

ВВЕДЕНИЕ

Технологии этого модуля: REST API

Время на выполнение: 3 часа

К вам обратилась компания «FlightPool» - 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 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
<p>URL: {host}/api/register</p> <p>Method: POST</p> <p>Headers</p> <p>- Content-Type: application/json</p> <p>Body:</p> <pre>{ "first_name": "Ivan", "last_name": "Ivanov", "phone": "89001234567", "document_number": "7567999222", "password": "paSSword" }</pre>	<p>----- Successful -----</p> <p>Status: 204</p> <p>----- Validation error -----</p> <p>Status: 422</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "error": { "code": 422, "message": "Validation error", "errors": { <key>: <массив ошибок> } } }</pre>

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" }	<div>----- Successful ----- Status: 200 Content-Type: application/json Body: { "data": { "token": <сгенерированный token> } }</div> <div>----- Validation error ----- Status: 422 Content-Type: application/json Body: { "error": { "code": 422, "message": "Validation error", "errors": { <key>: <массив ошибок> } } }</div> <div>----- Unauthorized ----- Status: 401 Content-Type: application/json Body: { "error": { "code": 401, "message": "Unauthorized", "errors": { "phone": ["phone or password incorrect"] } } }</div>

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	<p>----- Successful -----</p> <p>Status: 200 Content-Type: application/json Body:</p> <pre>{ "data": { "items": [{ "name": "Sheremetyevo", "iata": "SVO" }] } }</pre> <p>----- No results -----</p> <p>Status: 200 Content-Type: application/json Body:</p> <pre>{ "data": { "items": [] } }</pre>

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 (date2), 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): <ul style="list-style-type: none">- from (SVO)- to (KZN)- date1 (2020-10-01)- date2 (2020-10-13)- passengers (2)	<p>----- Successful -----</p> <p>Status: 200 Content-Type: application/json Body:</p> <pre>{ "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",</pre>

```

        "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 длиной 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: <pre>{ "data": { "code": "AKIJF", "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 }] } }</pre>

	<pre>], "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 }] }</pre>
--	---

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
<p>URL: {host}/api/booking/{code}/seat</p> <p>Method: GET</p>	<pre>----- Success ----- Status: 200 Content-Type: application/json Body: { "data": { "occupied_from": [{ "passenger_id": 1, "place": "7B" }], "occupied_back": [] } }</pre>

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
<p>URL: {host}/api/booking/{code}/seat</p> <p>Method: PATCH</p> <p>Headers</p> <p>- Content-Type: application/json</p> <p>Body:</p> <pre>{ "passenger": 1, "seat": "7B", "type": "from" }</pre>	<p>----- Success -----</p> <p>Status: 200</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "data": { "id": 1, "first_name": "Ivan", "last_name": "Ivanov", "birth_date": "1990-02-20", "document_number": "1234567890", "place_from": "7B", "place_back": null } }</pre> <p>----- Seat is occupied -----</p> <p>Status: 422</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "error": { "code": 422, "message": "Seat is occupied" } }</pre> <p>----- Forbidden -----</p> <p>Status: 403</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "error": { "code": 403, "message": "Passenger does not apply to booking" } }</pre> <p>----- Validation error -----</p> <p>Status: 422</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "error": { "code": 422,</pre>

	<pre> "message": "Validation error", "errors": { <key>: <массив ошибок> } } } </pre>
--	--

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}	<pre> ----- 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", </pre>

```
        "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
<p>URL: {host}/user</p> <p>Method: GET</p> <p>Headers</p> <ul style="list-style-type: none">- Content-Type: application/json- Authorization: Bearer {token}	<p>----- Success -----</p> <p>Status: 200</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "first_name": "Ivan", "last_name": "Ivanov", "phone": "89001234567", "document_number": "1224567890" }</pre> <p>----- Unauthorized -----</p> <p>Status: 401</p> <p>Content-Type: application/json</p> <p>Body:</p> <pre>{ "error": { "code": 401, "message": "Unauthorized" } }</pre>

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/>, 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.

СИСТЕМА ОЦЕНКИ

Секция	Критерий	Сумма
A	Организация работы и управление	2,00
B	Коммуникация и навыки межличностного общения	2,00
C	Графический дизайн	0,00
D	Верстка	0,00
E	Программирование на стороне клиента	0,00
F	Программирование на стороне сервера	14,00
G	CMS	0,00
Всего		18,00