**ТИПОВОЕ КОНКУРСНОЕ ЗАДАНИЕ**

***ДЛЯ РЕГИОНАЛЬНЫХ ЧЕМПИОНАТОВ***

***чемпионатного цикла 2021/2022***

**компетенции**

«ВЕБ-ТЕХНОЛОГИИ»

**для основной возрастной категории**

**16-22 года**

*Конкурсное задание включает в себя следующие разделы:*

[1.](#_heading=h.gjdgxs) Форма участия в конкурсе: 2

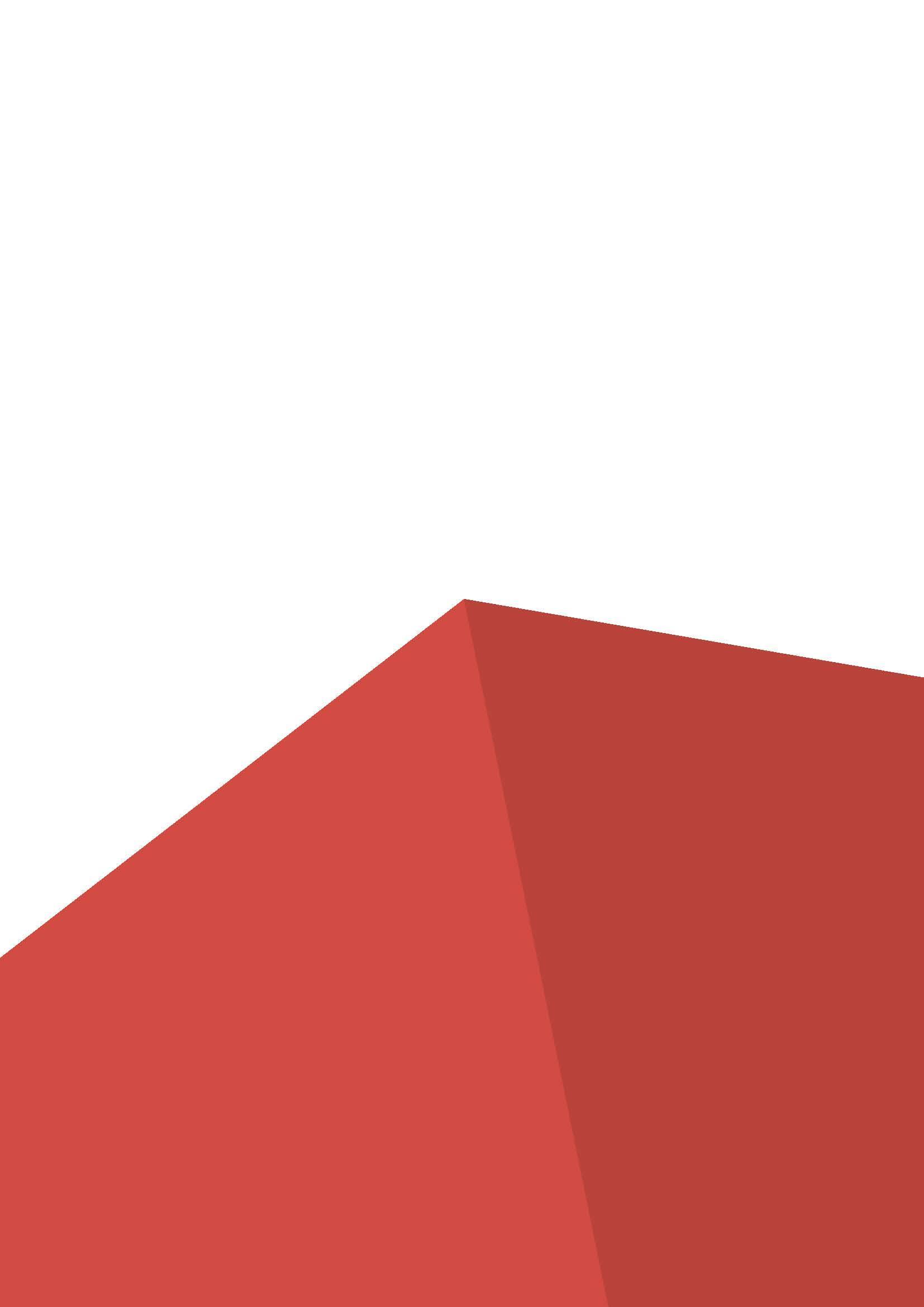
[2.](#_heading=h.30j0zll) Общее время на выполнение задания: 2

[3.](#_heading=h.1fob9te) Задание для конкурса 2

[4.](#_heading=h.3znysh7) Модули задания и необходимое время 2

[5.](#_heading=h.tyjcwt) Критерии оценки. 3

[6.](#_heading=h.3dy6vkm) Приложения к заданию. 4



1. **Форма участия в конкурсе**: Индивидуальный конкурс.
2. **Общее время на выполнение задания:** 15 ч.
3. **Задание для конкурса**

Конкурсное задание содержит 4 модуля: Дизайн и верстка веб-приложения; Серверное программирование REST API; Клиентское программирование Single Page Application; CMS WordPress.

В Конкурсном задании могут использоваться различные PHP и JavaScript фреймворки которые предусмотрены для каждого модуля. Также могут быть включены дополнительные библиотеки, которые могут быть включены в модули для реализации отдельного функционала.

Продолжительность Конкурсного задания должна быть 15 часов. Возрастной ценз участников для выполнения Конкурсного задания от 16 до 22 лет. Вне зависимости от количества модулей, Конкурсного задания включает оценку по каждому из разделов WSSS и не должно выходить за пределы WSSS. Оценка знаний участника проводится исключительно через практическое выполнение Конкурсного задания. При выполнении Конкурсного задания не оценивается знание правил и норм WSR.

Оценка производится в отношении работы модулей.

Если участник конкурса не выполняет требования техники безопасности, подвергает опасности себя или других конкурсантов, такой участник может быть отстранен от конкурса.

Детали конкурсного задания в зависимости от конкурсных условий могут быть изменены членами жюри.

Конкурсное задание должно выполняться помодульно. Оценка также происходит от модуля к модулю.

1. **Модули задания и необходимое время**

Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование модуля** | | **Соревновательный день (С1, С2, С3)** | **Время на задание** |
| **A** | Дизайн и верстка веб-приложения | С1 | 6 часов |
| **B** | Клиентское программирование Single Page Application | С2 | 3 часа |
| **C** | Серверное программирование REST API | С2 | 3 часа |
| n | CMS WordPress | С3 | 3 часа |

**Модули с описанием работ**

**Модуль A. Дизайн и верстка веб-приложения**

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

**Время на выполнение:** 6 часа

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 «Вокруг света», предоставляющая услуги пассажирских перевозок на современных комфортабельных автобусах. Главная цель компании – развить связность между городами путем организации автобусных перевозок на непостоянной основе по мере накопления запросов на перевозки.

Основные принципы работы сервиса:

1. Выбрать удобную дату отправления.
2. Накопить больше 50% бронирований на рейс.
3. Получить снижение стоимости.

Призыв: Открой Россию вместе с нами!

Вам необходимо использовать все имеющиеся навыки в дизайне и верстке, чтобы создать 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 экскурсионной компании, в основном, пользуются люди в возрасте от 17 до 40 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.

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 мест в салоне автобуса

**Главная страница (landing page)**

Главная страница должна содержать следующие блоки:

* Шапка сайта
  + Логотип компании
  + Меню навигации
    - Акции,
    - Поиск,
    - Регистрация на рейс,
    - Личный кабинет
* Секция(и) описания принципа накопления бронирований для осуществления перевозки с целью снижения его стоимости. Три принципа необходимо визуализировать при помощи анимированной инфографики на основе вводимых демо данных посетителем. Например, выбор условной даты, ввод числа бронирований и демонстрация изменения стоимости и вероятности отправления.
* Секция о доверии компании.
* Форма поиска. Должна содержать следующие поля ввода:
  + Откуда – город и адрес отправления
  + Куда – город и адрес назначения
  + Туда – дата отправления туда
  + Обратно – дата отправления обратно
  + Количество пассажиров (от 1 до 8 включительно)
  + Кнопка для поиска билетов
* Акции (список акций доступен в медиафайлах). Каждая акция должна содержать следующую информацию:
  + Изображение
  + Название акции
  + Краткое описание акции
  + Кнопка для просмотра акции
* Форма для подписки на закрытые акции. Должна содержать следующие поля:
  + Поле для ввода Email
  + Кнопка для подписки
* Подвал сайта
  + Телефон “8 (800) 100-10-10”
  + 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 автобусов). На этой странице необходимо отобразить все найденные перевозки (туда и обратно), а именно:

* Номер рейса
* Автобус
* Дата отправления
* Время отправления
* Время прибытия
* Время в пути
* Стоимость
* Вероятность отправления

Вероятность отправления – это визуальное отображение зависящие от наполненности экскурсионной перевозки в указанную дату. Если автобус заполнен на половину (30 из 60 пассажиров), то вероятность отправления – 100%. Если автобус заполнен на четверть (15 из 60 пассажиров), то вероятность отправления – 50% и т.д.

Предусмотрите способ для выбора понравившихся автобусных перевозок (одна перевозка “туда” и одна перевозка “обратно”) и кнопку для перехода к бронированию.

**Страница бронирования**

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

* Данные об автобусных перевозках (для каждой перевозки)
  + Номер рейса
  + Название города отправления
  + Адрес отправления
  + Дата отправления
  + Время отправления
  + Название города назначения
  + Адрес назначения
  + Время прибытия
  + Стоимость
* Данные о пассажирах (для каждого пассажира)
  + Имя
  + Фамилия
  + Дата рождения
  + Номер документа
* Кнопка для добавления пассажира для добавления еще одного пассажира в бронирование
* Кнопка для удаления пассажира
* Финальная стоимость
* Кнопка для оформления бронирования

**Страница управления бронированием**

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

* Информация о бронировании
  + Код бронирования
  + Стоимость бронирования
* Информация об автобусной перевозке (для каждой перевозки)
  + Номер рейса
  + Автобус
  + Название города отправления
  + Адрес отправления
  + Дата отправления
  + Время отправления
  + Название города назначения
  + Адрес назначения
  + Время прибытия
  + Время в пути
  + Вероятность отправления
* Информация о пассажирах (для каждого пассажира)
  + Имя
  + Фамилия
  + Дата рождения
  + Номер документа
  + Место в автобусе туда
  + Место в автобусе обратно
* Кнопка для выбора места

**Страница выбора места в автобусе**

На этой странице необходимо отобразить схему автобуса с возможностью выбрать место пассажира. В компании “Вокруг света” используется автобус, который имеет 2 этажа по 24 места на каждом этаже.

**Страница регистрации в личном кабинете**

На этой странице вам необходимо сделать форму со следующими полями:

* Имя
* Фамилия
* Номер документа
* Телефон
* 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 личного кабинета**

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

* Имя
* Фамилия
* Количество круизов
* Кнопка выхода из личного кабинета
* Предстоящие бронирования. Каждое бронирование должно отражать следующую информацию:
  + Код бронирования
  + Дата отправления
  + Время отправления
  + Время прибытия
  + Адрес города отправления
  + Адрес города назначения

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 мест в салоне автобуса – 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.

Если вы плохо знакомы с графическими редакторами, то вы можете сверстать страницы и отобразить свои знания дизайна в верстке, но в итоге страницы должны быть сохранены в png формате. Знания верстки оцениваться не будут.

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

**Модуль B. Клиентское программирование Single Page Application**

**Технологии этого модуля:** Клиентское программирование

**Время на выполнение**: 3 часа

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.

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://server-m3.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 автобуса

Приложение должно обладать следующим функционалом:

1. Домашний экран – на данном экране располагается статическая информация и форма для поиска подходящих рейсов со следующими полями:
   1. Откуда (From where) – город отправления
   2. Куда (To where) – город прибытия
   3. Туда (Departing) – дата отправления
   4. Обратно (Returning) – дата возвращения обратно
   5. Количество пассажиров (Passengers) – от 1 до 6

При отправке формы пользователю должен отобразиться экран с найденными рейсами.

Получить список мест отправления по введенном запросу можно с помощью следующего запроса:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/station  **Method:** GET  **Query string (GET parameters):**  - query | **------------------------- Successful --------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  “data”: {  “items”: [  {  “name”: “Cheboksary Bus Station”,  “iata”: “CSY”  }  ]  }  }  **---------------------------- No results -----------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  “data”: {  “items”: []  }  } |

1. 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:
   1. Номер рейса (Flight)
   2. Автобус (Bus)
   3. Дата и время отправления (Date and time of departure)
   4. Время прибытия (Arrival time)
   5. Время перевозки (Flight time)
   6. Общую цену, учитывая количество пассажиров (Cost)

Если пользователь не указал дату возвращения обратно (Returning), то список рейсов должен включать в себя только отправление из точки А в точку В. Если пользователь указал дату возвращения обратно (Returning), то список рейсов должен включать в себя и отправление 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 отправления – должны отображаться рейсы время отправления которых находится 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 круизов можно используя следующий запрос:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/flight  **Method:** GET  **Query string (GET parameters):**  - from *(CHB)*  - to *(KZN)*  - date1 *(2020-05-01)*  - date2 *(2020-05-01)*  - passengers *(1)* | **------------------------- Successful -----------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  " flights\_to": [  {  " flight\_id": 2,  " flight\_code": "FP1200",  "from": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2021-05-01",  "time": "05:30"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station ",  "iata": "KZN",  "date": "2021-05-01",  "time": "18:00"  },  "cost": 3000,  "availability": 156  },  {  "flight\_id": 14,  "flight\_code": "FP 1201",  "from": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2021-05-01",  "time": "21:00"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station ",  "iata": "KZN",  "date": "2021-05-02",  "time": "08:00"  },  "cost": 3000,  "availability": 156  }  ],  "flights\_back": [  {  "flight\_id": 1,  "flight\_code": "FP 2100",  "from": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2021-05-01",  "time": "21:00"  },  "to": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "KZN",  "date": "2021-05-02",  "time": "08:00"  },  "cost": 4000,  "availability": 156  },  {  "flight\_id": 13,  "flight\_code": "FP 2101",  "from": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2021-05-02",  "time": "08:00"  },  "to": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2021-05-02",  "time": "21:30"  },  "cost": 4000,  "availability": 156  }  ]  }  }  **---------------------- Validation error ------------------**  **Status:** 422  **Content-Type:** application/json  **Body:**  {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <массив ошибок>  }  }  } |

1. 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:

* 1. Имя (First name)
  2. Фамилия (Last name)
  3. Дата рождения (Date of Birth)
  4. 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": "2021-05-01"  },  "flight\_back": {  "id": 2,  "date": "2021-05-01"  },  "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>  }  }  } |

1. Экран управления бронированием – на данном экране необходимо отобразить всю имеющуюся информацию о бронировании:
   1. Информация о бронировании
      1. Код бронирования
      2. Стоимость бронирования
   2. Информация о рейсе (для каждого рейса) – обратите внимание, что если бронирование оформлено “туда-обратно”, то рейсов должно быть два (туда и обратно):
      1. Номер рейса (Flight)
      2. Автобус (Bus)
      3. Откуда (From where)
      4. Дата и время отправления (Date and time of departure)
      5. Куда (To where)
      6. Время прибытия (Arrival time)
      7. Время в перевозке (Flight time)
   3. Информация о пассажирах
      1. Имя (First name)
      2. Фамилия (Last name)
      3. Дата рождения (Date of Birth)
      4. Номер документа (Document number)
      5. Место (Seat)
   4. Также на этом экране присутствует кнопка для выбора места “Select place”, при нажатии на которую пользователь должен перейти на экран с выбором места на этаже автобуса 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": 7000,  "flights": [  {  "flight\_id": 1,  "flight\_code": "FP2100",  "from": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2020-05-01",  "time": "05:30"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "18:00"  },  "cost": 3000,  "availability": 56  },  {  "flight\_id": 2,  "flight\_code": "FP1200",  "from": {  "city": "Kazan",  "station": "Kazan",  "iata": "KZN",  "date": "2020-05-01",  "time": "21:00"  },  "to": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2020-05-02",  "time": "08:00"  },  "cost": 4000,  "availability": 56  }  ],  "passengers": [  {  "id": 1,  "first\_name": "Ivan",  "last\_name": "Ivanov",  "birth\_date": "1990-02-20",  "document\_number": "1234567890",  "place\_from": "23",  "place\_back": null  },  {  "id": 2,  "first\_name": "Tatiana",  "last\_name": "Ignateva",  "birth\_date": "1983-03-30",  "document\_number": "1224567890",  "place\_from": null,  "place\_back": null  }  ]  }  } |

1. Экран с выбором места – на данном экране должна быть представлена схема автобуса 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": "23",  "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": "23",  "place\_back": null  }  }  **------------------------ Seat is occupied -------------------**  **Status:** 422  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 422,  "message": "Place 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>  }  }  } |

1. 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>  }  }  } |

1. 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” ]  }  }  } |

1. Личный кабинет – на странице с профилем пользователь должен увидеть следующую информацию:
   1. О себе
      1. Имя
      2. Фамилия
      3. Количество пройденных рейсов
   2. Избранные рейсы. Каждый рейс должен отображать следующие данные:
      1. Город отправления
      2. Город прибытия
      3. Время отправления
      4. Время прибытия
      5. 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.
      6. Кнопка “Remove from favorites” – при нажатии на эту кнопку рейс должен быть удален из избранного.
   3. Прошедшие рейсы. Каждый рейс должен отображать следующие данные:
      1. Номер рейса
      2. Город отправления
      3. Город прибытия
      4. Время отправления
      5. Время прибытия
      6. Кнопка “Add to favorites” – при нажатии на эту кнопку рейс должен быть добавлен в избранное.
   4. Предстоящие бронирования. Каждое бронирование должно содержать следующую информацию:
      1. Код бронирования
      2. Дата отправления
      3. Время отправления
      4. Время прибытия
      5. Откуда
      6. 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": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CHB",  "date": "2020-05-01",  "time": "05:30"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "18:00"  },  "cost": 3000,  "availability": 58  },  {  "flight\_id": 2,  "flight\_code": "FP1200",  "from": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "21:00"  },  "to": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2020-05-02",  "time": "08:00"  },  "cost": 4000,  "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": "Tatiana",  "last\_name": "Ignateva",  "birth\_date": "1983-03-30",  "document\_number": "1224567890",  "place\_from": null,  "place\_back": null  }  ]  }  ]  }  }  **----------------------- Unauthorized -----------------**  **Status:** 401  **Content-Type:** application/json  **Body:**  {  "error": {  "code": 401,  "message": "Unauthorized"  }  } |

1. Должна быть возможность выйти из личного кабинета.
2. Необходимо позаботиться об уведомлении пользователей о каких-либо действиях (ошибки валидации, подтверждения и т.п).

Разработанное приложение должно быть доступно по адресу http://xxxxxx-m3.wsr.ru/, где xxxxxx - логин участника (указан на индивидуальной карточке).

**Модуль C. Серверное программирование REST API**

**Технологии этого модуля:** REST API

**Время на выполнение**: 3 часа

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 автовокзалов рейсов**

Запрос на поиск рейсов 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 автовокзала (полное название или часть названия)
* IATA-код города (CHB, KZN, DME и т.д).

Все следующие запросы должны вернуть в результатах поиска Cheboksary Bus station, т.к. все варианты подходят:

* GET {host}/api/station?query=cheboksary
* GET {host}/api station?query=CHEBOKSARY
* GET {host}/api/station?query=Cheboksary
* GET {host}/api/station?query=Cheboksary Bus station
* GET {host}/api/station?query=Cheb
* GET {host}/api/station?query=CSY

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/flight  **Method:** GET  **Query string (GET parameters):**  - query | **------------------------- Successful ---------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  “data”: {  “items”: [  {  “name”: “Cheboksary Bus station”,  “iata”: “CSY”  }  ]  }  }  **---------------------------- 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 речного порта отправления, обязательно, должен существовать
* to - iata-код речного порта 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.flight\_to должны быть рейсы из from в to.

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

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

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/flight  **Method:** GET  **Query string (GET parameters):**  - from *(CHB)*  - to *(KZN)*  - date1 *(2020-10-01)*  - date2 *(2020-10-13)*  - passengers *(2)* | **------------------------- Successful ----------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "flight\_to": [  {  "flight\_id": 2,  "flight\_code": "FP1200",  "from": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CHB",  "date": "2020-05-01",  "time": "05:30"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "18:00"  },  "cost": 3000,  "availability": 156  },  {  "flight\_id": 14,  "flight\_code": "FP 1201",  "from": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2020-05-01",  "time": "21:00"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-02",  "time": "08:00"  },  "cost": 3000,  "availability": 156  }  ],  "flight\_back": [  {  "flight\_id": 1,  "flight\_code": "FP 2100",  "from": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "21:00"  },  "to": {  "city": "Cheboksary",  " station ": "Cheboksary Bus station",  "iata": "CHB",  "date": "2020-05-02",  "time": "08:00"  },  "cost": 4000,  "availability": 156  },  {  "flight\_id": 13,  "flight\_code": "FP 2101",  "from": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-02",  "time": "08:00"  },  "to": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CHB",  "date": "2020-05-02",  "time": "20:30"  },  "cost": 4000,  "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 круизов из базы данных, даты круизов (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 круизах есть свободные места. Если на каком-то из круизов 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:**  {  "filght\_from": {  "id": 1,  "date": "2020-05-01"  },  "flight\_back": {  "id": 2,  "date": "2020-05-01"  },  "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": "Cheboksary",  "flight": "Cheboksary Bus station",  "iata": "CHB",  "date": "2020-05-01",  "time": "05:30"  },  "to": {  "city": "Kazan",  "flight": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "18:00"  },  "cost": 3000,  "availability": 56  },  {  "flight\_id": 2,  "flight\_code": "FP1200",  "from": {  "city": "Kazan",  "flight": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "21:00"  },  "to": {  "city": "Cheboksary",  "flight": "Cheboksary Bus station",  "iata": "CSY",  "date": "2020-05-02",  "time": "08:00"  },  "cost": 4000,  "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": "Tatiana",  "last\_name": "Ignateva",  "birth\_date": "1983-03-30",  "document\_number": "1224567890",  "place\_from": null,  "place\_back": null  }  ]  }  } |

place\_from и place\_back должны быть null пока не выбрана каюта.

**Получение занятых мест в автобусе**

Данный запрос должен возвращать список занятых мест в автобусе. 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": "23"  }  ],  "occupied\_back": []  }  } |

**Выбор места в автобусе**

Данный запрос должен позволять изменить место в автобусе 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": "23",  "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": "23",  "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}/api/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": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CHB",  "date": "2020-05-01",  "time": "05:30"  },  "to": {  "city": "Kazan",  "station": "Kazan Bus station",  "iata": "KZN",  "date": "2020-05-01",  "time": "18:00"  },  "cost": 4000,  "availability": 58  },  {  "flight\_id": 2,  "flight\_code": "FP1200",  "from": {  "city": "Kazan",  "station": "Kazan",  "iata": "KZN",  "date": "2020-05-01",  "time": "21:00"  },  "to": {  "city": "Cheboksary",  "station": "Cheboksary Bus station",  "iata": "CSY",  "date": "2020-05-02",  "time": "08:00"  },  "cost": 4000,  "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": "Tatiana",  "last\_name": "Ignateva",  "birth\_date": "1983-03-30",  "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}/api/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.

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

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

**Модуль D. CMS WordPress**

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

**Время на выполнение**: 3 часа

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

Ваша задача – разработать сайт библиотеки 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 библиотеке (секция);
  + Книги (отдельная страница);
  + Контакты (секция);
  + Издательство (отдельная страница);
  + Издательство (отдельная страница);
  + Издательство (отдельная страница);
* Слайдер (не менее трех слайдов с использованием стороннего плагина).
* О библиотеке (краткая информация о библиотеке из медиа).
* Интересные книги (от 6 штук, с возможностью перехода на отдельную страницу книги). Решение о публикации книги принимает администратор сайта, путем изменения признака необходимости вывода книги на главную страницу. Секция должна иметь ссылку на страницу со всеми книгами. Элементы карточки книги:
  + Главное изображение;
  + Краткое название;
  + Год выпуска;
* Наши контакты должна содержать:
  + Номер телефона;
  + Адрес библиотеки;
  + Правила посещения;
  + Время работы;
* Форма заявки на посещение библиотеки в режиме ограничительных мер от covid-19:
  + Дата посещения;
  + Время посещения;
  + Имя;
  + WhatsApp номер;
  + Кнопка отправки формы;
* Карта (заглушка в виде изображения из медиа).
* Подвал сайта:
  + Продублированное меню;
  + Адрес библиотеки;
  + Ссылки на социальные сети (должно быть реализовано с использованием стороннего плагина);
  + Номер телефона;
  + Логотипы Издательство, Издательство и Издательство с ссылкой на соответствующие страницы сайта;

**Страница со всеми книгами**

На странице со всеми книгами отображаются все книги с пагинацией и фильтрацией по категориям:

* Детектив;
* Романтика;
* Книги для детей;
* Наука и техника;
* Увлечения;
* Психология.

Карточка книги содержит:

* Главное изображение;
* Краткое название;
* Год выпуска;

**Страница книги**

На странице книги отображается следующая информация (каждый элемент должен быть представлен и настраиваться как отдельное поле или свойство книги, порядок на ваше усмотрение):

* Главное изображение;
* Дополнительные изображения;
* Полное название;
* Категория;
* Краткое название;
* Год выпуска;
* Издательство;
* Ссылки на дополнительные материалы (http external links);
* QR-код с ссылкой на текущую страницу (вставляется с помощью собственного плагина QRPage);
* Описание;

На странице книги необходимо предусмотреть возможность вывода на печать информационной таблички для дальнейшего размещения ее рядом с книгой в зале. Шаблон для печати:

**Краткое название, год выпуска**



**Полное название**

*Издательство*

Описание

**Страница «Читай-город»**

## Информация о партнере «Читай-город»

**Страница МБУК «Объединение библиотек г. Чебоксары»**

Информация о партнере МБУК «Объединение библиотек г. Чебоксары».

**Страница «Национальная библиотека ЧР»**

## Информация о партнере «Национальная библиотека Чувашской Республики».

**Плагины**

Вам необходимо разработать два плагина.

Random Line

Плагин, который при загрузке любой страницы на сайте выводит случайную строку из поста. Строка должна выводиться в случайной части страницы во время прокрутки поверх всех элементов, но не перекрывая контент. Появление должно сопровождаться анимацией и привлекать внимание, но не сильно отвлекать. Пост-источник должен указываться в настройках данного плагина в панели управления. Данный функционал планируется использовать для демонстрации малоизвестных фактах об экспонатах музея и объектов связи.

QRPage  
Плагин, который реализует шорткод [qrpage]. Шорткод должен выводить изображение с QR-кодом, который содержит ссылку на текущую страницу. QR-код должен генерироваться с помощью предоставленной JavaScript библиотеки QRCode.js.

Разработанный веб-сайт должен быть доступен по адресу http://xxxxxx-m2.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.

1. **Критерии оценки.**

Таблица 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерий** | | **Баллы** | | |
| **Судейские аспекты** | **Объективная оценка** | **Общая оценка** |
| **A** | **Дизайн и верстка веб приложения** | **19** | **20** | **39** |
| **B** | **Программирование на стороне клиента** | **2** | **19** | **21** |
| **C** | **Программирование на стороне сервера** | **1** | **19** | **20** |
| **D** | **Системы управления контентом** | **7** | **13** | **20** |
| **Итого** | | **29** | **71** | **100** |