<table>
<tr><td colspan="3">ID: 1</td></tr>
<tr><td colspan="3">Title: Get all bear objects list</td></tr>
<tr><td colspan="3">Description: Send GET request to /bear endpoint for getting all bear objects</td></tr>
<tr><td colspan="3">Steps</td></tr>
<tr><td colspan="2">Steps</td><td>Expected result</td></tr>
<tr><td>#1</td><td>Bear objects list is empty. Send GET request to /bear endpoint</td><td>response status_code == 200
response body == []</td></tr>
<tr><td>#2</td><td>Send POST request with correct json bear object (default) to endpoint /bear for creating new bear</td><td>response status_code == 200</td></tr>
<tr><td>#3</td><td>Send GET request to /bear endpoint</td><td>response status_code == 200
response body = list with bear object created in step #2, bear_id == step #2 response body, bear_name modified to upper case
response body has following json schema:
<pre>{
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "bear_id": {"type": "integer"},
      "bear_name": {"type": "string"},
      "bear_type": {"enum": ["POLAR", "BROWN", "BLACK", "GUMMY"]},
      "bear_age": {
        "type": "number",
        "minimum": 0,
        "maximum": 1
      }
    }
  }
}</pre></td></tr>
</table>

<table>
<tr><td colspan="3">ID: 2</td></tr>
<tr><td colspan="3">Title: Get single bear object by existing id</td></tr>
<tr><td colspan="3">Description: Send GET request to /bear/:id endpoint for getting single bear object by existing id</td></tr>
<tr><td colspan="3">Steps</td></tr>
<tr><td colspan="2">Steps</td><td>Expected result</td></tr>
<tr><td>#1</td><td>Send POST request with correct json bear object (default) to endpoint /bear for creating new bear</td><td>response status_code == 200</td></tr>
<tr><td>#2</td><td>Send GET request to /bear/:id endpoint where id == step #1 response body</td><td>response status_code == 200
response body == dict with data specified in step #1, bear_id == step #1 response body
response body has following schema:
<pre>{
  "type": "object",
  "properties": {
    "bear_id": {"type": "integer"},
    "bear_name": {"type": "string"},
    "bear_type": {
      "enum": ["POLAR", "BROWN", "BLACK", "GUMMY"]},
    "bear_age": {
      "type": "number",
      "minimum": 0,
      "maximum": 100
    }
  }
}</pre></td></tr>
</table>

<table>
<tr><td colspan="3">ID: 3</td></tr>
<tr><td colspan="3">Title: Get single bear object by not existing id and invalid id</td></tr>
<tr><td colspan="3">Description: Send GET request to /bear/:id endpoint for getting single bear object by not existing and invalid id</td></tr>
<tr><td colspan="3">Steps</td></tr>
<tr><td colspan="2">Steps</td><td>Expected result</td></tr>
<tr><td>#1</td><td>Objects list is empty. Send GET request to /bear/:id endpoint where id is not existing but valid</td><td>response status_code == 404
response body == Error message:
“Not found!”</td></tr>
<tr><td>#2</td><td>Send GET request to /bear/:id endpoint where id is invalid</td><td>response status_code == 400
response body == Error message:
“Invalid Identifier!”</td></tr>
</table>

ID: 4		
Title: Delete all bear objects list		
Description: Send DELETE request to /bear endpoint for deleting all bear objects		
Steps		
Steps		Expected result
#1	Send 3 POST requests with correct json bear objects (default) to endpoint /bear for creating new bears	response status_code == 200
#2	Send DELETE request to /bear endpoint	response status_code == 200 response body == OK
#3	Send GET request to /bear endpoint	response status_code == 200 response body = []
#4	Send DELETE request again to /bear endpoint	response status_code == 200 response body = OK

ID: 5		
Title: Delete single bear object by existing id		
Description: Send DELETE request to /bear/:id endpoint for deleting single bear object by existing id		
Steps		
Steps		Expected result
#1	Send POST requests with correct json bear object (default) to endpoint /bear for creating new bear	response status_code == 200
#2	Send DELETE request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == OK
#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 404 response body == Error message: “Not found!”

ID: 6		
Title: Delete single bear object by not existing and invalid id		
Description: Send DELETE request to /bear/:id endpoint for deleting single bear object by not existing and invalid id		
Steps		
Steps		Expected result
#1	Objects list is empty. Send DELETE request to /bear/:id endpoint where id is not existing	response status_code == 404 response body == Error message: “Not found!”
#2	Send DELETE request to /bear/:id endpoint where id is invalid	response status_code == 400 response body == Error message “Invalid Identifier!”

ID: 7		
Title: Delete single bear object by id twice		
Description: Send DELETE request to /bear/:id endpoint for deleting single bear object twice		
Steps		
Steps		Expected result
#1	Send POST requests with correct json bear object (default) to endpoint /bear for creating new bear	response status_code == 200
#2	Send DELETE request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == OK
#3	Send DELETE request to /bear/:id endpoint where id == step #1 response body again	response status_code == 404 response body == Error message: “Not found!”

ID: 8		
Title: Create bear object with correct data		
Description: Send POST request with correct json to /bear endpoint for creating new bear object		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json (default):
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send GET request to /bear/:id endpoint where id == id bear object created in step #1	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body, bear_name modified to upper case

ID: 9		
Title: Create bear object with different correct bear_type key values		
Description: Send POST request with correct json and different correct bear_type key value to /bear endpoint		
Steps		
Steps		Expected result
#1	Send POST request with correct json bear object (default) to endpoint /bear for creating new bear	response status_code == 200
#2	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body, bear_name modified to upper case
#3	Repeat Steps ##1-2 with following bear_type key values: “BROWN”, “BLACK”, “GUMMY”	

ID: 10		
Title: Create bear object with different incorrect bear_type key value data types		
Description: Send POST request with correct json and different incorrect bear_type key value data types to /bear endpoint		
Steps		
Steps		Expected result

#1	Objects list is empty. Send POST request to /bear endpoint with following json (incorrect bear_type data type):
	{
	"bear_type": true,
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 400
		response body == Error message:
		“Invalid ‘bear_type’ data type!”

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list
#3	Repeat Steps ##1-2 with following bear_type key values (incorrect bear_type data type): null, random int, [], {}.	

ID: 11		
Title: Create bear object with different incorrect bear_type key values		
Description: Send POST request with correct json and different incorrect bear_type key value to /bear endpoint		
Steps		
Steps		Expected result

#1	Objects list is empty. Send POST request to /bear endpoint
	with following json:
	{
	"bear_type": "black",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 400
		response body == Error message:
		“Invalid ‘bear_type’ value!”

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list
#3	Repeat Steps ##1-2 with following bear_type key values: “polar”, “brown”, “gummy”, random_str, “”	

ID: 12		
Title: Create bear object with different correct bear_age values		
Description: Send POST request with correct json and different correct bear_age key value to /bear endpoint		
Steps		
Steps		Expected result

#1	Send POST request to /bear/endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 0
	}
		response status_code == 200

#2	Send GET request to /bear/:id endpoint where id == id bear object created in step #1	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body, bear_name modified to upper case
#3	Repeat Steps ##1-2 with following bear_age key values: 0.01, 99.99, 100	

ID: 13		
Title: Create bear object with different incorrect bear_age values		
Description: Send POST request with correct json and different incorrect bear_age key values to /bear endpoint		
Steps		
Steps		Expected result

#1	Objects list is empty. Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": -0.01
	}
		response status_code == 400
		response body == Error message:
		“’bear_type’ value out of range!”

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list
#3	Repeat Steps ##1-2 with following bear_age key values: 100.01	

ID: 14		
Title: Create bear object with different incorrect bear_age key value data type		
Description: Send POST request with correct json and different incorrect bear_age key value data type to /bear endpoint		
Steps		
Steps		Expected result

#1	Objects list is empty. Send POST request to /bear endpoint with following json (incorrect bear_age data type)
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": true
	}
		response status_code == 400
		response body == Error message:
		“Invalid ‘bear_age’ data type!”

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list
#3	Repeat Steps ##1-2 with following bear_age key values (incorrect bear_age data type): null, random_str, [], {}	

ID: 15		
Title: Create bear object with different correct bear_name key values		
Description: Send POST request with correct json and different correct bear_name key value to /bear endpoint		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body, bear_name modified to upper case
#3	Repeat Steps #1-2 with following bear_name key values: random string	

ID: 16		
Title: Create bear object with different incorrect bear_name key value data types		
Description: Send POST request with correct json and different incorrect bear_name key value data types to /bear endpoint		
Steps		
Steps		Expected result

#3	Objects list is empty. Send POST request to /bear endpoint
	with following json (incorrect bear_name data type):
	`{`
	`  "bear_type": "BLACK",`
	`  "bear_name":  true,`
	`  "bear_age": 17.5`
	`}`
		response status_code == 400
		response body == Error message:
		“Invalid `bear_name` data type!”

#4	Send GET request to /bear endpoint	response status_code == 200 response body == empty list
#5	Repeat Step ##3-4 with following bear_name key values (incorrect bear_name data type): null, random_number, [], {}	

ID: 17		
Title: Create bear object with not supported field		
Description: Send POST request with correct json and not supported field to /bear endpoint		
Steps		
Steps		Expected result

#1	Objects list is empty. Send POST request to /bear endpoint with following json:
	`{`
	`  "bear_type": "BLACK",`
	`  "bear_name": "mikhail",`
	`  "bear_age": 17.5,`
	`  "random_str": "random_str"`
	`}`
		response status_code == 400
		response body == Error message:
		“Not supported field ‘random_str’”

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list

ID: 18		
Title: Create bear object with all incorrect values and check errors list		
Description: Send POST request with all incorrect values and check errors list		
Steps		
Steps		Expected result

#1	Object list is empty. Send POST request to /bear endpoint with following json:
	{
	"bear_type": null,
	"bear_name": null,
	"bear_age": null
	}
		response status_code == 400
		response body == Error message:
		“Invalid `bear_type` data type!
		Invalid ‘bear_name’ data type!
		Invalid ‘bear_age’ data type”

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list

ID: 19		
Title: Create bear object without required field		
Description: Send POST request without required field		
Steps		
Steps		Expected result

#1	Object list is empty. Send POST request to /bear endpoint with following json:
	{
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 400
		response body == Error. Pls fill all parameters

#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list

#3	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_age": 17.5
	}
		response status_code == 400
		response body == Error. Pls fill all parameters

#4	Send GET request to /bear endpoint	response status_code == 200 response body == empty list

#5	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail"
	}
		response status_code == 400
		response body == Error. Pls fill all parameters

#6	Send GET request to /bear endpoint	response status_code == 200 response body == empty list

ID: 20		
Title: Create bear object with empty body		
Description: Send POST request with empty body		
Steps		
Steps		Expected result
#1	Object list is empty. Send POST request to /bear endpoint with empty body	response status_code == 400 response body == Error. Pls fill all parameters
#2	Send GET request to /bear endpoint	response status_code == 200 response body == empty list

ID: 21		
Title: Check autoincrement bear_id key value after deleting		
Description: Check that after deleting an object, its id is not reused		
Steps		
Steps		Expected result
#1	Send POST requests with correct json bear object to endpoint /bear for creating new bear	response status_code == 200
#2	Send DELETE request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == OK
#3	Repeat Step #1	response status_code == 200 response body == step #1 response body + 1
#4	Send GET request to /bear/:id endpoint where id == step #3 response body	response status_code == 200 response body == dict with data specified in step #3, bear_id == step #3 response body

ID: 22		
Title: Create two bear objects with the same values		
Description: Send two POST requests with identical bodies		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	`{`
	`  "bear_type": "BLACK",`
	`  "bear_name": "mikhail",`
	`  "bear_age": 17.5`
	`}`
		response status_code == 200
		response body == id of created bear object ($id1)

#2	Repeat Step #1	response status_code == 200 response body == step #1 response body + 1 ($id2)
#3	Send two GET requests to /bear/:id endpoint where id == $id1 and $id2. Compare response bodies.	All fields except bear_id are identical. bear_id values differ by one.

ID: 23		
Title: Update single object by existing id		
Description: Send PUT request to /bear/:id endpoint for single object by existing id		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_name": "random_valid_str"
	}
		response status_code == 200
		response body == OK

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1 (except bear_name), bear_id == step #1 response body, bear_name == step #2 bear_name modified to upper case

ID: 24		
Title: Update single object by not existing and invalid id		
Description: Send PUT request to /bear/:id endpoint for single object		
Steps		
Steps		Expected result
#1	Objects list is empty. Send PUT request to /bear/:id endpoint where id is not existing	response status_code == 404 response body == Error message: “Not found!”
#2	Send PUT request to /bear/:id endpoint where id is invalid	response status_code == 400 response body == Error message: “Invalid Identifier!”

ID: 25		
Title: Update bear_type of bear object with correct values		
Description: Send PUT request to /bear/:id endpoint for updating bear_type of bear object with different correct values		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_type": "POLAR"
	}
		response status_code == 200
		response body == OK

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1 (except bear_type), bear_id == step #1 response body, bear_type == step #2 bear_type
#4	Repeat Steps ##2-3 with following bear_type key values: “GUMMY”, “BROWN”, “BLACK”	

ID: 26		
Title: Update bear_type of bear object with incorrect values		
Description: Send PUT request to /bear/:id endpoint for updating bear_type of bear object with different incorrect values		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id where id == step #1 response body with following json:
	{
	"bear_type": "polar"
	}
		response status_code == 400
		response body == Error message: “Invalid ‘bear_type’ value!”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body
#4	Repeat Steps #2-3 with following bear_type key values: “gummy”, “brown”, “black”, random_str, “”	

ID: 27		
Title: Update bear_type of bear object with incorrect value data types		
Description: Send PUT request to /bear/:id endpoint for updating bear_type of bear object with different incorrect value data types		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	`{`
	`  "bear_type": "BLACK",`
	`  "bear_name": "mikhail",`
	`  "bear_age": 17.5`
	`}`
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json (incorrect bear_type data type):
	`{`
	`  "bear_type": true`
	`}`
		response status_code == 400
		response body == Error message:
		“Invalid ‘bear_type’ data type!”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body
#4	Repeat Steps #2-3 with following bear_type key values (incorrect bear_type data type): null, randon_number, [], {}.	

ID: 28		
Title: Update bear_name of bear object with correct values		
Description: Send PUT request to /bear/:id endpoint for updating bear_name of bear object with correct values		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_name": ""
	}
		response status_code == 200
		response body == OK

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1 (except bear_name), bear_id == step #1 response body, bear_name == step #2 bear_name modified to upper case
#4	Repeat Steps #2-3 with following bear_name values: random_str	

ID: 29		
Title: Update bear_name of bear object with incorrect value data types		
Description: Send PUT request to /bear/:id endpoint for updating bear_name of bear object with incorrect value data types		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json (incorrect bear_name type):
	{
	"bear_name": true
	}
		response status_code == 400
		response body == Error message: “Invalid ‘bear_name’ data type!”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body
#4	Repeat Steps ##2-3 with following bear_name key values (incorrect bear_name data type): null, random_number, [], {}	

ID: 30		
Title: Update bear_age of bear object with correct values		
Description: Send PUT request to /bear/:id endpoint for updating bear_age of bear object with correct values		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_age": 0
	}
		response status_code == 200
		response body == OK

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1 (except bear_age), bear_id == step #1 response body, bear_age == step #2 bear_age
#4	Repeat Steps ##2-3 with following bear_age key values: 0.01, 99.99, 100	

ID: 31		
Title: Update bear_age of bear object with incorrect values		
Description: Send PUT request to /bear/:id endpoint for updating bear_age of bear object with incorrect values		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_age": -0.01
	}
		response status_code == 400
		response body == Error message:
		“’bear_age’ value is out of range!”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body
#4	Repeat Steps ##2-3 with following bear_age key values: 100.01	

ID: 32		
Title: Update bear_age of bear object with incorrect value data types		
Description: Send PUT request to /bear/:id endpoint for updating bear_age of bear object with incorrect value data types		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json (incorrect bear_age data type):
	{
	"bear_age": true
	}
		response status_code == 400
		response body == Error message:
		“Invalid ‘bear_age’ data type!”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body
#4	Repeat Steps ##2-3 with following bear_age key values (incorrect bear_age data type): null, randon_str, [], {}	

ID: 33		
Title: Update bear object with empty body		
Description: Send PUT request to /bear/:id endpoint for updating bear_age of bear object with empty body		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with empty body	response status_code == 400 response body == Error message: “Error. Pls fill at least one parameter”
#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body

ID: 34		
Title: Update all fields of bear object with correct values		
Description: Send PUT request to /bear/:id endpoint for updating all fields of bear object with correct values		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_type": "valid other than step #1",
	"bear_name": "valid other than step #1",
	"bear_age": valid other than step #1
	}
		response status_code == 200
		response body == OK

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #2, bear_id == step #1 response body

ID: 35		
Title: Update all fields of bear object with incorrect values and check errors list		
Description: Send PUT request to /bear/:id endpoint for updating all fields of bear object with incorrect values and check errors list		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"bear_type": null,
	"bear_name": null,
	"bear_age": null
	}
		response status_code == 400
		response body == Error message:
		“Invalid `bear_type` data type!
		Invalid ‘bear_name’ data type!
		Invalid ‘bear_age’ data type”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body

ID: 36		
Title: Update bear object with not supported field		
Description: Send PUT request to /bear/:id endpoint with not supported field		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send PUT request to /bear/:id endpoint where id == step #1 response body with following json:
	{
	"random_str": "random_str"
	}
		response status_code == 400
		response body == Error message:
		“Not supported field ‘random_str’!”

#3	Send GET request to /bear/:id endpoint where id == step #1 response body	response status_code == 200 response body == dict with data specified in step #1, bear_id == step #1 response body

ID: 37		
Title: Update bear object with data like another bear object		
Description: Send PUT request to /bear/:id endpoint with data like another bear object		
Steps		
Steps		Expected result

#1	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200

#2	Send POST request to /bear endpoint with following json:
	{
	"bear_type": "valid other than step #1",
	"bear_name": "valid other than step #1",
	"bear_age": valid other than step #1,
	}
		response status_code == 200

#3	Send PUT request to /bear/:id endpoint where id == step #2 response body with following json:
	{
	"bear_type": "BLACK",
	"bear_name": "mikhail",
	"bear_age": 17.5
	}
		response status_code == 200
		response body == OK

#4	Send two GET requests to /bear/:id endpoint where id == step #1 response body and step #2 response body. Compare response bodies.	All fields except bear_id are identical