**Assumptions**:

* Tests to be created in Postman
* Content-Type:application/json

**Tests**:

1. **GET request when at least one employee stored in DB**

//*This is a test to verify that the endpoint for GET of all employees is working. Should return a list of all available employees.*

*//Precondition: there is at least one employee stored in DB*

*//GET request:* [https://endpoint.com/employee/](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

1. **GET request when no employee stored in DB**

*//Precondition: there is no employees stored in DB*

*//GET request:* [https://endpoint.com/employee/](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

pm.test(“Number of employees”,function(){

pm.expect(pm.response.json().length).to.eql(0);

});

1. **POST** of one employee with all mandatory fields, with only one address  
      
   Body:  
   {  
    “firstName”: “Pawel”,  
    “lastName”: “Rakowiecki”,  
    “age”: 18,  
    “gender”: “Male”,  
    “addresses”:

[{

“street”:“Piękna 64C/11”,

“postcode”: “50-506”,

“city”: “Wrocław”

}]

}

Assert:  
 pm.test("Status code is 201", function ()   
 {  
 pm.response.to.have.status(201);  
 });  
   
 //then checking if new employee has been added by **GET** request, query value

[https://endpoint.com/employee?firstName=Pawel&lastName=Rakowiecki](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)

pm.test("Status code is 200", function ()

{

pm.response.to.have.status(200);

});

1. **POST** with two addresses, then verification with GET request if both addresses are stored  
   Body:  
   {  
    “firstName”: “Piotr”,  
    “lastName”: “Pacyna”,  
    “age”: “33”,  
    “gender”: “Male”,  
    “addresses”:

[{

“street” : “Wiejska 11”,

“postcode” : “12-345”,

“city” : “Warszawa”

},

{

“street” : “Zielona 12”

“postcode” : “44-987”

“city” : “Jelonka”

}]

}

Assert:  
 pm.test("Status code is 201", function ()   
 {  
 pm.response.to.have.status(201);  
 });

//then checking if employee with two addresses have been stored by GET request

[https://endpoint.com/employee?firstName=Piotr&lastName=Pacyna](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)

let response = pm.response.json();

pm.test("Address count test", function ()

{

pm.expect(response.results[0].addresses.length).to.eql(2);

});

1. **Negative test of POST request** when not all mandatory fields provided

Body:  
{  
 “firstName”: “Adam”,  
 “lastName”: “Kowalski”,  
 “age”: “33”,  
 “gender”: “Male”,  
}

Assert:  
 pm.test("Status code is 400", function ()   
 {  
 pm.response.to.have.status(400);  
 });

1. **Negative test of POST** **request for 11th employee**  
      
   //first request to check if DB contains 10 objects/employees by running GET request

[https://endpoint.com/employee](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)/

pm.test(“Number of employees”,function(){

pm.expect(pm.response.json().length).to.eql(10);

});

//sending new POST request should return request error

pm.test("Status code is 400", function ()

{

pm.response.to.have.status(400);

});

1. **PATCH**

//Update age of one employee with firstName “Anna” and lastName “Kochanowska”

//first by POST request new employee is added

Body:  
{  
 “firstName”: “Anna”,  
 “lastName”: “Kochanowska”,  
 “age”: 18,  
 “gender”: “Female”,  
 “addresses”:

[{

“street”:“Nyska 44”,

“postcode”: “51-516”,

“city”: “Wrocław”

}]

}

Tests:

pm.globals.set("new\_age", 90);

//then by running PATCH request, age is being updated

Body:

{

“age”: {{new\_age}}

}

Tests:

pm.test("Status code is 202", function ()

{

pm.response.to.have.status(202);

});

//verify if age has been updated by new value using GET

[https://endpoint.com/employee?firstName=Anna&lastName=](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)Kochanowska

let response = pm.response.json();

pm.test("Address count test", function ()

{

pm.expect(response.results[0].age).to.eql(pm.globals.get(new\_age));

});

1. **PUT**

//first by POST request new employee is created with only one address

Body:  
{  
 “firstName”: “Krzystof”,  
 “lastName”: “Malina”,  
 “age”: “30”,  
 “gender”: “Male”,  
 “addresses”:

[{

“street” : “Kościelna 551”,

“postcode” : “57-345”,

“city” : “Szklary”

}]

}

Assert:  
 pm.test("Status code is 201", function ()   
 {  
 pm.response.to.have.status(201);  
 });

//then by PUT request the same employee gets second address

Body:  
{  
 “firstName”: “Krzystof”,  
 “lastName”: “Malina”,  
 “age”: “30”,  
 “gender”: “Male”,  
 “addresses”:

[{

“street” : “Kościelna 551”,

“postcode” : “57-345”,

“city” : “Szklary”

},

{

“street” : “Szybowcowa 33”,

“postcode” : “57-505”,

“city” : “Szczecin”

}]

}

Assert:  
 pm.test("Status code is 200", function ()   
 {  
 pm.response.to.have.status(200);  
 });

//then checking if employee has been updated by adding second address by GET request

[https://endpoint.com/employee?firstName=Krzysztof&lastName](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)=Malina

let response = pm.response.json();

pm.test("Address count test", function ()

{

pm.expect(response.results[0].addresses.length).to.eql(2);

});

1. **DELETE**  
    //first GET request to get number of employees

[https://endpoint.com/employee/](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)

let employee\_response = pm.response.json();

pm.globals.set("number\_of\_employee", pm.response.json()).length);

//then DELETE request by query value

[https://endpoint.com/employee?firstName=Pawel&lastName=Rakowiecki](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)  
 pm.test("Status code is 204", function ()

{  
 pm.response.to.have.status(204);  
 });

//second GET request to check if number of employees has decreased by 1

[https://endpoint.com/employee/](https://endpoint.com/employee/?FirstName=Pawel&LastName=Rakowiecki)

pm.test(“Number of employees”,function(){

pm.expect(pm.response.json().length).equal.to(pm.globals.get(number\_of\_employee)-1)

});