**ВВЕДЕНИЕ**

**Технологии этого модуля:** REST API

**Время на выполнение**: 3 часа

К вам обратилась компания «RailwayPool» - новая и энергичная железнодорожная компания, предоставляющая услуги пассажирских перевозок скоростными поездами. 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:
{

 “status” : ”validation error”,

 “errors” : [

 {

 “error” : <message>

 }

 ]

}

Обратите внимание, что вместо <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:

{

 “status” : ”validation error”,

 “errors” : [

 {

 “error” : “field phone can not be blank”,

 “error” : “field password can not be blank”

 }

 ]

}

Учтите, что 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:** 200**Content-Type:** application/json**Body:**{ “status” : ”successful”}**---------------------- Validation error --------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”validation error”, “errors” : [ { “error” : <message> } ]} |

**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:**{ “status” : ”successful”, “token” : <token>}**---------------------- Validation error -----------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”validation error”, “errors” : [ { “error” : <message> } ]}**----------------------- Unauthorized -------------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”unauthorized”, “error” : “incorrect login or password”} |

**Список вокзалов**

Запрос на поиск вокзалов 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 вокзала (полное название или часть названия)
* IATA-код вокзала (SVO, KZN, DME и т.д).

Все следующие запросы должны вернуть в результатах поиска вокзал Saint-Petersburg, т.к. все варианты подходят:

* GET {host}/api/airport?query=Saint-Petersburg
* GET {host}/api/airport?query=SAINT-PETERSBURG
* GET {host}/api/airport?query=saint-petersburg
* GET {host}/api/airport?query= SPT

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/stations**Method:** GET **Query string (GET parameters):**- query (*MKU*) | **------------------------- Successful --------------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”successful”, “stations” : [ { “sity” : “Moscow”, “station” : “Kursky”, “iata” : “MKU” } ]}**---------------------------- No results -----------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”successful”, “stations” : [ ]} |

**Поиск рейсов**

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

* from - iata-код вокзала отправления, обязательно, должен существовать
* to - iata-код вокзала прибытия, обязательно, должен существовать
* date1 - дата отправления 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 поле data. train \_to должны быть рейсы из from в to.

Если указана дата возвращения (data2), то в поле data. train \_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 *(PPO)*- to *(KKU)*- date1 *(2022-04-01)*- date2 *(2022-05-05)*- passengers *(2)* | **------------------------- Successful --------------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ” successful”, “flights” : [ { “train\_to” : [ “train\_id” : <train\_id>, “train\_message” : <train\_message>, “from” : [ “city” : <city>, “station” : <station>, “way” : <way>, “iata” : <iata>, “date” : <date>, “time” : <time> ], “to” : [ “city” : <city>, “station” : <station>, “way” : <way>, “iata” : <iata>, “date” : <date>, “time” : <time> ], “cost” : <cost> ], “train\_back” : [ “train\_id” : <train\_id>, “train\_message” : <train\_message>, “from” : [ “city” : <city>, “station” : <station>, “way” : <way>, “iata” : <iata>, “date” : <date>, “time” : <time> ], “to” : [ “city” : <city>, “station” : <station>, “way” : <way>, “iata” : <iata>, “date” : <date>, “time” : <time> ], “cost” : <cost> ] } ]}**---------------------- Validation error -----------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”validation error”, “errors” : [ { “error” : <message> } ]} |

**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

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:** { "train\_from": { "id": 1, "date": "2020-09-20" }, "train\_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:** 200**Content-Type:** application/json**Body:**{ “status” : “successful”, “code” : <code>}**----------------- Validation error -------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”validation error”, “errors” : [ { “error” : <message> } ]} |

**Информация о бронировании**

Получить информации о бронировании можно по коду бронирования.

|  |  |
| --- | --- |
| Request | Response |
| **URL:** {host}/api/booking/manager/**Method:** GET**Query string (GET parameters):**- code *(12563)* | **Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”successful”, "code": <code>, "cost": <cost>, "trains": [ “train\_from” : [ { "train\_id": <train\_id>, "train\_message": <train\_message>, "from": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "to": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "cost": <cost> } ], “train\_back” : [ { "train\_id": <train\_id>, "train\_message": <train\_message>, "from": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "to": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "cost": <cost> } ] ], "passengers": [ { "id": <id>, "first\_name": "<first\_name>, "last\_name": <last\_name>, "birth\_date": <birth\_date>, "document\_number": <document\_number> }, { "id": <id>, "first\_name": "<first\_name>, "last\_name": <last\_name>, "birth\_date": <birth\_date>, "document\_number": <document\_number> } ]}**----------------- Validation error -------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”validation error”, “error” : “invalid code”} |

**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}**Query string (GET parameters):**- document\_number *(1256324)* | **-------------------------- Success ----------------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”successful”, "code": <code>, "cost": <cost>, "trains": [ “train\_from” : [ { "train\_id": <train\_id>, "train\_message": <train\_message>, "from": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "to": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "cost": <cost> } ], “train\_back” : [ { "train\_id": <train\_id>, "train\_message": <train\_message>, "from": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "to": { "city": <city>, "station": <station>, “way” : <way>, "iata": <iata>, "date": <date> }, "cost": <cost> } ] ], "passengers": [ { "id": <id>, "first\_name": "<first\_name>, "last\_name": <last\_name>, "birth\_date": <birth\_date>, "document\_number": <document\_number> }, { "id": <id>, "first\_name": "<first\_name>, "last\_name": <last\_name>, "birth\_date": <birth\_date>, "document\_number": <document\_number> } ]}**---------------------- Validation error --------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : ”validation error”, “message” : <message>}**----------------------- Unauthorized ------------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : “unauthorized”, “error” : <error>} |

**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:**{ “status” : ”success”, “user” : [ { “first\_name” : <first\_name>, “last\_name” : <last\_name>, “phone” : <phone>, “document\_number” : <document\_number> } ]}**----------------------- Unauthorized ------------------------****Status:** 200**Content-Type:** application/json**Body:**{ “status” : “unauthorized”, “error” : “invalid token”} |

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 |