МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г.Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Технологии Web-программирования

Лабораторная работа №5

**REST API**

Выполнил: студент группы ВТ-41

Клесов М.И.

Проверил:

Картамышев С.В.

Белгород 2020

**Цель:** изучить основы разработки API для web-приложений. Разработать REST API для своего проекта.

**Задание к лабораторной работе:**

1. Изучить структуру формата представления данных JSON.
2. Изучить типы запросов к API: HEAD, GET, POST, PUT, DELETE.
3. Спроектировать и реализовать собственное REST API (Получение, создание, изменение и удаление каких-либо объектов).
4. В отчёт необходимо предоставить документацию к использованию методов. (Либо словесным описание, либо через Swagger)

**Выполнение**

В ходе выполнения работы было разработано REST API к проекту, а так же дополнены функционал моделей БД.

**Общее описание к методам:**

Базовый URL: <http://192.168.0.195::1149/v1>

**Контроллеры:**

1. Контроллер маршрутов RoutsController

URL: /routs

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionRouts | Возвращает все маршруты | Нет |  |  |
| /routs |
| actionRout | Возвращает маршрут по его id | id | integer | Да |
| /rout |

1. Контроллер посадочных мест SeatsController

URL: /seats

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionVoyage\_empty\_seats | Возвращает свободные посадочные места для рейса voyageId | voyageId | Integer | Да |
| /voyage\_empty\_seats |

1. Контроллер станций StationsController

URL: /stations

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionStation | Возвращает станцию по ей id | id | Integer | Да |
| /station |
| actionAll\_departs | Возвращает все станции отправления | Нет |  |  |
| /all\_departs |
| actionAll\_arrives | Возвращает все станции прибытия | Нет |  |  |
| /all\_arrives |

1. Контроллер билетов TicketsController

URL: /tickets

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionUser\_tickets | Возвращает билеты пользователя по его id | userId | Integer | Да |
| /user\_tickets |
| actionVoyage\_tickets | Возвращает билеты рейса по его id | vouageId | Integer | Да |
| /voyage\_tickets |
| actionTickets | Возвращает все билеты | Нет |  |  |
| /tickets |

Тип запроса: POST

Действия:

|  |  |  |
| --- | --- | --- |
| Название/url | Описание | Данные |
| actionBuy | Создаёт билет с привязкой к пользователю и посадочному месту по их id, устанавливает посадочное место как занятое | {userId, seatId} |
| /buy |

1. Контроллер поездов TrainsController

URL: /trains

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionVoyage\_train | Возвращает поезд для рейса по id рейса | voyageId | Integer | Да |
| /voyage\_train |
| actionTrains | Возвращает все поезда | Нет |  |  |
| /trains |

1. Контроллер пользователей UsersController

URL: /users

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionLogin | Выполняет авторизацию пользователя по его email и password\_md5. Возвращает соответствующего пользователя или исключение с кодом 406 – неверный логин или пароль | email  password\_md5 | Integer  String(32) | Да  Да |
| /login |

Тип запроса: POST

Действия:

|  |  |  |
| --- | --- | --- |
| Название/url | Описание | Данные |
| actionRegistration | Создаёт пользователя с указанным email и password\_md5, если пользователя с таким email не существует, иначе – исключение с кодом 406 | {email, password\_md5} |
| /registration |

Тип запроса: PUT

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionChange | Редактирует указанные атрибуты пользователя по его id | id  name  phone  email  password\_md5 | Integer  String(128)  String(32)  String(128)  String(32) | Да  Нет  Нет  Нет  Нет |
| /change |

1. Контроллер рейсов VoyagesController

URL: /voyages

Тип запроса: GET

Действия:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название/url | Описание | Параметры | | |
| параметр | тип | обязательный |
| actionVoyage | Возвращает рейс по его id | id | Integer | Да |
| /voyage |

Пример реализации контроллера:

<?php  
  
namespace app\modules\v1\controllers;  
use app\modules\v1\models\LoginForm;  
use app\modules\v1\models\User;  
use PharIo\Manifest\InvalidEmailException;  
use yii\db\Exception;  
use yii\web\NotAcceptableHttpException;  
  
class UsersController extends ApiController {  
 public function actionLogin($email, $password\_md5) {  
 $form = User::*find*()  
 ->where(['email' => $email])  
 ->one();  
 if ($form != null or $form != ''){  
 if ($form->checkPass($password\_md5))  
 return $form->getUser();  
 }  
 throw new NotAcceptableHttpException('Неверный логин или пароль!'); *//406* }  
  
 public function actionRegistration() {  
 $data = \Yii::*$app*->request->getBodyParams();  
 if (User::*find*()  
 ->where(['email' => $data['email']])  
 ->one() != null){  
 throw new NotAcceptableHttpException('Email занят!'); *//406  
 //return null;* }  
  
 $user = new User();  
 $user->load($data, '');  
 $str = *strpos*($user->getAttribute('email'), "@");  
 $user->setAttribute('name', *substr*($user->getAttribute('email'), 0, $str));  
 $user->save();  
 return $user;  
 }  
  
 public function actionChange($id, $name=null, $phone=null, $email=null, $password\_md5=null){  
 $user = User::*find*()  
 ->where(['id' => $id])  
 ->one();  
 if ($name != null)  
 $user->setAttribute('name', $name);  
 if ($phone != null)  
 $user->setAttribute('phone', $phone);  
 if ($email != null)  
 $user->setAttribute('email', $email);  
 if ($password\_md5 != null)  
 $user->setAttribute('password\_md5', $password\_md5);  
 $user->save();  
 return $user;  
 }  
}