

Evidencias Servicio Rest API Person

Tabla de contenido

[1. Estructura de documentos en MongoDB](#)

[1.1. Entidad Persona y familiares:](#)

[1.2. Entidad de Auditoría](#)

[2. Evidencias de pruebas](#)

[2.1. Crear persona](#)

[2.1.1. Request](#)

[2.1.2. Response](#)

[2.1.3. Evidencia Mongo DB](#)

[2.1.4. Auditoria](#)

[2.2. Consultar persona por id](#)

[2.2.1. Request y Response](#)

[2.2.2. Evidencia Mongo DB Auditoria](#)

[2.3. Consultar todas las personas](#)

[2.3.1. Request y Response](#)

[2.3.2. Auditoria](#)

[2.4. Actualizar personas](#)

[2.4.1. Request y Response](#)

[2.4.1. Auditoria](#)

[2.5. Eliminar personas](#)

[2.5.1. Request y Response](#)

[2.5.1. Auditoria](#)

1. Estructura de documentos en MongoDB

1.1. Entidad Persona y familiares:

```
object ▶ persona ▶ relativos ▶ 1 ▶
└─ object {5}
  └─ persona {5}
    id : 1000000
    name : Bart
    lastName : Simpson
    birthDate : 2020-01-11
    relativos {5}
      0 {2}
        relativeType : AUNT
        person {3}
          name : Selma
          lastName : Bouvier
          birthDate : 2020-01-11
      1 {2}
        relativeType : SISTER
        person {3}
      2 {2}
      3 {2}
      4 {2}
```

1.2. Entidad de Auditoría

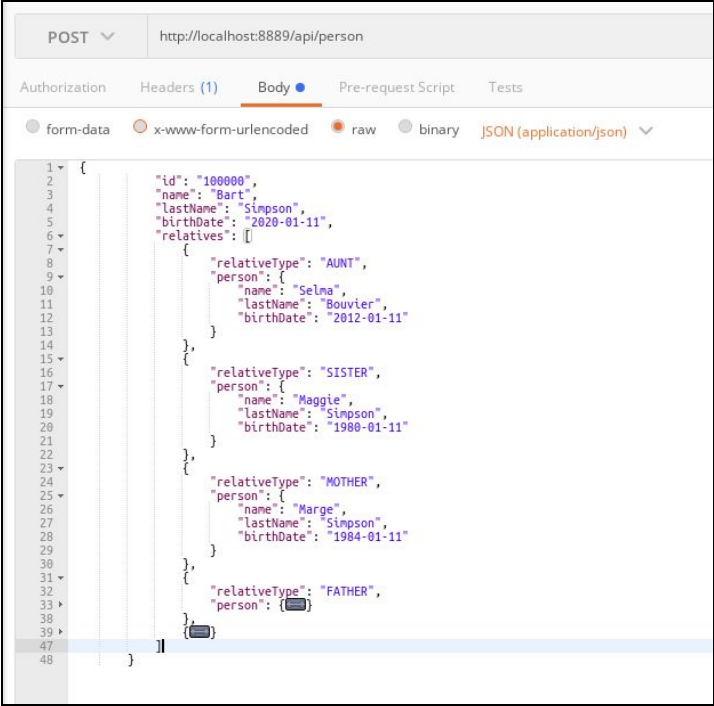
```
object ▶ persona ▶ relativos ▶ 1 ▶
└─ object {6}
  _id : 5e1aeaa26665f3535ebe131a
  time : 2020-01-12T09:45:06.384Z
  eventType : INPUT
  inputParams : PersonDto(id=100000, name=Bart,
    lastName=Simpson, birthDate=2020-01-11,
    relativos=[RelativeDto(relativeType=AUNT,
      person=PersonDto(id=null, name=Selma,
        lastName=Bouvier, birthDate=2020-01-11,
        relativos=null)),
      RelativeDto(relativeType=SISTER,
        person=PersonDto(id=null, name=Maggie,
          lastName=Simpson, birthDate=1980-01-11,
          relativos=null)),
      RelativeDto(relativeType=MOTHER,
        person=PersonDto(id=null, name=Marge,
          lastName=Simpson, birthDate=1984-01-11,
          relativos=null)),
      RelativeDto(relativeType=FATHER,
        person=PersonDto(id=null, name=Homer,
          lastName=Simpson, birthDate=2020-01-11,
          relativos=null)),
      RelativeDto(relativeType=GRANDFATHER,
        person=PersonDto(id=null, name=Abraham,
          lastName=Simpson, birthDate=2020-01-11,
          relativos=null))])
  message : Method [POST] RemoteAddress: [172.25.0.1]
  _class : co.com.nxs.person.entities.Audit
```

2. Evidencias de pruebas

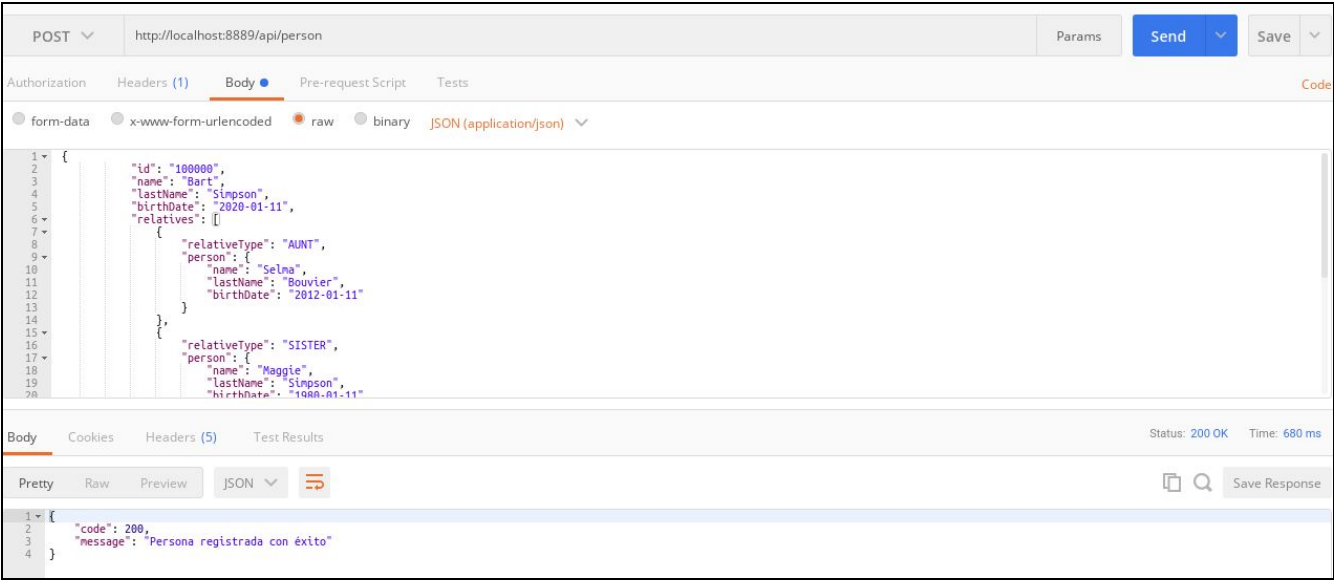
2.1. Crear persona

POST /api/person: Crear una persona con sus familiares (relatives)

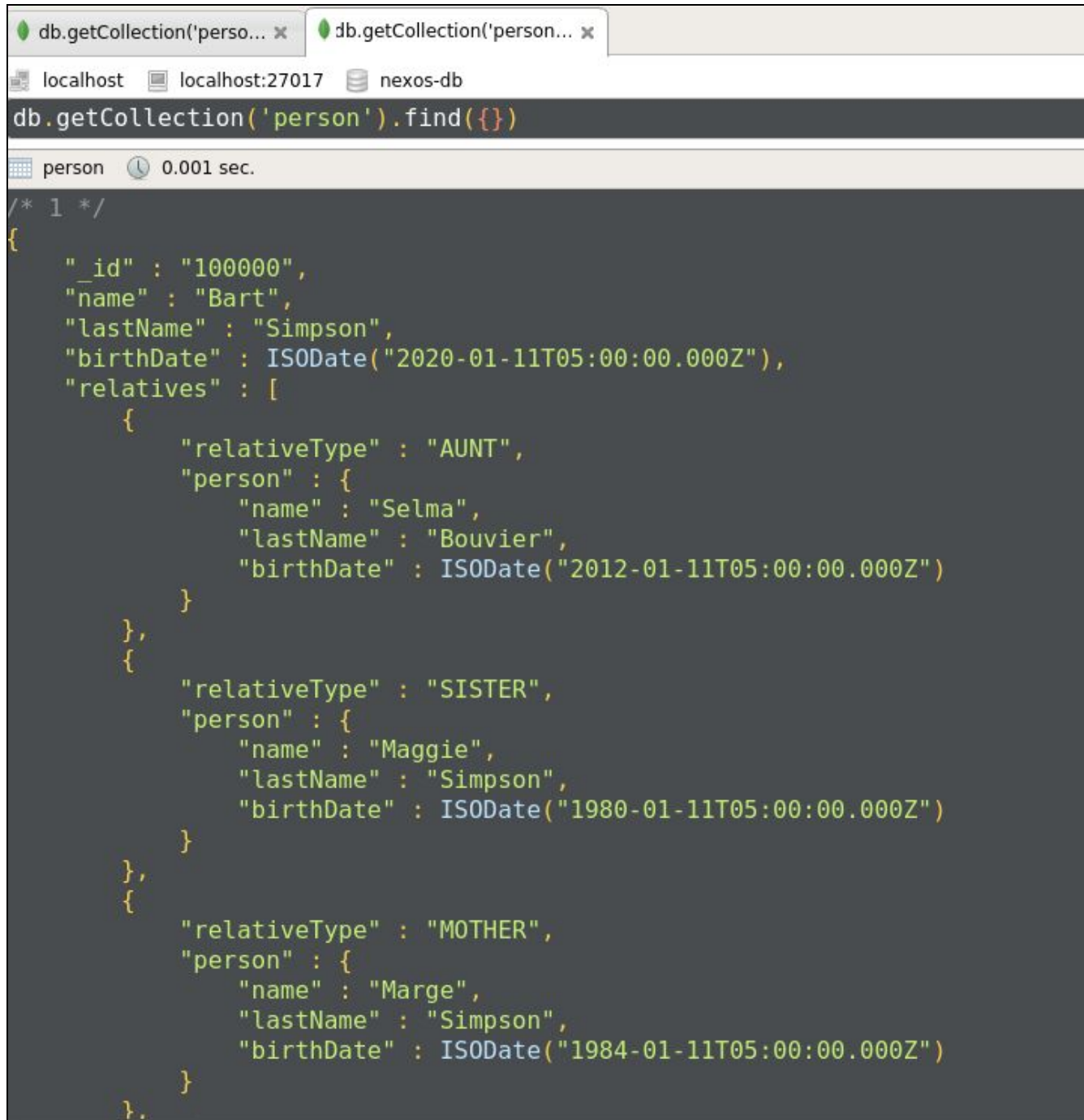
2.1.1. Request



2.1.2. Response



2.1.3. Evidencia Mongo DB



The screenshot shows a MongoDB web interface with two tabs at the top, both containing the text `db.getCollection('perso... x`. Below the tabs, the address bar shows `localhost`, `localhost:27017`, and `nexos-db`. The main command area contains the query `db.getCollection('person').find({})`. Below this, a status bar indicates the collection `person` and the execution time `0.001 sec.`. The result is displayed in a dark-themed code editor, showing a single document with the following fields: `_id` (100000), `name` (Bart), `lastName` (Simpson), `birthDate` (ISODate("2020-01-11T05:00:00.000Z")), and `relatives` (an array of three objects representing family members: Aunt Selma Bouvier, Sister Maggie Simpson, and Mother Marge Simpson).

```
/* 1 */
{
  "_id" : "100000",
  "name" : "Bart",
  "lastName" : "Simpson",
  "birthDate" : ISODate("2020-01-11T05:00:00.000Z"),
  "relatives" : [
    {
      "relativeType" : "AUNT",
      "person" : {
        "name" : "Selma",
        "lastName" : "Bouvier",
        "birthDate" : ISODate("2012-01-11T05:00:00.000Z")
      }
    },
    {
      "relativeType" : "SISTER",
      "person" : {
        "name" : "Maggie",
        "lastName" : "Simpson",
        "birthDate" : ISODate("1980-01-11T05:00:00.000Z")
      }
    },
    {
      "relativeType" : "MOTHER",
      "person" : {
        "name" : "Marge",
        "lastName" : "Simpson",
        "birthDate" : ISODate("1984-01-11T05:00:00.000Z")
      }
    }
  ]
}
```

2.1.4. Auditoria

db.getCollection('perso... x db.getCollection('perso... x db.getCollection('logger'... x

localhost localhost:27017 nexos-db

db.getCollection('logger').find({})

logger 0.001 sec. 0 50

```
/* 1 */
{
  "_id" : ObjectId("5e1aeaa26665f3535ebe131a"),
  "time" : ISODate("2020-01-12T09:45:06.384Z"),
  "eventType" : "INPUT",
  "inputParams" : "PersonDto(id=100000, name=Bart, lastName=Simpson, birthDate=2020-01-11, relatives=[RelativeDto(relativeType=AUNT, pe",
  "message" : "Method [POST] RemoteAddress: [172.25.0.1]",
  "_class" : "co.com.nxs.person.entities.Audit"
}

/* 2 */
{
  "_id" : ObjectId("5e1aeaa26665f3535ebe131b"),
  "time" : ISODate("2020-01-12T09:45:06.733Z"),
  "eventType" : "OUTPUT",
  "outputParams" : "ResponseDto(code=200, message=Persona registrada con éxito, responseCode=null, data=null)",
  "message" : "Method [POST] RemoteAddress: [172.25.0.1]",
  "_class" : "co.com.nxs.person.entities.Audit"
}
```

2.2. Consultar persona por id

GET /api/person/{id} : Obtener la información de una persona identificada por {id}

2.2.1. Request y Response

Filter

History Collections

All Me Team

nexos 3 requests

GET Get all persons

GET Get persons by Id

POST create person

Postman Echo 37 requests

Get all persons create person Get persons by Id + ...

No Environment

Examples (0)

GET http://localhost:8889/api/person/100000 Params Send Save

Authorization Headers Body Pre-request Script Tests

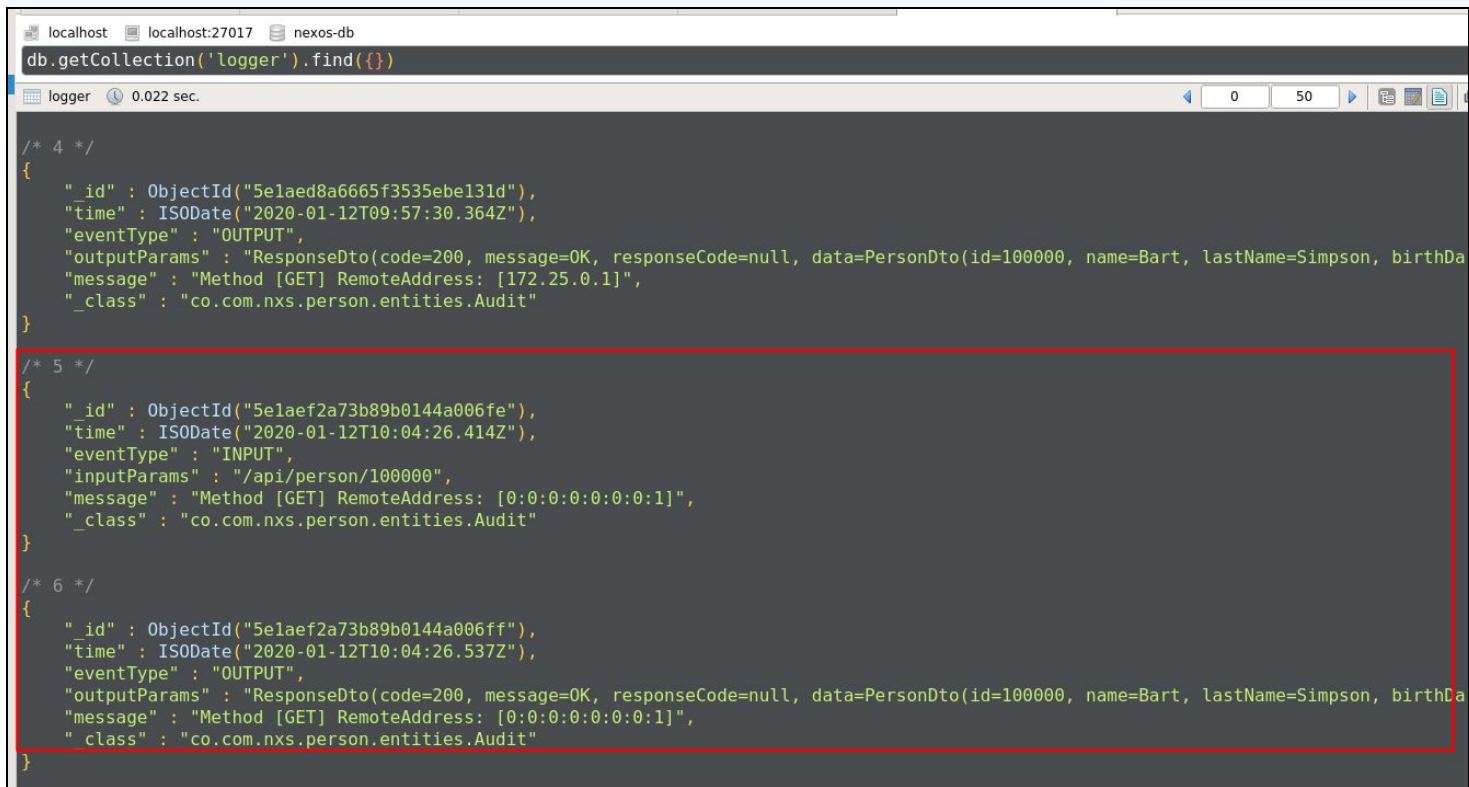
Type No Auth

Body Cookies Headers (5) Test Results Status: 200 OK Time: 79 ms

Pretty Raw Preview JSON Save Response

```
1 {
2   "code": 200,
3   "message": "OK",
4   "data": {
5     "id": "100000",
6     "name": "Bart",
7     "lastName": "Simpson",
8     "birthDate": "2020-01-11",
9     "relatives": [
10      {
11        "relativeType": "AUNT",
12        "person": {
13          "name": "Selma",
14          "lastName": "Bouvier",
15          "birthDate": "2012-01-11"
16        }
17      },
18      {
19        "relativeType": "SISTER",
20        "person": {
21          "name": "Maggie",
22          "lastName": "Simpson",
23          "birthDate": "1980-01-11"
24        }
25      },
26      {
27        "relativeType": "MOTHER",
28        "person": {
29          "name": "Marge",
30          "lastName": "Simpson",
31          "birthDate": "1984-01-11"
32        }
33      }
34    ]
35  }
36 }
```

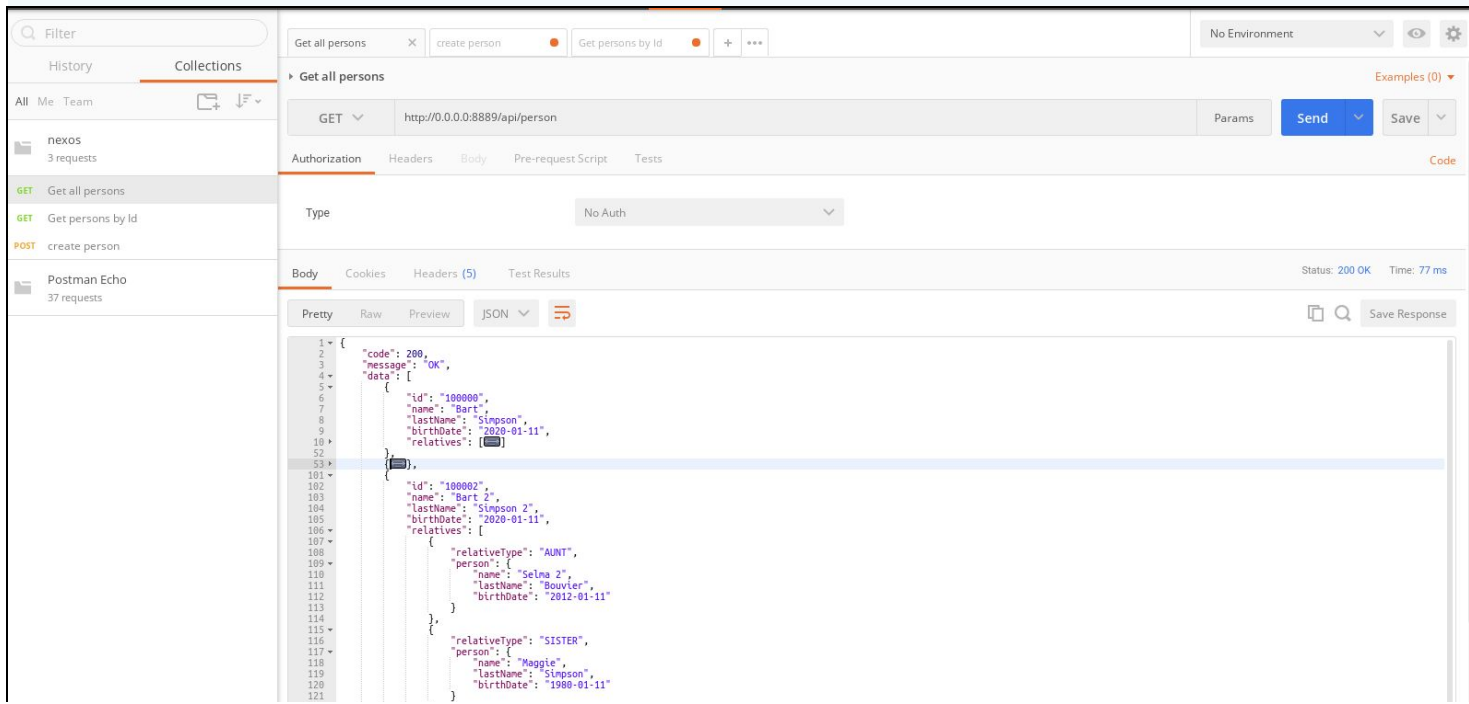
2.2.2. Evidencia Mongo DB Auditoria



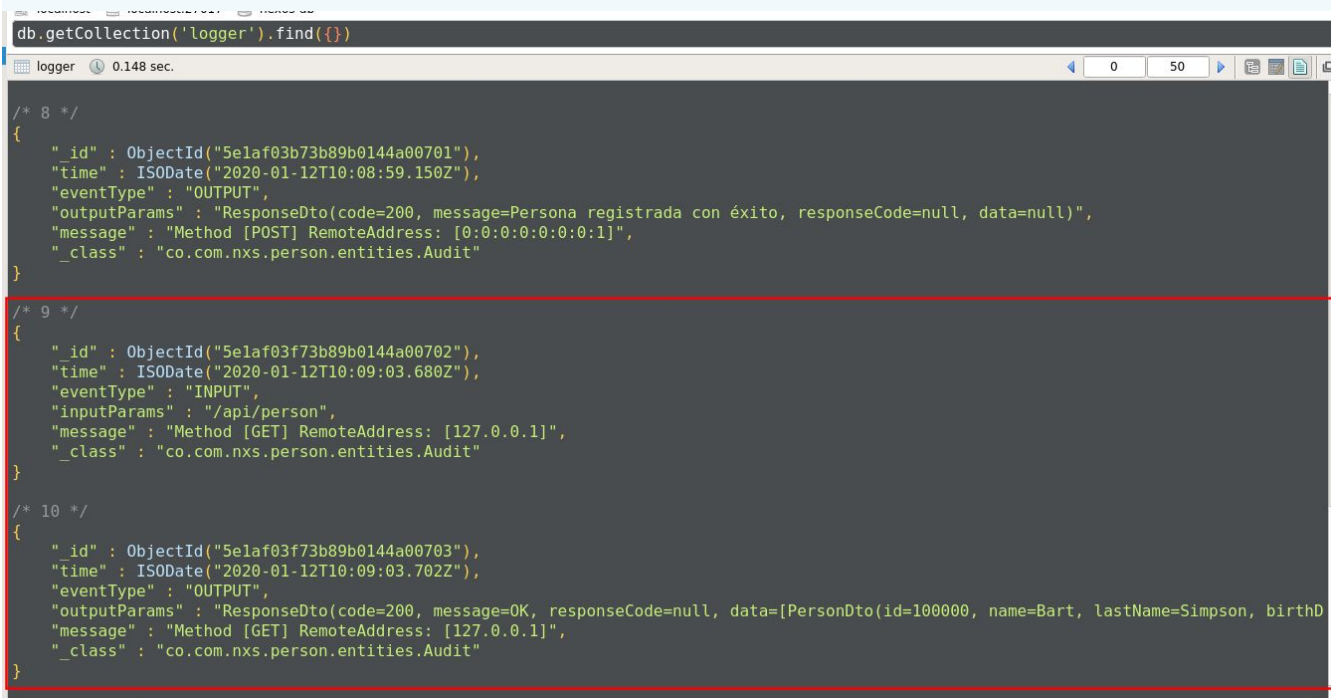
2.3. Consultar todas las personas

GET /api/person: Obtener todas las personas

2.3.1. Request y Response



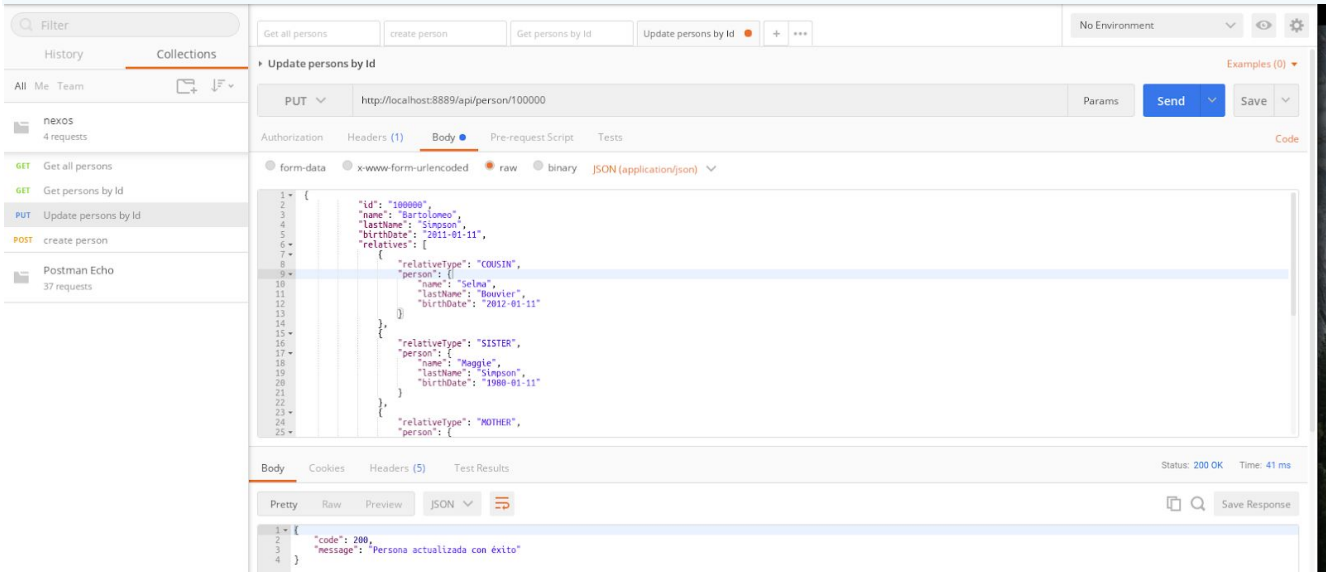
2.3.2. Auditoria



2.4. Actualizar personas

PUT `/api/person/{id}` : Actualizar la información de una persona identificada por `{id}`

2.4.1. Request y Response



localhost localhost:27017 nexos-db

```
db.getCollection('person').find({
  "_id" : "100000"
})
```

person 0.001 sec.

Key	Value	Type
(1) 100000	{ 6 fields }	Object
_id	100000	String
name	Bartolomeo	String
lastName	Simpson	String
birthDate	2011-01-11 05:00:00.000Z	Date
relatives	[5 elements]	Array
[0]	{ 2 fields }	Object
relativeType	COUSIN	String
person	{ 3 fields }	Object
name	Selma	String
lastName	Bouvier	String
birthDate	2012-01-11 05:00:00.000Z	Date
[1]	{ 2 fields }	Object
[2]	{ 2 fields }	Object
[3]	{ 2 fields }	Object
[4]	{ 2 fields }	Object
_class	co.com.nxs.person.entities.Person	String

2.4.1. Auditoria

localhost localhost:27017 nexos-db

```
db.getCollection('logger').find({})
```

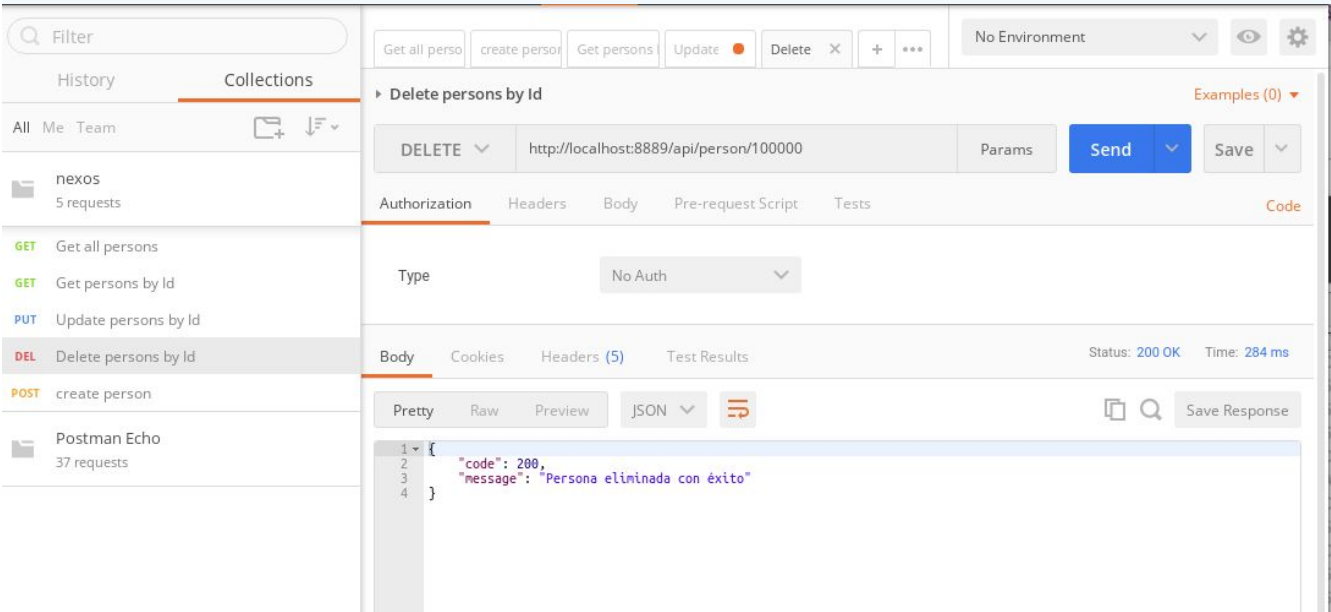
logger 0.058 sec.

/* 10 */	{ "id": ObjectId("5e1af03f73b89b0144a00703"), "time": ISODate("2020-01-12T10:09:03.702Z"), "eventType": "OUTPUT", "outputParams": "ResponseDto(code=200, message=OK, responseCode=null, data=[PersonDto(id=100000, name=Bart, lastName=Simpson, birthD", "message": "Method [GET] RemoteAddress: [127.0.0.1]", "_class": "co.com.nxs.person.entities.Audit" }
/* 11 */	{ "id": ObjectId("5e1af1df73b89b0144a00704"), "time": ISODate("2020-01-12T10:15:59.191Z"), "eventType": "INPUT", "inputParams": "PersonDto(id=100000, name=Bartolomeo, lastName=Simpson, birthDate=2011-01-11, relatives=[RelativeDto(relativeType=C0", "message": "Method [PUT] RemoteAddress: [0:0:0:0:0:0:1]", "_class": "co.com.nxs.person.entities.Audit" }
/* 12 */	{ "id": ObjectId("5e1af1df73b89b0144a00705"), "time": ISODate("2020-01-12T10:15:59.198Z"), "eventType": "OUTPUT", "outputParams": "ResponseDto(code=200, message=Persona actualizada con éxito, responseCode=null, data=null)", "message": "Method [PUT] RemoteAddress: [0:0:0:0:0:0:1]", "_class": "co.com.nxs.person.entities.Audit" }

2.5. Eliminar personas

DELETE /api/person/{id} : Eliminar la información de una persona identificada por {id}

2.5.1. Request y Response



2.5.1. Auditoria

