Модуль Б. Разработка Веб-приложения на стороне клиента

# **ВВЕДЕНИЕ**

**Технологии этого модуля:** Клиентское программирование

**Время на выполнение**: 3 часа

К вам обратилась компания «FellowTraveler» - новая компания, предоставляющая услуги по поиску попутчиков в дорожных путешествиях. 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.

# **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://yddfzaez-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 автомобиле

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 – город отправления
   2. Куда – город прибытия
   3. Туда – дата выезда
   4. Обратно – дата возвращения
   5. Количество людей – от 1 до 4

При отправке формы пользователю должен отобразиться экран с найденными рейсами.

Получить список городов для поездки по введенном запросу можно с помощью следующего запроса:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/city  **Method:** GET  **Query string (GET parameters):**  - query | **---------------------Successful-------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "items": [  {  "city": "Рязань"  },  {  "city": "Москва"  },  {  "city": "Санкт-Петербург"  },  {  "city": "Севастополь"  },  {  "city": "Спас-Клепики"  },  {  "city": "Тула"  }  ]  }  }  **---------------------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
   2. Дата и время отправления
   3. Время прибытия
   4. Время в пути
   5. Общую цену, учитывая количество пассажиров

Если пользователь не указал дату возвращения обратно, то список рейсов должен включать в себя только поездки из точки А в точку В. Если пользователь указал дату возвращения обратно, то список рейсов должен включать в себя и поездки 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 «Перейти к бронированию» для перехода на экран с оформлением бронирования авто.

На данном экране должна быть предусмотрена фильтрация рейсов следующими способами:

* Фильтрация по времени выезда – 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 автомобиль, в котором не осталось свободных мест.

Получить список рейсов можно используя следующий запрос:

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/passage  **Method:** GET  **Query string (GET parameters):**  - from *(Тула)*  - to *(Москва)*  - date1 *(2021-12-05)*  - date2 *(2021-12-10)*  - passengers *(1)* | **------------------------Successful----------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "passages\_to": [  {  "passage\_id": 2,  "passage\_code": "38384",  "from": {  "city": "Тула",  "date": "2021-12-05",  "time": "01:35"  },  "to": {  "city": "Москва",  "date": "2021-12-05",  "time": "23:50"  },  "cost": 10000,  "availability": 8  }  ],  "passages\_back": [  {  "passage\_id": 8,  "passage\_code": "45322",  "from": {  "city": "Москва",  "date": "2021-12-10",  "time": "02:35"  },  "to": {  "city": "Тула",  "date": "2021-12-10",  "time": "00:50"  },  "cost": 10500,  "availability": 8  }  ]  }  }  **------------------ 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 «Добавить пассажира». Для каждого пассажира необходимо заполнить следующие поля:

* 1. Имя
  2. Фамилия
  3. Дата рождения
  4. Номер документа

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

Для подтверждения бронирования необходимо нажать на кнопку «Подтвердить». 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:**  {  "passage\_from": {  "id": 2,  "date": "2021-12-15"  },  "passage\_back": {  "id": 8,  "date": "2021-12-20"  },  "passengers": [  {  "first\_name": "Иван",  "last\_name": "Иванов",  "birth\_date": "2000-01-22",  "document\_number": "6118077983"  },  {  "first\_name": "Владимир",  "last\_name": "Попов",  "birth\_date": "1990-03-15",  "document\_number": "6117836219"  }  ]  } | **----------------------Successful-----------------------**  **Status:** 201  **Content-Type:** application/json  **Body:**  {  "data": {  "code": "LYNJT"  }  }  **--------------------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 поездке (для каждого рейса) – обратите внимание, что если бронирование оформлено «туда-обратно», то рейса должно быть два (туда и обратно):
      1. Номер автомобиля
      2. Модель автомобиля
      3. Откуда
      4. Дата и время отправления
      5. Куда
      6. Время прибытия
      7. Время в поездке
   3. Информация о пассажирах
      1. Имя
      2. Фамилия
      3. Дата рождения
      4. Номер документа
      5. Место
   4. Также на этом экране присутствует кнопка для выбора места «Выбрать место», 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": "LYNJT",  "cost": 38200,  "passages": [  {  "passage\_id": 2,  "passage\_code": "38384",  "from": {  "city": "Тула",  "date": "2021-12-15",  "time": "12:00"  },  "to": {  "city": "Москва",  "date": "2021-12-15",  "time": "14:35"  },  "cost": 9500,  "availability": 6  },  {  "passage\_id": 8,  "passage\_code": "45322",  "from": {  "city": "Москва",  "date": "2021-12-20",  "time": "11:00"  },  "to": {  "city": "Тула",  "date": "2021-12-20",  "time": "13:35"  },  "cost": 9600,  "availability": 6  }  ],  "passengers": [  {  "id": 2,  "first\_name": "Иван",  "last\_name": "Иванов",  "birth\_date": "2000-01-22",  "document\_number": "6118077983",  "place\_from": null,  "place\_back": null  },  {  "id": 3,  "first\_name": "Владимир",  "last\_name": "Попов",  "birth\_date": "1990-03-15",  "document\_number": "6117836219",  "place\_from": null,  "place\_back": null  }  ]  }  } |

1. Экран с выбором места – на данном экране должна быть представлена схема автомобиля и список пассажиров. Должна быть возможность выбрать пассажира, а после для него место. При нажатии на кнопку «Назад» экран должен смениться на предыдущий без сохранения выбранных мест. При нажатии на кнопку «Подтвердить» 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": 2,  "seat": "2B",  "type": "from/back"  } | **------------------------Success-------------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "id": 2,  "first\_name": "Иван",  "last\_name": "Иванов",  "birth\_date": "2000-01-22",  "document\_number": "6118077983",  "place\_from": "2B",  "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>  }  }  } |

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”: “Иван”,  “last\_name”: “Иванов”,  “phone”: “89997773344”,  “document\_number”: “6118077983”,  “password”: “qwerty123”  } | **------------------------Success-------------------------**  **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”: “89997773344”,  “password”: “qwerty123”  } | **------------------------Success-------------------------**  **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 прошедших поездок
   2. Избранные рейсы. Каждый рейс должен отображать следующие данные:
      1. Город отправления
      2. Город прибытия
      3. Время отправления
      4. Время прибытия
      5. Кнопка «Забронировать сейчас» – 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 “Удалить из избранного” – 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 «Добавить в избранное» – 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}/api/user  **Method:** GET  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer {token} | **------------------------Success-------------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "first\_name": "Иван",  "last\_name": "Иванов",  "phone": "89997773344",  "document\_number": "6118077983",  "number\_of\_passages": 1  }  **-------------------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}/api/user/booking  **Method:** GET  **Headers**  **- Content-Type:** application/json  **- Authorization:** Bearer {token} | **------------------------Success-------------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  "data": {  "items": [  {  "code": "LYNJT",  "cost": 38200,  "passages": [  {  "passage\_id": 2,  "passage\_code": "38384",  "from": {  "city": "Тула",  "date": "2021-12-15",  "time": "12:00"  },  "to": {  "city": "Москва",  "date": "2021-12-15",  "time": "14:35"  },  "cost": 9500,  "availability": 6  },  {  "passage\_id": 8,  "passage\_code": "45322",  "from": {  "city": "Москва",  "date": "2021-12-20",  "time": "11:00"  },  "to": {  "city": "Тула",  "date": "2021-12-20",  "time": "13:35"  },  "cost": 9600,  "availability": 6  }  ],  "passengers": [  {  "id": 2,  "first\_name": "Иван",  "last\_name": "Иванов",  "birth\_date": "2000-01-22",  "document\_number": "6118077983",  "place\_from": "2B",  "place\_back": "2B"  },  {  "id": 3,  "first\_name": "Владимир",  "last\_name": "Попов",  "birth\_date": "1990-03-15",  "document\_number": "6117836219",  "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).

# **ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА**

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