”: “Ivanov”,

“phone”: “89001234567”,

“document\_number”: “7567999222”,

“password”: “paSSword”

}

------------------------- Successful -----------------------

Status: 204

---------------------- Validation error --------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

7. Вход в личный кабинет – неавторизованный пользователь должен иметь возможность войти в систему на странице со входом.

Для аутентификации пользователя предусмотрен следующий запрос:

Request Response

URL: {host}/api/login

Method: POST

Headers

- Content-Type: application/json

Body:

{

“phone”: “89001234567”,

“password”: “paSSword”

}

------------------------- Successful --------------------------

Status: 200

Content-Type: application/json

Body:

{

“data”: {

“token”: <сгенерированный token>

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

----------------------- Unauthorized -------------------------

Status: 401

Content-Type: application/json

Body:

{

“error”: {

“code”: 401,

“message”: “Unauthorized”,

“errors”: {

“phone”: [ “phone or password incorrect” ]

}

}

}

8. Личный кабинет – на странице с профилем пользователь должен увидеть следующую информацию:

a. О себе

i. Имя

ii. Фамилия

iii. Количество прошедших полетов

b. Избранные рейсы. Каждый рейс должен отображать следующие данные:

i. Город вылета

ii. Город прибытия

iii. Время отправления

iv. Время прибытия

v. Кнопка “Book now” – при нажатии на эту кнопку экран должен смениться на “Экран бронирования” для оформления нового бронирования. На экране бронирования автоматически должен быть заполнен авторизованный пассажир, а в блоке с рейсами должна быть возможность выбрать дату для рейса.

vi. Кнопка “Remove from favorites” – при нажатии на эту кнопку рейс должен быть удален из избранного.

c. Прошедшие рейсы. Каждый рейс должен отображать следующие данные:

i. Номер рейса

ii. Город вылета

iii. Город прибытия

iv. Время отправления

v. Время прибытия

vi. Кнопка “Add to favorites” – при нажатии на эту кнопку рейс должен быть добавлен в избранное.

d. Предстоящие бронирования. Каждое бронирование должно содержать следующую информацию:

i. Код бронирования

ii. Дата вылета

iii. Время вылета

iv. Время прилета

v. Откуда

vi. Куда

При нажатии на код бронирования пользователь должен перейти на страницу управления бронированием.

Для получения информации о пользователе вы можете использовать следующий запрос:

Request Response

URL: {host}/user

Method: GET

Headers

- Content-Type: application/json

- Authorization: Bearer {token}

-------------------------- Success ----------------------------

Status: 200

Content-Type: application/json

Body:

{

"first\_name": "Ivan",

"last\_name": "Ivanov",

"phone": "89001234567",

"document\_number": "1224567890"

}

----------------------- Unauthorized ------------------------

Status: 401

Content-Type: application/json

Body:

{

"error": {

"code": 401,

"message": "Unauthorized"

}

}

Для получения информации о бронированиях пользователя вы можете использовать следующий запрос:

Request Response

URL: {host}/user/booking

Method: GET

Headers

- Content-Type: application/json

- Authorization: Bearer {token}

-------------------------- Success ----------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"items": [

{

"code": "MGERS",

"cost": 40000,

"flights": [

{

"flight\_id": 1,

"flight\_code": "FP2100",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "08:35"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "10:05"

},

"cost": 10500,

"availability": 58

},

{

"flight\_id": 2,

"flight\_code": "FP1200",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-12",

"time": "12:00"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-12",

"time": "13:35"

},

"cost": 9500,

"availability": 58

}

],

"passengers": [

{

"id": 1,

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890",

"place\_from": null,

"place\_back": null

},

{

"id": 2,

"first\_name": "Ivan",

"last\_name": "Sergeev",

"birth\_date": "1990-03-20",

"document\_number": "1224567890",

"place\_from": null,

"place\_back": null

}

]

}

]

}

}

----------------------- Unauthorized ------------------------

Status: 401

Content-Type: application/json

Body:

{

"error": {

"code": 401,

"message": "Unauthorized"

}

}

9. Должна быть возможность выйти из личного кабинета.

10. Необходимо позаботиться об уведомлении пользователей о каких-либо действиях (ошибки валидации, подтверждения и т.п).

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

Разработанное приложение должно быть доступно по адресу http://xxxxxx-m2.wsr.ru/, где xxxxxx - логин участника (указан на индивидуальной карточке). 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 В. Разработка Веб-приложения на стороне сервера

**Технологии этого модуля:** REST API

**Время на выполнение**: 3 часа

**Задание:**

К вам обратилась компания «ICANFLY» - новая авиакомпания, предоставляющая услуги пассажирских авиаперевозок. Вам необходимо использовать все имеющиеся навыки в серверной разработке для создания REST API.

Заказчик хочет, чтобы api можно было легко поддерживать, поэтому использование фреймворков будет плюсом.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

Обратите внимание, что компания пока что работает только в пределах Центрального Федерального Округа, поэтому данные о времени вылета и прилета находятся в пределах одного часового пояса.

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.

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

Ваша задача – реализовать REST API, которое будет отвечать требованиям заказчика.

Для вашего удобства, во всех URL будет использоваться переменная {host} которая обозначает адрес http://xxxxxx-m2.wsr.ru/, где xxxxxx - логин участника.

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 должно быть следующим

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

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

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

}

}

}

Учтите, 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.

Регистрация

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

● first\_name – обязательное поле, строка

● last\_name – обязательное поле, строка

● phone – обязательное и уникальное поле, строка

● document\_number - обязательное, строка из 10 цифр (может быть с ведущим нулем)

● password – обязательное поле, строка

Request Response

URL: {host}/api/register

Method: POST

Headers

- Content-Type: application/json

Body:

{

“first\_name”: “Ivan”,

“last\_name”: “Ivanov”,

“phone”: “89001234567”,

“document\_number”: “7567999222”,

“password”: “paSSword”

}

------------------------- Successful --------------------------

Status: 204

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

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/login

Method: POST

Headers

- Content-Type: application/json

Body:

{

“phone”: “89001234567”,

“password”: “paSSword”

}

------------------------- Successful --------------------------

Status: 200

Content-Type: application/json

Body:

{

“data”: {

“token”: <сгенерированный token>

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

----------------------- Unauthorized -------------------------

Status: 401

Content-Type: application/json

Body:

{

“error”: {

“code”: 401,

“message”: “Unauthorized”,

“errors”: {

“phone”: [ “phone or password incorrect” ]

}

}

}

Список аэропортов

Запрос на поиск аэропортов по названию города или по IATA-коду. Поиск без учета регистра.

При отправке запроса обязательно нужно передать параметр query, который может содержать одно из следующих значений:

● название города (полное название или часть названия)

● название аэропорта (полное название или часть названия)

● IATA-код аэропорта (SVO, KZN, DME и т.д).

Все следующие запросы должны вернуть в результатах поиска аэропорт Sheremetyevo, т.к. все варианты подходят:

● GET {host}/api/airport?query=oscow

● GET {host}/api/airport?query=MOSCOW

● GET {host}/api/airport?query=Moscow

● GET {host}/api/airport?query=Sheremetyevo

● GET {host}/api/airport?query=Shere

● GET {host}/api/airport?query=SVO

Request Response

URL: {host}/api/airport

Method: GET

Query string (GET parameters):

- query ------------------------- Successful --------------------------

Status: 200

Content-Type: application/json

Body:

{

“data”: {

“items”: [

{

“name”: “Sheremetyevo”,

“iata”: “SVO”

}

]

}

}

---------------------------- No results -----------------------

Status: 200

Content-Type: application/json

Body:

{

“data”: {

“items”: []

}

}

Поиск рейсов

Запрос на поиск рейсов по указанным параметрам. Должна быть возможность передать следующие GET параметры:

● from - iata-код аэропорта вылета, обязательно, должен существовать

● to - iata-код аэропорта назначения, обязательно, должен существовать

● date1 - дата вылета туда, обязательно, в формате YYYY-MM-DD

● date2 - дата возвращения обратно, не обязательно, в формате YYYY-MM-DD

● passengers - число пассажиров (от 1 до 8 включительно), обязательно

В ответе на запрос должен быть список найденных рейсов из from в to, на которые еще остались места в заданные даты.

В поле data.flights\_to должны быть рейсы из from в to.

Если указана дата возвращения (data2), то в поле data.flights\_back должны быть обратные рейсы (из to в from), а иначе пустой массив.

В базе данных вам предоставлены рейсы и аэропорты. Дата рейсов не указана, это означает, что рейсы осуществляются ежедневно.

Request Response

URL: {host}/api/flight

Method: GET

Query string (GET parameters):

- from (SVO)

- to (KZN)

- date1 (2020-10-01)

- date2 (2020-10-13)

- passengers (2)

------------------------- Successful --------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"flights\_to": [

{

"flight\_id": 2,

"flight\_code": "FP1200",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "12:00"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "13:35"

},

"cost": 9500,

"availability": 156

},

{

"flight\_id": 14,

"flight\_code": "FP 1201",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "08:35"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "10:05"

},

"cost": 10500,

"availability": 156

}

],

"flights\_back": [

{

"flight\_id": 1,

"flight\_code": "FP 2100",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-10",

"time": "08:35"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-10",

"time": "10:05"

},

"cost": 10500,

"availability": 156

},

{

"flight\_id": 13,

"flight\_code": "FP 2101",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-10",

"time": "12:00"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-10",

"time": "13:35"

},

"cost": 12500,

"availability": 156

}

]

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

Оформление бронирования

При оформлении бронирования клиент должен передать на сервер идентификаторы рейсов из базы данных, даты рейсов (в формате YYYY-MM-DD), а также список пассажиров. Каждый пассажир должен содержать следующие поля:

● first\_name – обязательно поле, строка

● last\_name – обязательно поле, строка

● birth\_date – обязательно поле, дата в формате YYYY-MM-DD

● document\_number – обязательное поле, строка из цифр длиною в 10 символов

При создании бронирования необходимо также проверить, что на выбранных рейсах есть свободные места. Если на каком-то из рейсов недостаточно мест, то всё бронирование не может быть оформлено.

В случае успешного создания бронирования, с сервера должен вернуться уникальный код бронирования, который может состоять из пяти латинских символов.

Request Response

URL: {host}/api/booking

Method: POST

Body:

{

"flight\_from": {

"id": 1,

"date": "2020-09-20"

},

"flight\_back": {

"id": 2,

"date": "2020-09-30"

},

"passengers": [

{

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890"

},

{

"first\_name": "Ivan",

"last\_name": "Gorbunov",

"birth\_date": "1990-03-20",

"document\_number": "1224567890"

}

]

}

------------------------- Successful --------------------------

Status: 201

Content-Type: application/json

Body:

{

“data”: {

“code”: “QSASE”

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

Информация о бронировании

Получить информации о бронировании можно по коду бронирования.

Request Response

URL: {host}/api/booking/{code}

Method: GET Status: 200

Content-Type: application/json

Body:

{

"data": {

"code": "АKIJF",

"cost": 40000,

"flights": [

{

"flight\_id": 1,

"flight\_code": "FP2100",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "08:35"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "10:05"

},

"cost": 10500,

"availability": 56

},

{

"flight\_id": 2,

"flight\_code": "FP1200",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-12",

"time": "12:00"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-12",

"time": "13:35"

},

"cost": 9500,

"availability": 56

}

],

"passengers": [

{

"id": 1,

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890",

"place\_from": "7B",

"place\_back": null

},

{

"id": 2,

"first\_name": "Ivan",

"last\_name": "Larin",

"birth\_date": "1990-03-20",

"document\_number": "1224567890",

"place\_from": null,

"place\_back": null

}

]

}

}

place\_from и place\_back должны быть null пока не выбрано место.

Получение занятых мест в самолете

Данный запрос должен возвращать список занятых мест в самолете. Если обратного рейса нет то occupied\_back должен содержать пустой массив.

Request Response

URL: {host}/api/booking/{code}/seat

Method: GET ------------------------ Success --------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"occupied\_from": [

{

"passenger\_id": 1,

"place": "7B"

}

],

"occupied\_back": []

}

}

Выбор места в салоне

Данный запрос должен позволять изменить место в салоне воздушного судна на выбранный рейс для определенного пассажира.

При отправке запроса клиент должен указать ID пассажира, выбранное место и тип рейса (from/back).

Request Response

URL: {host}/api/booking/{code}/seat

Method: PATCH

Headers

- Content-Type: application/json

Body:

{

"passenger": 1,

"seat": "7B",

"type": "from/back"

}

--------------------------- Success ----------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"id": 1,

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890",

"place\_from": "7B",

"place\_back": null

}

}

------------------------ Seat is occupied --------------------------

Status: 422

Content-Type: application/json

Body:

{

"error": {

"code": 422,

"message": "Seat is occupied"

}

}

------------------------ Forbidden --------------------------

Status: 403

Content-Type: application/json

Body:

{

"error": {

"code": 403,

"message": "Passenger does not apply to booking"

}

}

---------------------- Validation error -----------------------

Status: 422

Content-Type: application/json

Body:

{

“error”: {

“code”: 422,

“message”: “Validation error”,

“errors”: {

<key>: <массив ошибок>

}

}

}

Получение своих бронирований

Данный запрос должен возвращать все бронирования пользователя. Соотнести бронирования с аутентифицированным пользователем можно по номеру документа.

Request Response

URL: {host}/user/booking

Method: GET

Headers

- Content-Type: application/json

- Authorization: Bearer {token}

----------------------------- Success --------------------------------

Status: 200

Content-Type: application/json

Body:

{

"data": {

"items": [

{

"code": "MGERS",

"cost": 40000,

"flights": [

{

"flight\_id": 1,

"flight\_code": "FP2100",

"from": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-01",

"time": "08:35"

},

"to": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-01",

"time": "10:05"

},

"cost": 10500,

"availability": 58

},

{

"flight\_id": 2,

"flight\_code": "FP1200",

"from": {

"city": "Kazan",

"airport": "Kazan",

"iata": "KZN",

"date": "2020-10-12",

"time": "12:00"

},

"to": {

"city": "Moscow",

"airport": "Sheremetyevo",

"iata": "SVO",

"date": "2020-10-12",

"time": "13:35"

},

"cost": 9500,

"availability": 58

}

],

"passengers": [

{

"id": 1,

"first\_name": "Ivan",

"last\_name": "Ivanov",

"birth\_date": "1990-02-20",

"document\_number": "1234567890",

"place\_from": null,

"place\_back": null

},

{

"id": 2,

"first\_name": "Ivan",

"last\_name": "Sergeev",

"birth\_date": "1990-03-20",

"document\_number": "1224567890",

"place\_from": null,

"place\_back": null

}

]

}

]

}

}

-------------------------- Unauthorized ----------------------------

Status: 401

Content-Type: application/json

Body:

{

"error": {

"code": 401,

"message": "Unauthorized"

}

}

Получение информации о пользователе

Request Response

URL: {host}/user

Method: GET

Headers

- Content-Type: application/json

- Authorization: Bearer {token}

------------------------------ Success -------------------------------

Status: 200

Content-Type: application/json

Body:

{

"first\_name": "Ivan",

"last\_name": "Ivanov",

"phone": "89001234567",

"document\_number": "1224567890"

}

------------------------ Unauthorized --------------------------

Status: 401

Content-Type: application/json

Body:

{

"error": {

"code": 401,

"message": "Unauthorized"

}

}

Заказчик допускает возможность изменения базы данных в будущем, поэтому вам необходимо подготовить свой вариант схемы базы данных и сохранить его в корне с модулем. Сохраните файл с названием DB.png.

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

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

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

Проверяются 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.

##### Модуль Г. Разработка ИР с использованием готовых решений

**Технологии модуля:** HTML5, CSS3, CMS WordPress, JavaScript, граф. дизайн

**Время на выполнение**: 3 часа

**Задание:**

К вам обратилась администрация городской поликлиники. Сайт центральной районной больницы давно устарел и нуждается в обновлении. Вам предоставляют готовый HTML-шаблон, который нужно немного модернизировать и разработать новый сайт с использованием системы управления контентом Wordpress.

ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!

ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ

Ваша задача – разработать сайт центральной районной больницы с использованием предоставленного HTML-шаблона, указанной структурой и требуемого функционала.

Структура главной страницы

Главная страница должна содержать следующие секции/блоки последовательно:

● Меню (навигация по странице или к отдельной странице):

○ О больнице;

○ Вызов врача на дом;

○ Заявка на льготный рецепт;

○ Департамент Здравоохранения (отдельная страница);

○ СОГАЗ Мед (отдельная страница);

○ Альфа – страхование ОМС (отдельная страница);

● Слайдер (не менее трех слайдов с использованием собственного плагина).

● О больнице (краткая информация о больнице из медиа).

● Руководители профильных служб - краткая информация о руководителях структурных подразделений (должность, ФИО, контакты)

● Наши контакты должна содержать:

○ Номер телефона;

○ Адрес ЦРБ;

○ Правила посещения;

○ Время работы;

● Форма записи на прием к врачу (собственный плагин) должна содержать:

○ Дата посещения;

○ Время посещения;

○ ФИО;

○ Выбор врача;

○ Кнопка отправки формы;

● Карта (заглушка в виде изображения из медиа).

● Подвал сайта:

○ Продублированное меню;

○ Адрес больницы;

○ Ссылки на социальные сети (должно быть реализовано с использованием стороннего плагина);

○ Номер телефона;

○ Логотипы Департамента Здравоохранения, СОГАЗ Мед, Альфа – страхования ОМС со ссылкой на соответствующие страницы сайта;

Страница с видами деятельности больницы

На странице со всеми видами деятельности отображаются все отделения больницы с пагинацией и фильтрацией по категориям:

● Диагностика;

● Лабораторные анализы;

● Пластическая хирургия;

● Эндоскопия;

● Флебология;

● Стоматология;

Страница отделения

На странице отделения отображается следующая информация:

● Главное изображение;

● Название отделения;

● Основное направление деятельности;

● Методы лечения;

Страница Департамента Здравоохранения

Информация Департаменте Здравоохранения.

Страница СОГАЗ Мед

Информация о СОГАЗ Мед.

Страница Альфа – страхование ОМС

Информация о компании Альфа – страхование ОМС.

Плагины

Вам необходимо разработать два плагина.

Форма записи на прием

Форма записи на прием – это не многофункциональный комплекс «регистратура», но, тем не менее, достойный аналог.

Форма может проверить, занят или свободен конкретно выбранный час времени Вашего специалиста.

Форма позволяет осуществить предварительную запись.

● Форма записи к врачу на прием должна содержать:

○ Дата посещения;

○ Время посещения;

○ ФИО;

○ Выбор врача;

○ Кнопка отправки формы.

Slider

Устройство слайдера:

○ Экран;

○ Средства навигации;

○ Маркеры с общим количеством слайдом и текущим состоянием;

○ Содержит картинки, рекламную информацию;

○ Меняется в автоматическом режиме;

○ Меняются в ручном режиме при нажатии на клавишу мышки или клавиатуры.

ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА

Разработанный веб-сайт должен быть доступен по адресу http://xxxxxx-m4.wsr.ru/, где xxxxxx - логин участника (указан на индивидуальной карточке).

Вся информация (например, заголовки, текст, меню и т.д.) должна редактироваться в панели управления администратором сайта без знаний программирования, верстки или доступа к файловой системе сервера.

## 

## 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;

● наушники;

● аудиофайлы с музыкальными композициями (не более 30 файлов в

формате mp3). Файлы предоставляются на флеш-носителях в день C-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 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)