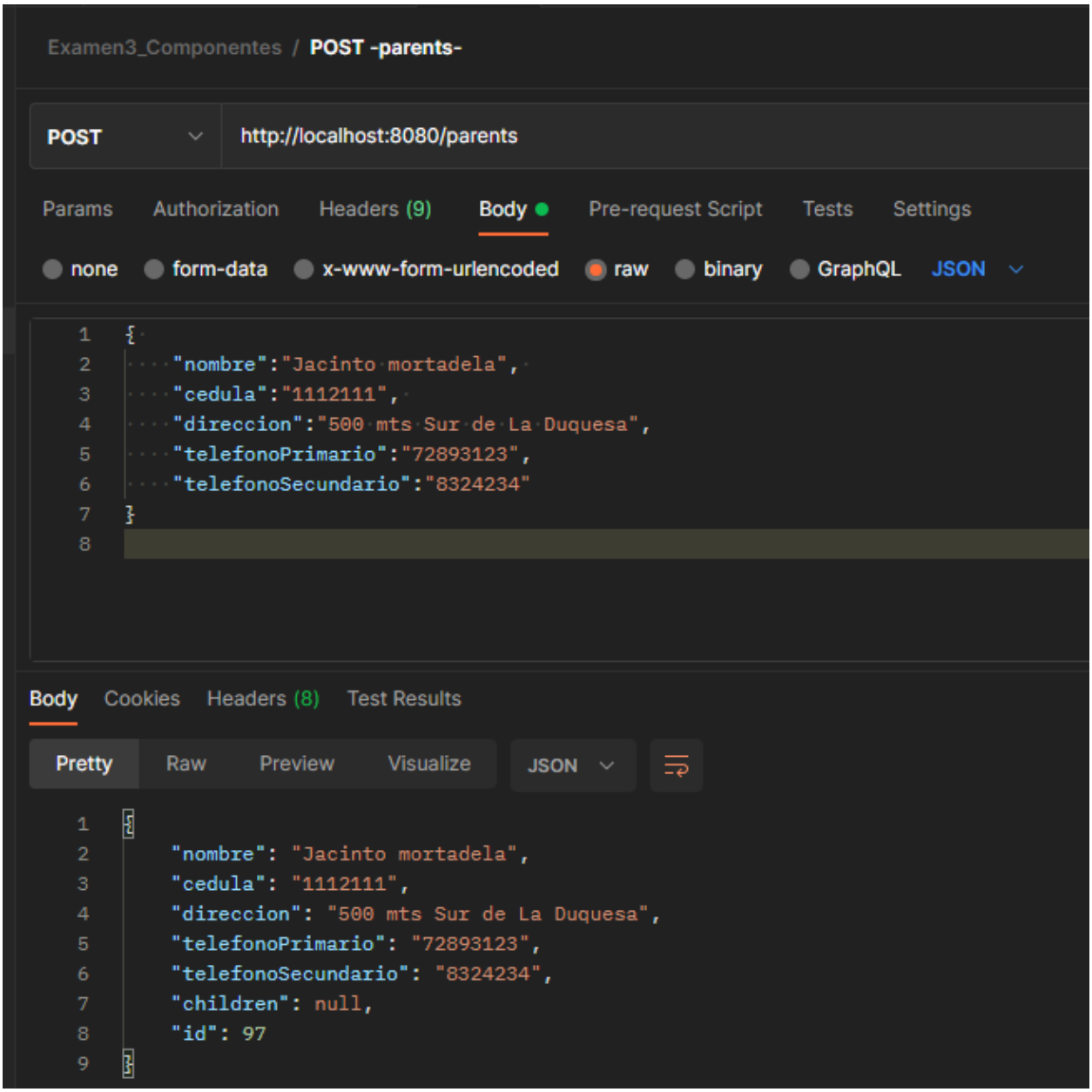
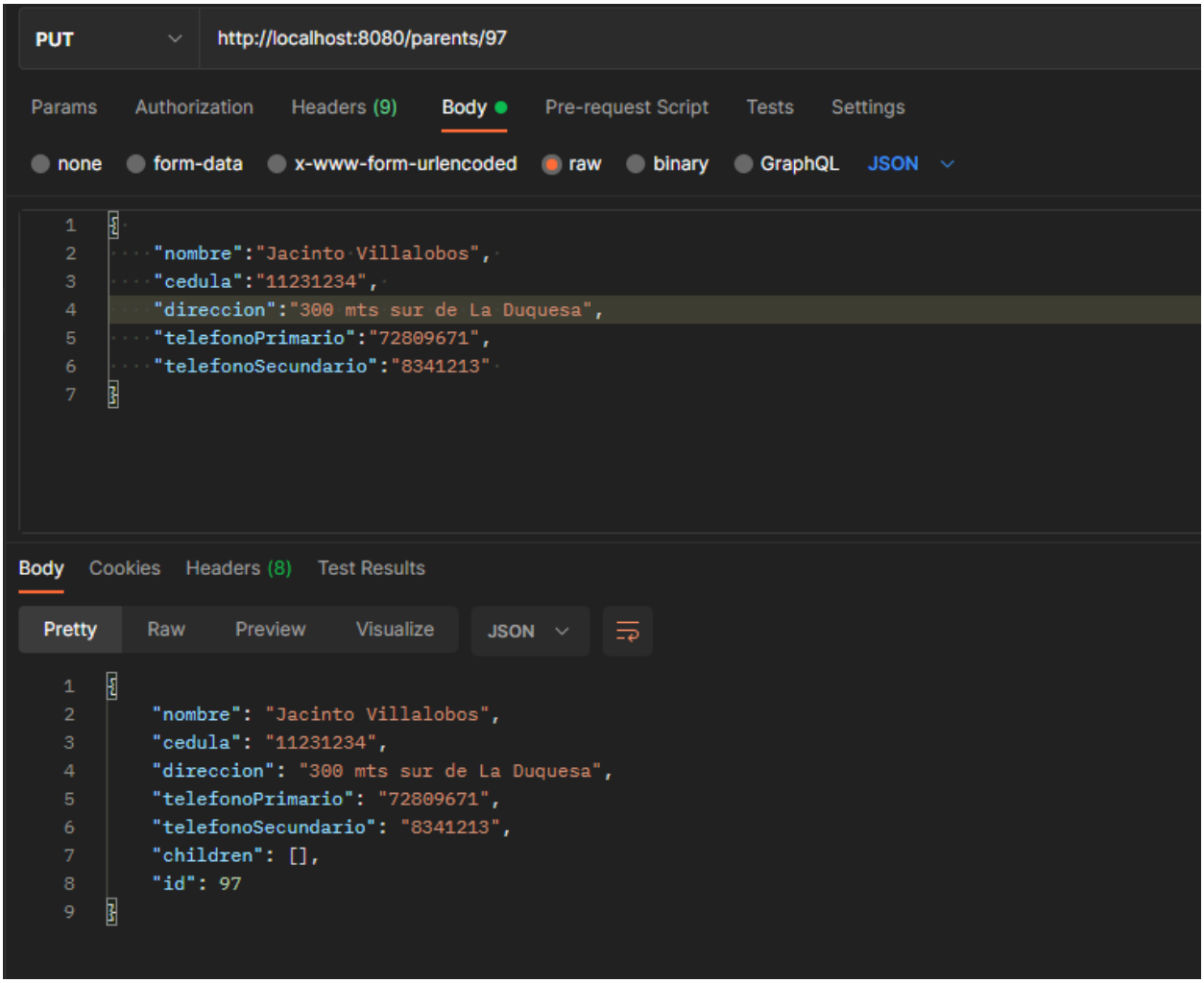
Evidencias:

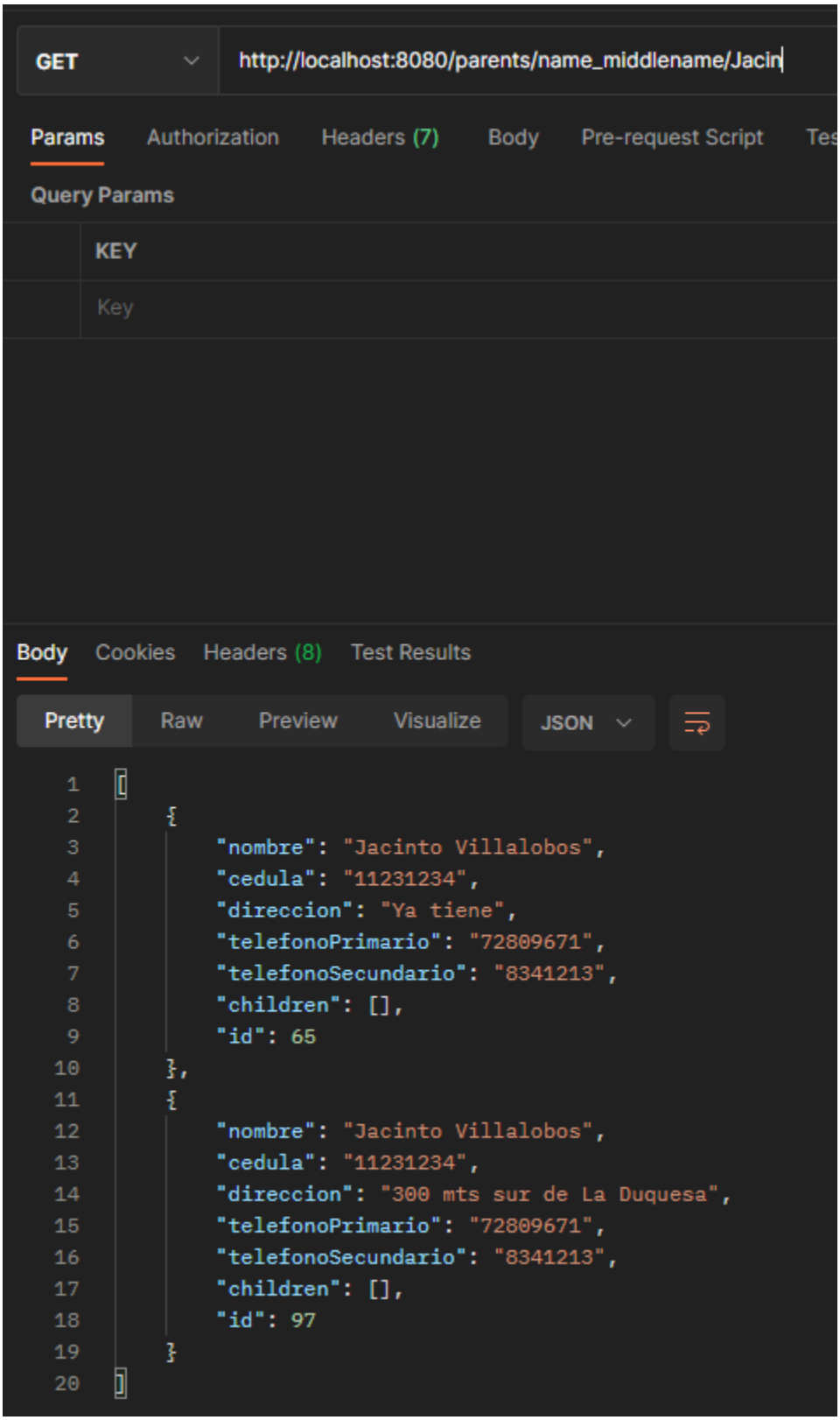
Registro de Padre/Madre de Familia: debe registrar su nombre, cedula, dirección, teléfono primario y telefono secundario. (5 PTS)

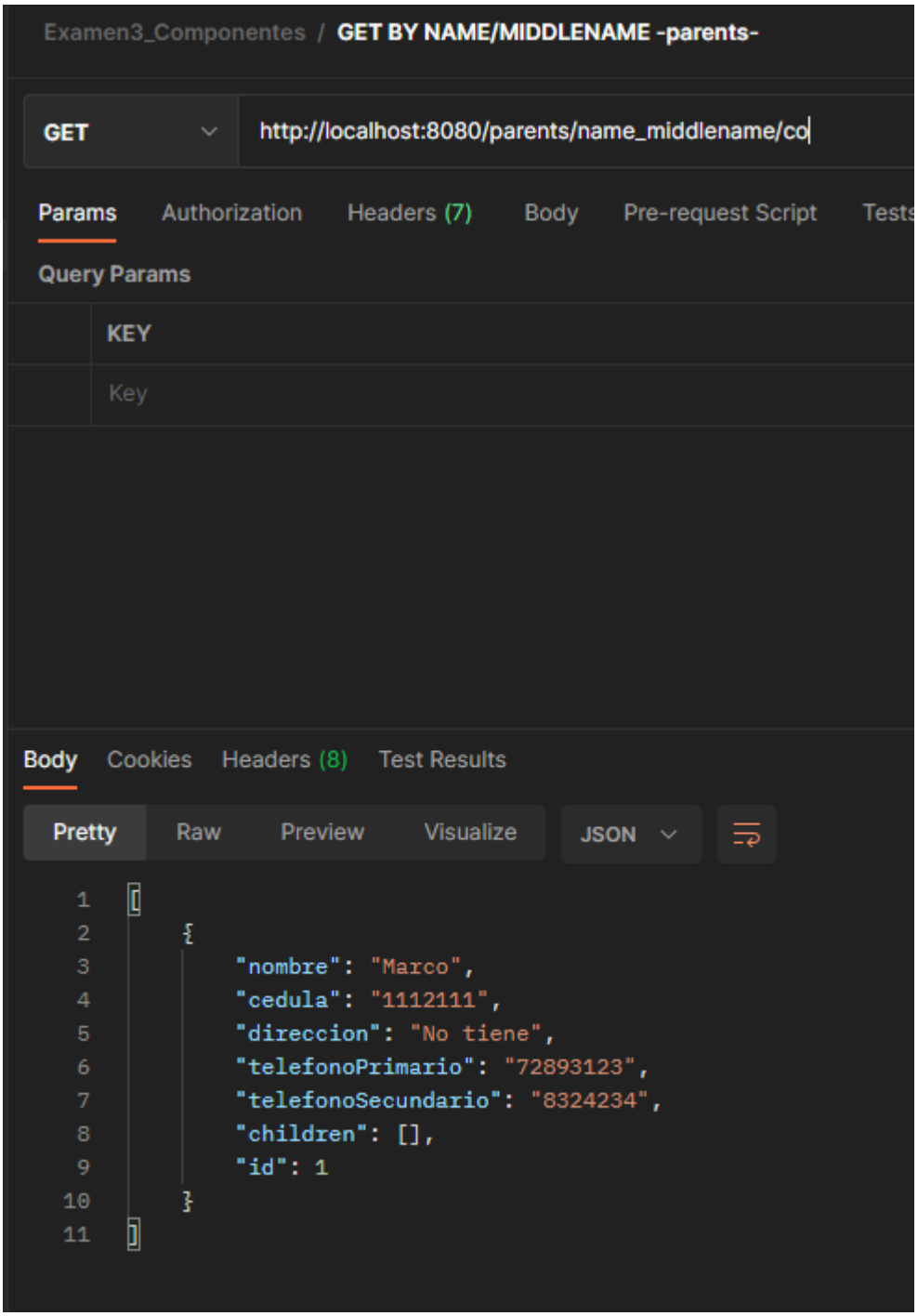


Debe hacer un endpoint para actualizar los datos de padre/madre (5pts)

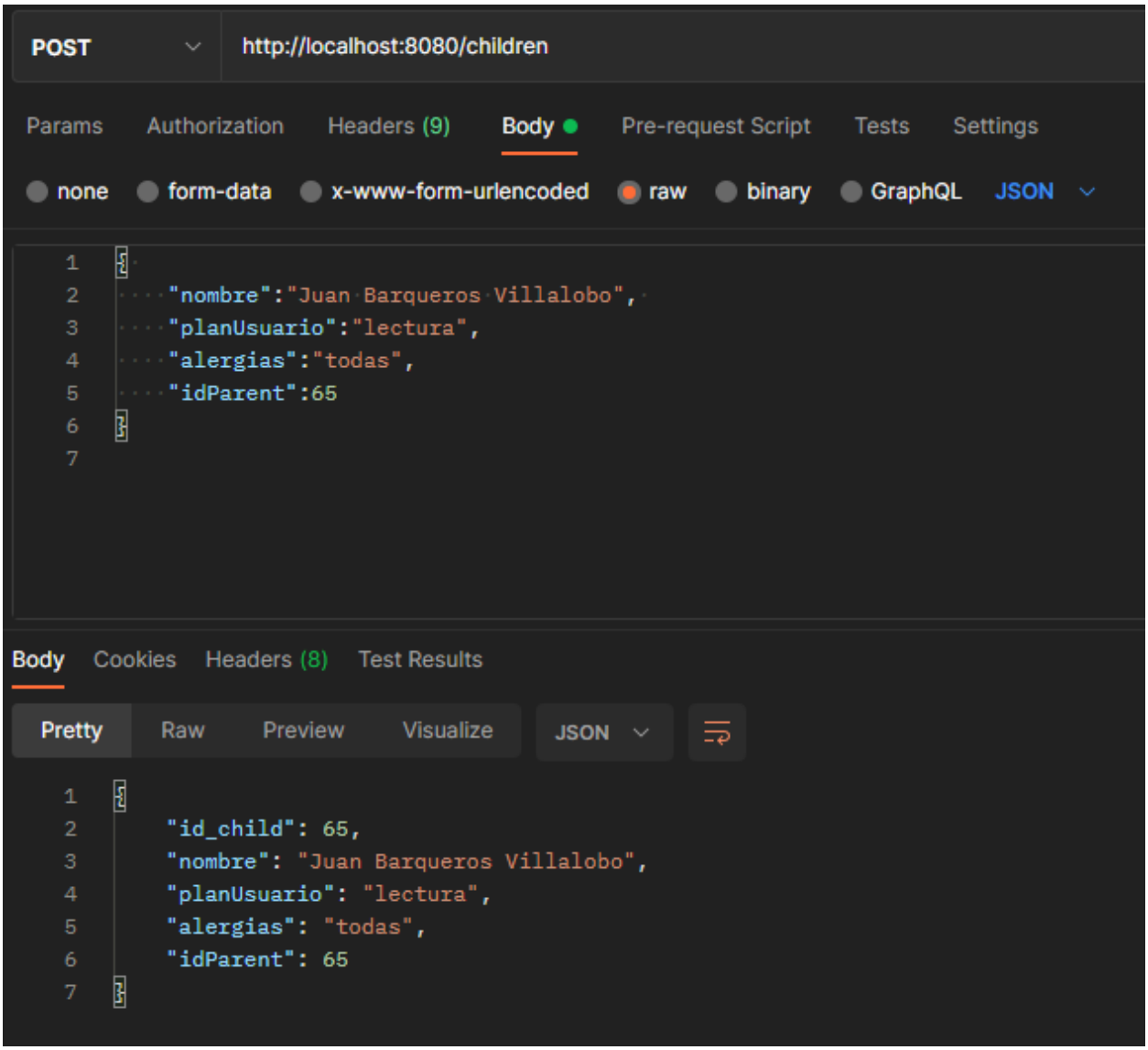


Otro endpoint para buscar los datos de un padre o madre por aproximación sobre el nombre y/o apellidos (5pts)

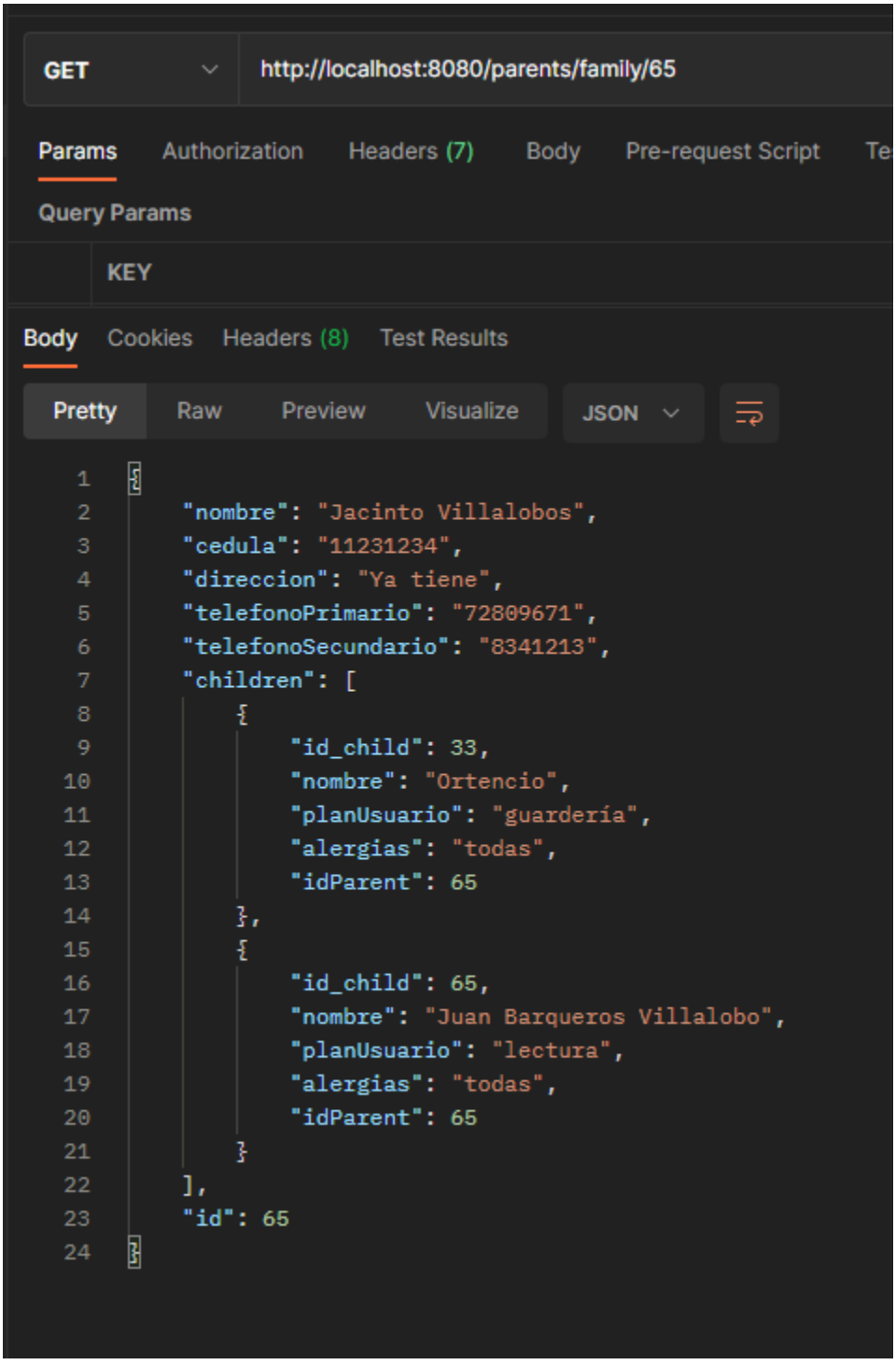



Examen3_Componentes / GET BY NAME/MIDDLENAME -parents-
GET
http://localhost:8080/parents/name_middlename/co
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Query Params
KEY
Key
Body
Cookies
Headers (8)
Test Results
Pretty
Raw
Preview
Visualize
JSON
[
{
"nombre": "Marco",
"cedula": "1112111",
"direccion": "No tiene",
"telefonoPrimario": "72893123",
"telefonoSecundario": "8324234",
"children": [],
"id": 1
}
]

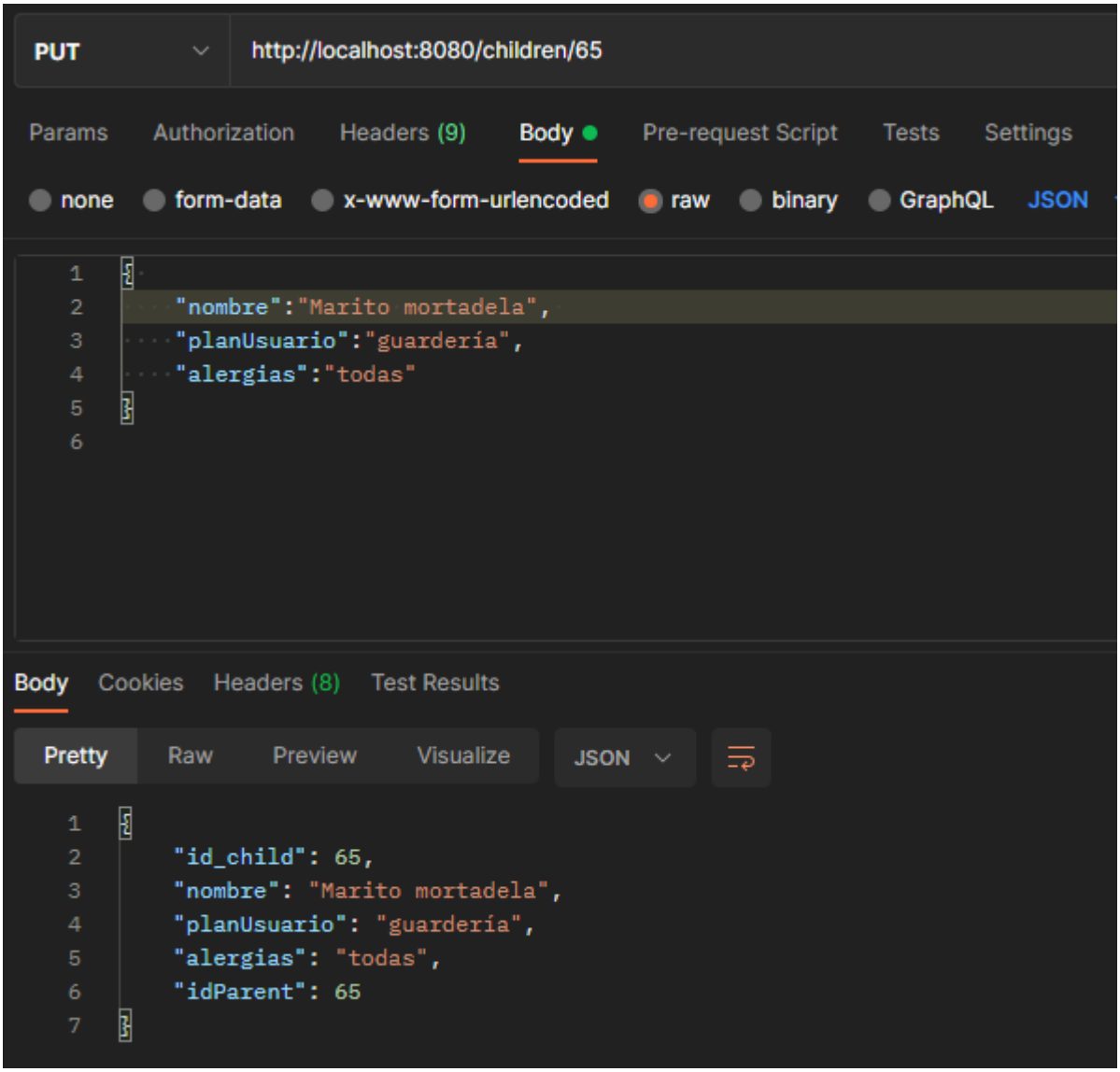
Registro de hijo/hija: debe registrar nombre, si es solo usuario de plan de lectura o de guardería o ambas y alergias (guárdelas como un solo texto) y asociarlos a sus padres. 10 pts



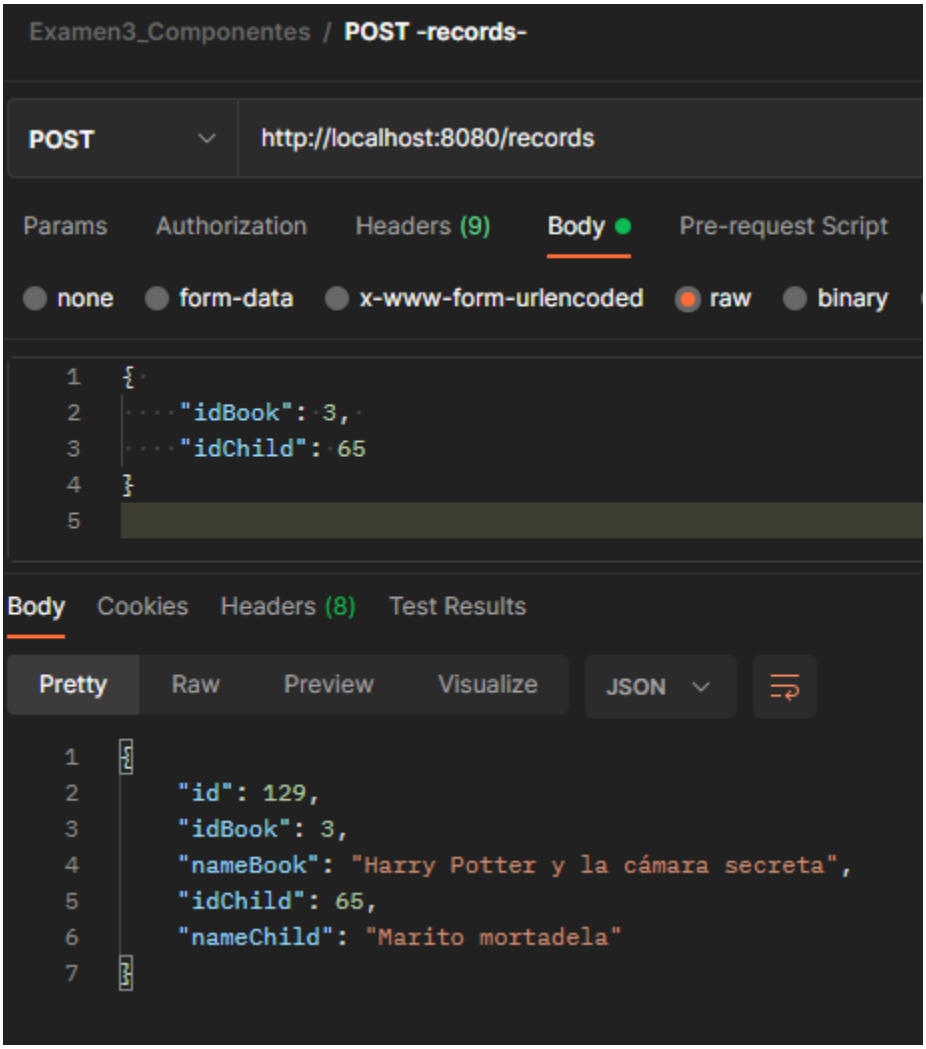
Debe hacer un endpoint que dado el id de un padre o madre liste sus datos y sus hijos o hijas 10 pts



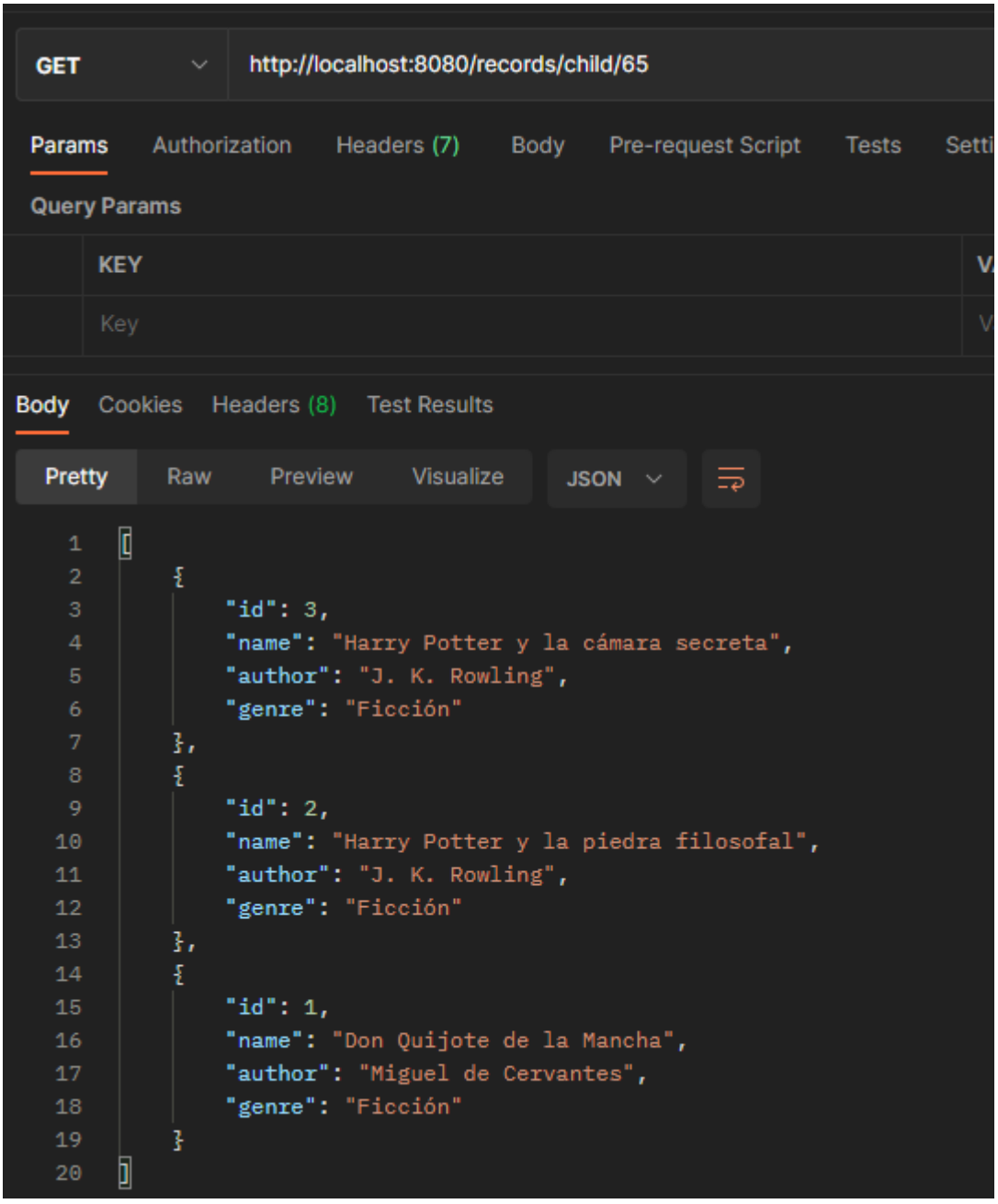
Debe hacer un endpoint para actualizar los datos del niño o niña 5 pts



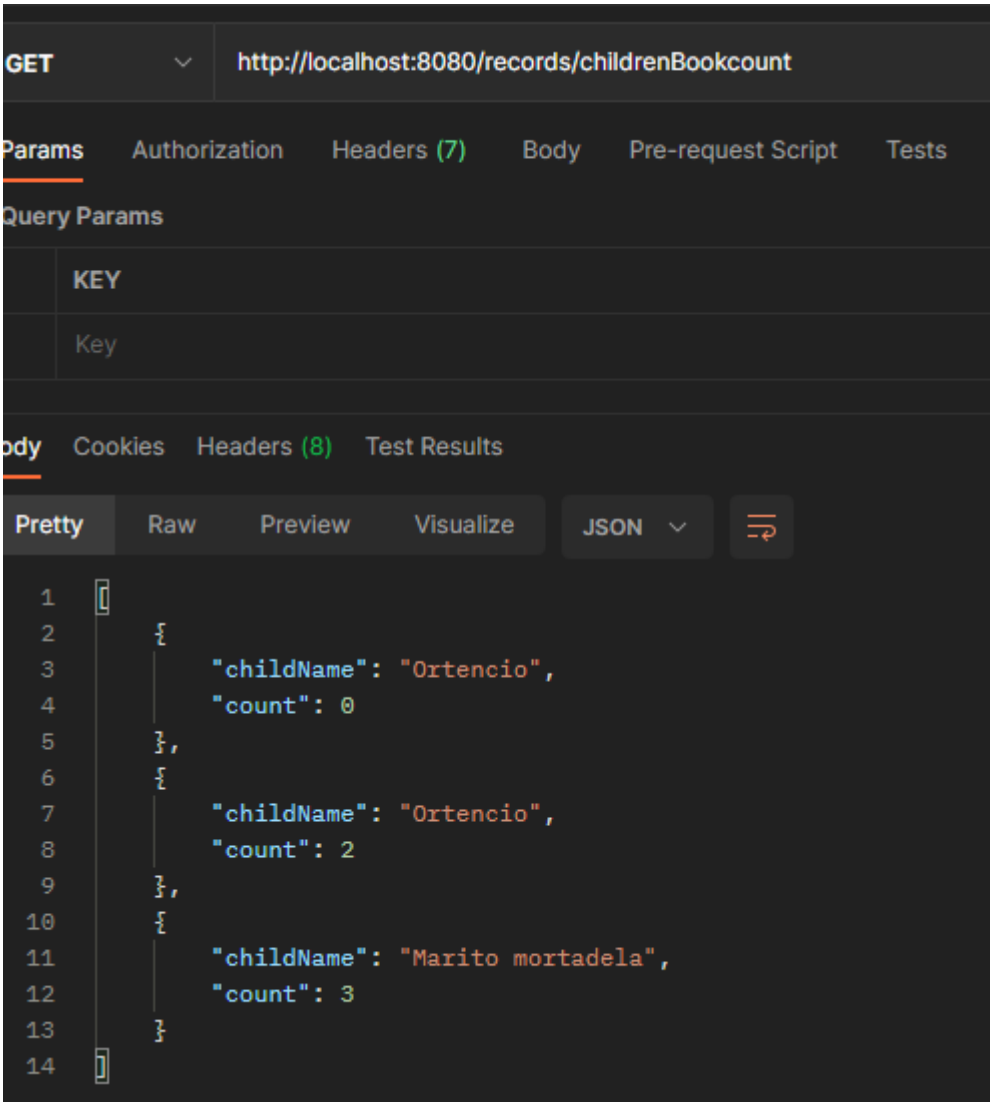
Debe registrar los libros que ha leído un niño 5pts



Debe hacer un endpoint que liste los libros que ha leído un niño 5pts



Debe hacer un endpoint que liste los nombres de los niños y cuantos libros ha leído 10pts



CON GRAPHQL

POST

```
mutation {
  createBook(name: "Harry potter", author: "J. K. K. Rowling", genre: "F
    id
  }
}
```

```
{
  "data": {
    "createBook": {
      "id": "33"
    }
  }
}
```

GET

```
{
  books(count: 10) {
    id
    nam
    aut
    genre
    status
  }
}
```

Query.books(count: Int)

```
{
  "data": {
    "books": [
      {
        "id": "1",
        "name": "banASADS",
        "author": "123queso",
        "genre": "toyota",
        "status": "activo"
      },
      {
        "id": "33",
        "name": "Harry potter",
        "author": "J. K. K. Rowling",
        "genre": "Ficción",
        "status": "activo"
      },
      {
        "id": "34",
        "name": "Harry potter 2",
        "author": "J. K. K. Rowling",
        "genre": "Ficción",
        "status": "activo"
      }
    ]
  }
}
```

GET BY ID

```
{
  book(id: 34) {
    id
    name
    author
    genre
    status
  }
}
```

```
{
  "data": {
    "book": {
      "id": "34",
      "name": "Harry potter 2",
      "author": "J. K. K. Rowling",
      "genre": "Ficción",
      "status": "activo"
    }
  }
}
```

PUT

```
mutation {
  updateBook(id: 34, name: "Don Quijote", author: "Miguel", genre: "Ficción", status: "activo") {
    id
  }
}
```

```
{
  "data": {
    "updateBook": {
      "id": "34"
    }
  }
}
```

```
{
  "data": {
    "book": {
      "id": "34",
      "name": "Don Quijote",
      "author": "Miguel",
      "genre": "Ficción",
      "status": "activo"
    }
  }
}
```

DELETE

```
mutation {
  deleteBook(id: 34) {
    id
  }
}
```

```
{
  "data": {
    "deleteBook": {
      "id": "34"
    }
  }
}
```

```
{
  "data": {
    "book": {
      "id": "34",
      "name": "Don Quijote",
      "author": "Miguel",
      "genre": "Ficción",
      "status": "inactivo"
    }
  }
}
```