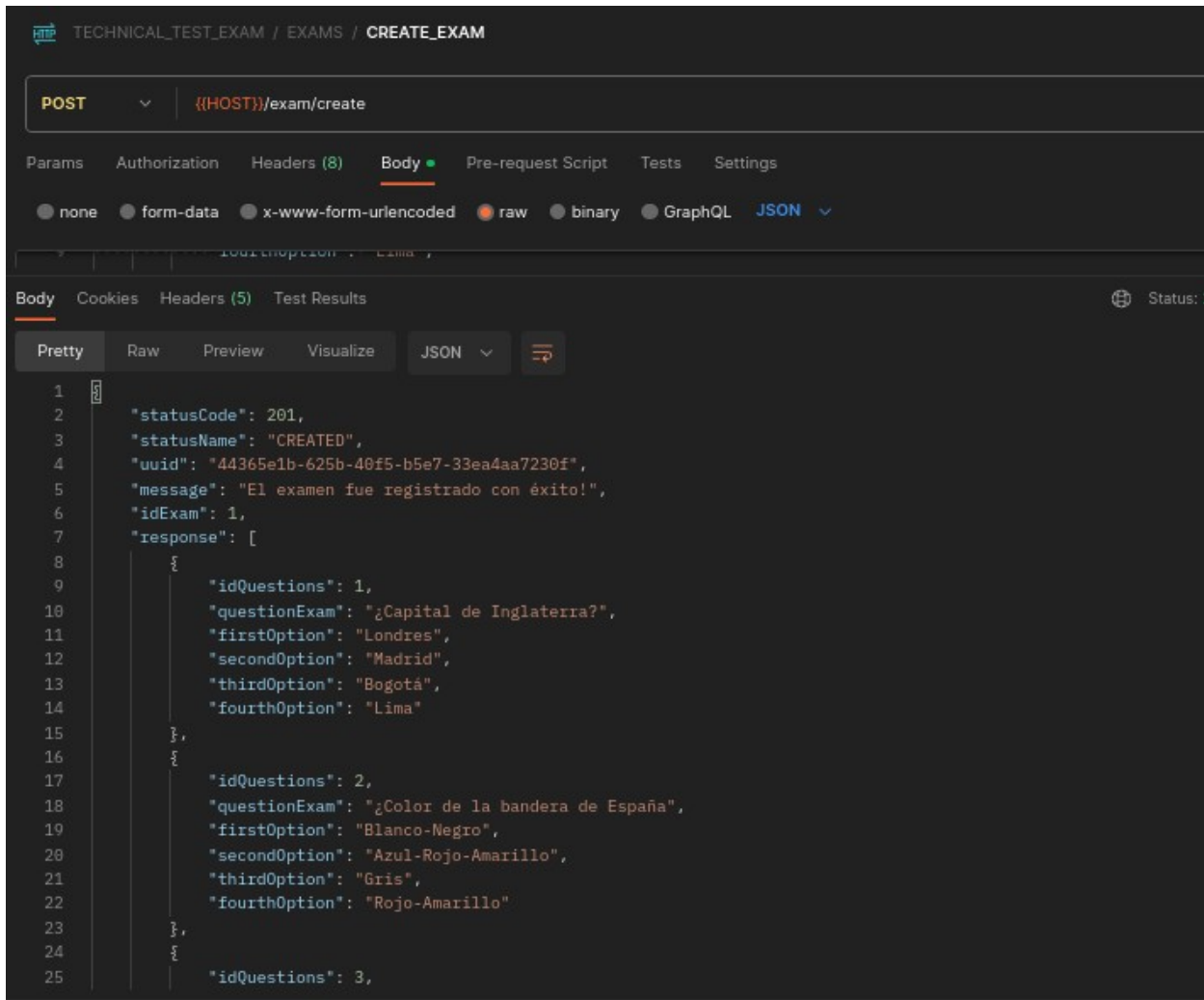


Documento

Este componente tiene como finalidad realizar los registros de exámenes, estudiantes, respuestas del examen, y recuperación de las calificaciones.

1. Se crea el examen, este tiene varias restricciones (ningún campo debe ser nulo o vacío), adicionalmente, cada examen debe tener 3 preguntas.

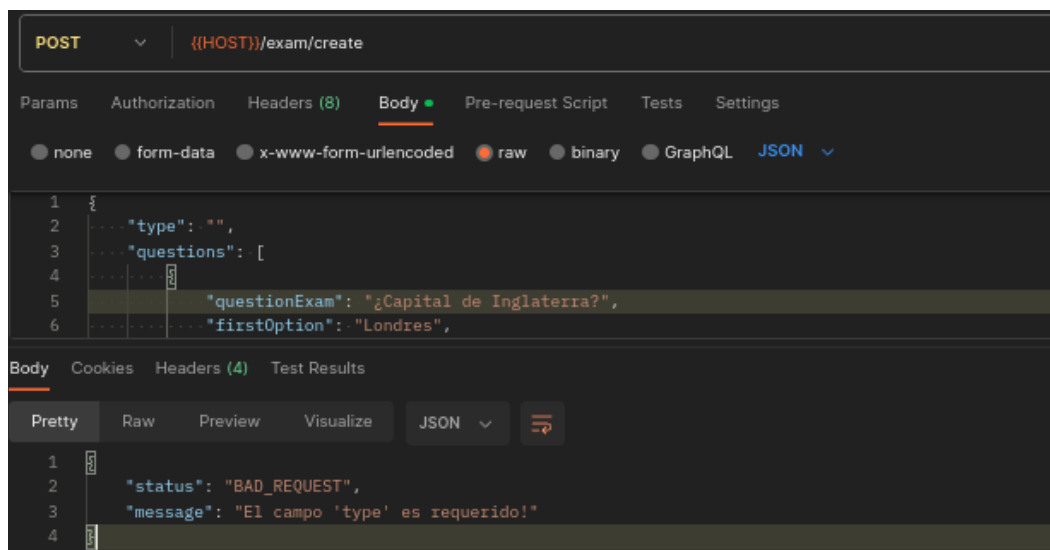
Respuestas:



The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** `{{HOST}}/exam/create`
- Body Type:** JSON
- Body Content (Pretty):**

```
1 {
2   "statusCode": 201,
3   "statusName": "CREATED",
4   "uuid": "44365e1b-625b-40f5-b5e7-33ea4aa7230f",
5   "message": "El examen fue registrado con éxito!",
6   "idExam": 1,
7   "response": [
8     {
9       "idQuestions": 1,
10      "questionExam": "¿Capital de Inglaterra?",
11      "firstOption": "Londres",
12      "secondOption": "Madrid",
13      "thirdOption": "Bogotá",
14      "fourthOption": "Lima"
15    },
16    {
17      "idQuestions": 2,
18      "questionExam": "¿Color de la bandera de España",
19      "firstOption": "Blanco-Negro",
20      "secondOption": "Azul-Rojo-Amarillo",
21      "thirdOption": "Gris",
22      "fourthOption": "Rojo-Amarillo"
23    },
24    {
25      "idQuestions": 3,
```



The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** `{{HOST}}/exam/create`
- Body Type:** JSON
- Body Content (Raw):**

```
1 {
2   ... "type": "",
3   ... "questions": [
4     ... {
5       "questionExam": "¿Capital de Inglaterra?",
6       ... "firstOption": "Londres",
```
- Body Content (Pretty):**

```
1 {
2   "status": "BAD_REQUEST",
3   "message": "El campo 'type' es requerido!"
4 }
```

2. Se pasa a realizar la inserción de un estudiante con los campos solicitados (nombre, edad, ciudad y zona horaria).

Tiene restricciones con los campos anteriormente mencionados, además, también se valida que el campo zona horaria sea un válido.

Respuestas:

```
POST {{HOST}}/student/create

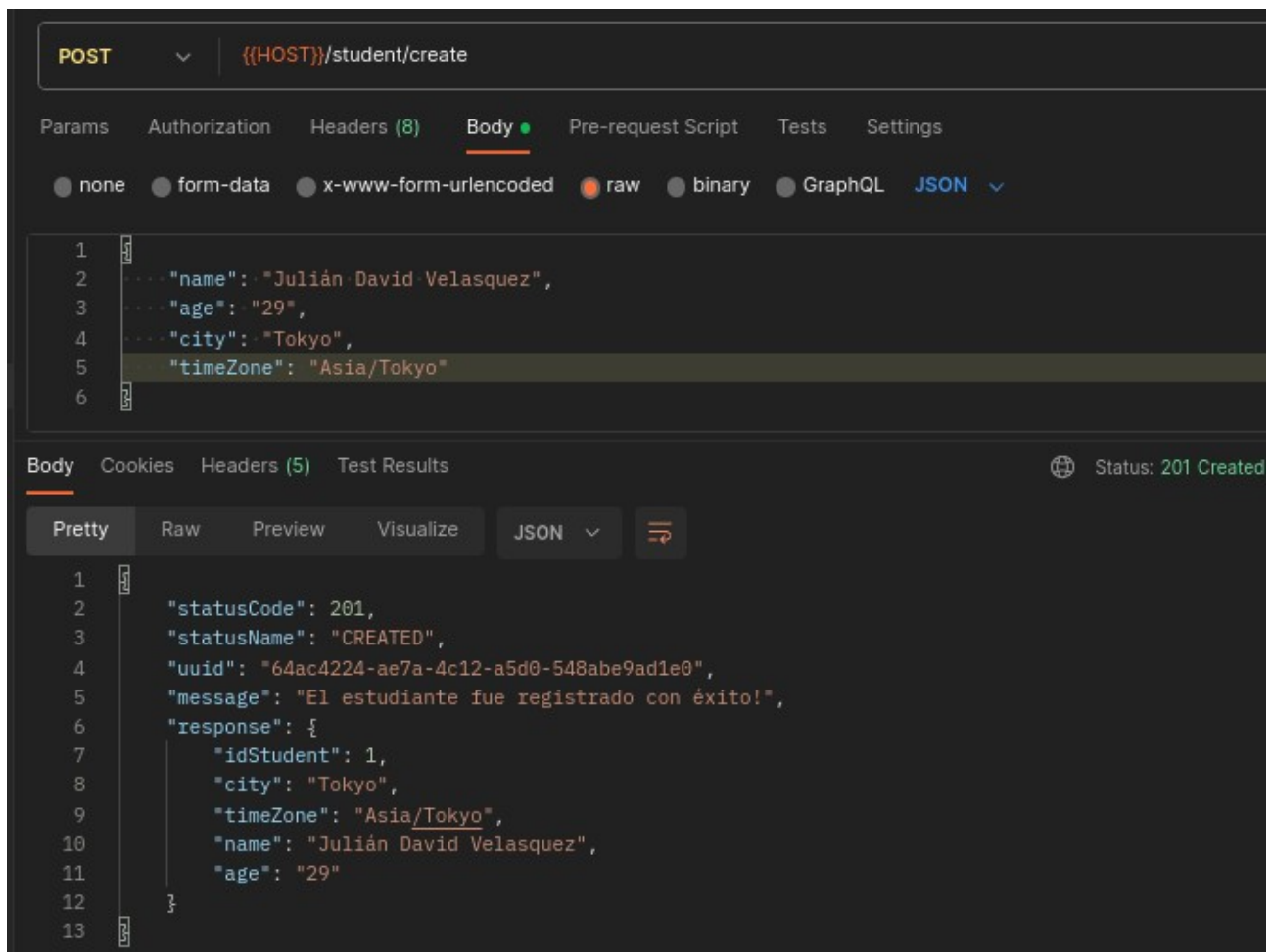
{
  "name": "Julían Velasquez",
  "age": "29",
  "city": "Soacha",
  "timeZone": "Pandequeso"
}
```

```
{
  "status": "BAD_REQUEST",
  "message": "El timezone ingresado es inválido!"
}
```

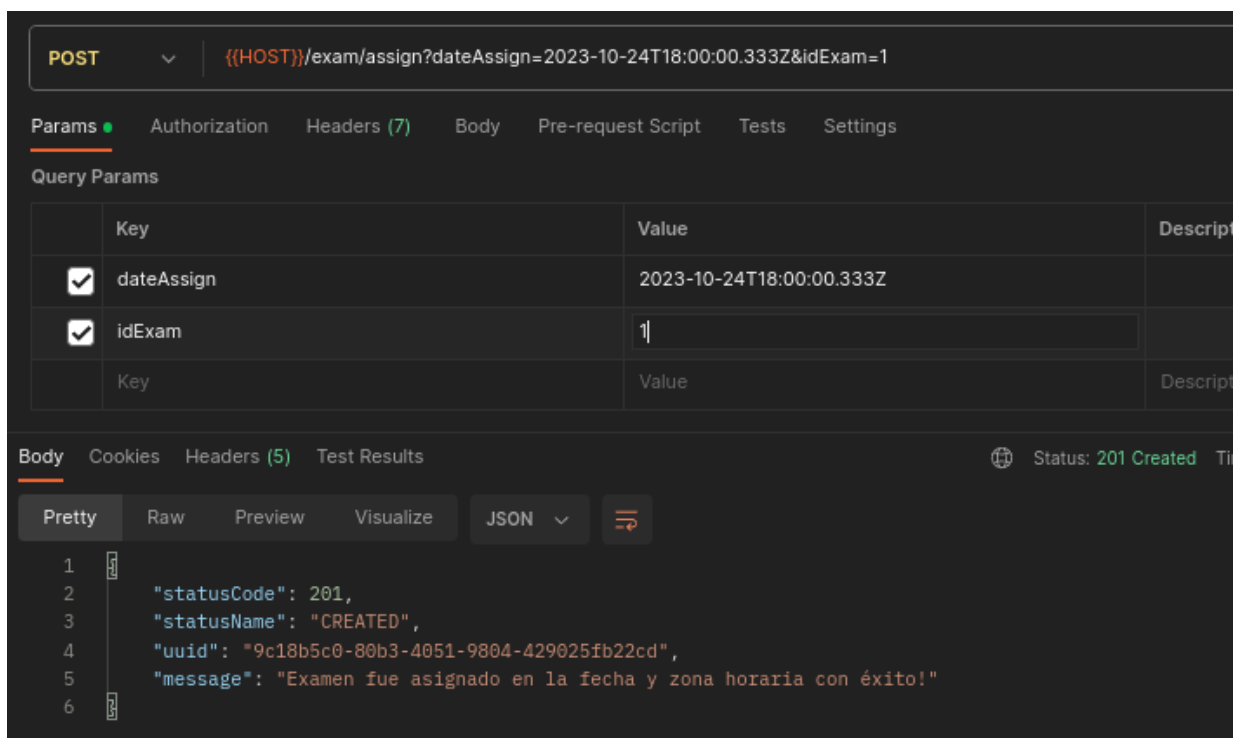
```
POST {{HOST}}/student/create

{
  "age": "29",
  "city": "Soacha",
  "timeZone": "Pandequeso"
}
```

```
{
  "status": "BAD_REQUEST",
  "message": "El campo 'name' es requerido!"
}
```



3. Luego de tener el estudiante y el examen ya creados, se procede a asignar el examen al estudiante, para hacer esto se debe pasar la fecha en formato (2023-10-24T18:00:00.333Z) y el id del examen.



GET

{{HOST}}/student/all

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

	Key	Value
	Key	Value

Body Cookies Headers (5) Test Results

Pretty

Raw

Preview

Visualize

JSON



```

1  {
2    "statusCode": 200,
3    "statusName": "OK",
4    "uuid": "d68bf651-36d3-47e4-bcab-3d6d53a2c044",
5    "message": "Respuesta exitosa!",
6    "response": [
7      {
8        "name": "Julián David Velasquez",
9        "idStudent": 1,
10       "city": "Tokyo",
11       "dateAssign": "2023-10-25 03:00:00.333",
12       "area": "Geografía",
13       "statusExam": "ACTIVE"
14     }
15   ]
16 }
```

POST

{{HOST}}/exam/assign?dateAssign=2023-10-24&idExam=1

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

	Key	Value	Description
<input checked="" type="checkbox"/>	dateAssign	2023-10-24	
<input checked="" type="checkbox"/>	idExam	1	
	Key	Value	Description

Body Cookies Headers (4) Test Results

Status: 400 Bad Request Time

Pretty

Raw

Preview

Visualize

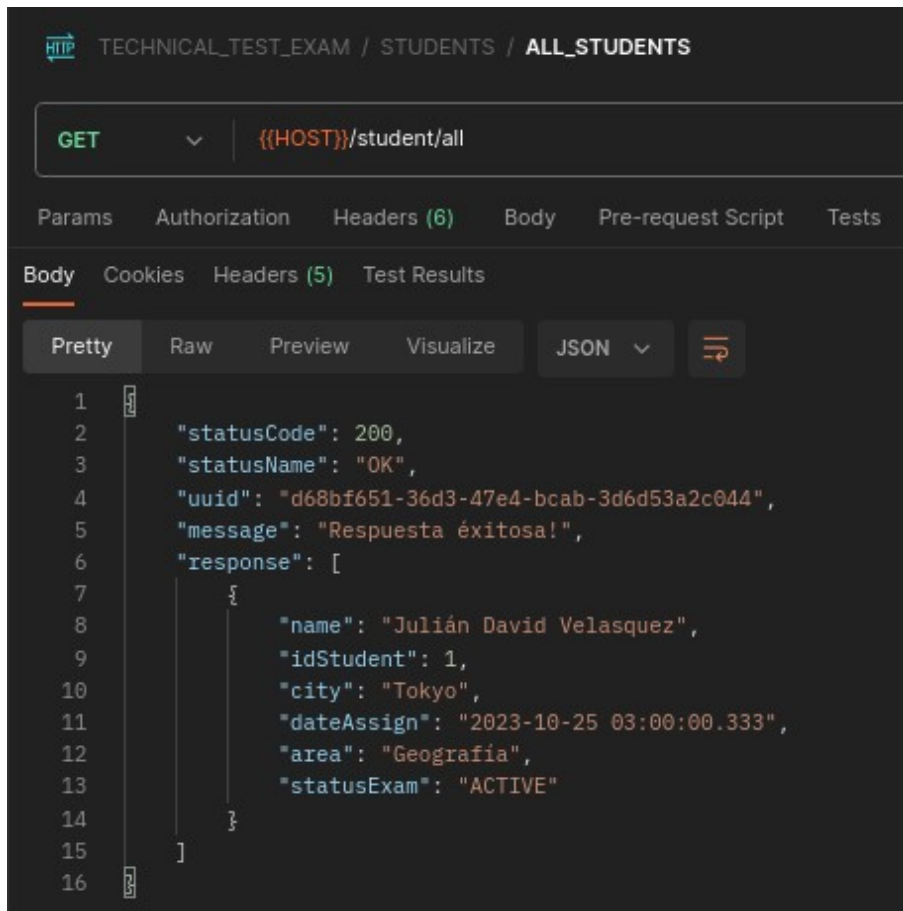
JSON



```

1  {
2    "status": "BAD_REQUEST",
3    "message": "La fecha ingresada tiene un patrón invalido! :: Valid -> 2023-01-01T00:00:00.333Z!"
4  }
```

4. Después de asignar un examen podemos consultar los estudiantes con examen asignado



```
HTTP TECHNICAL_TEST_EXAM / STUDENTS / ALL_STUDENTS

GET {{HOST}}/student/all

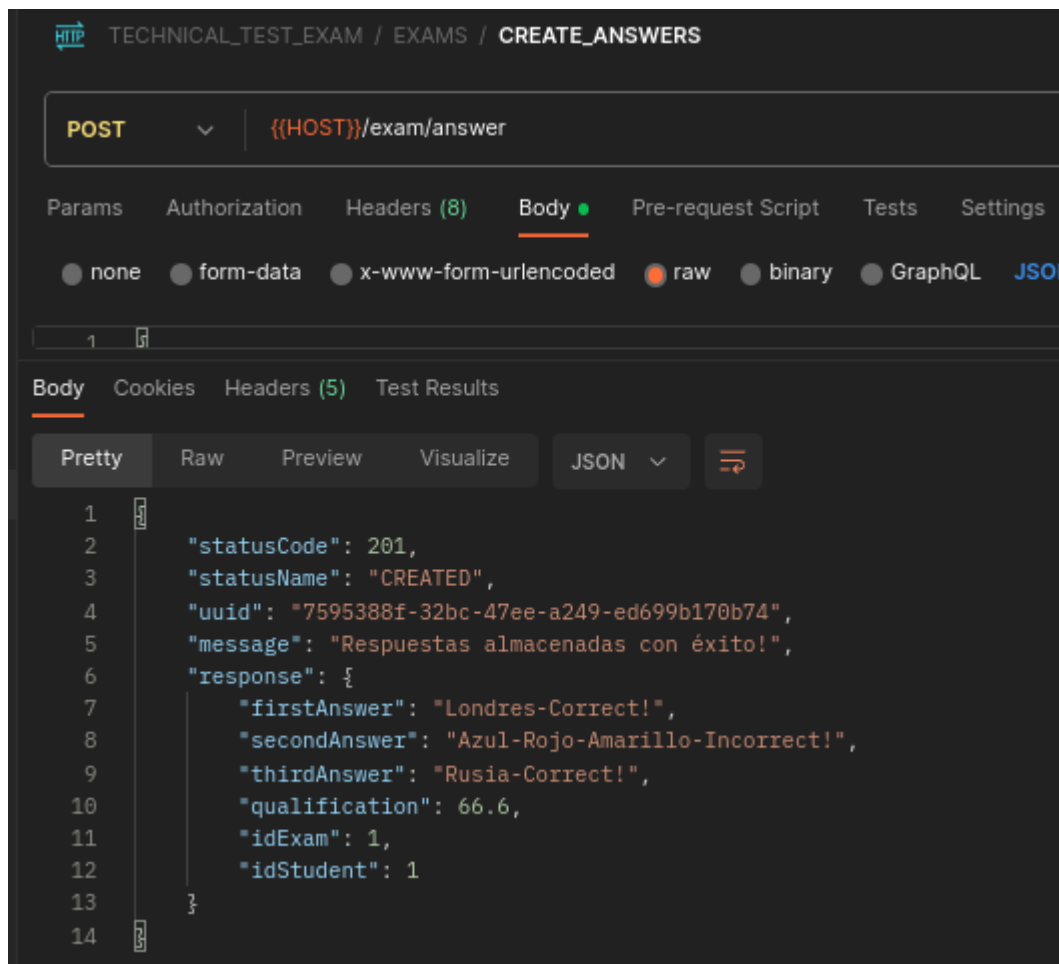
Params Authorization Headers (6) Body Pre-request Script Tests

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

1
2   "statusCode": 200,
3   "statusName": "OK",
4   "uuid": "d68bf651-36d3-47e4-bcab-3d6d53a2c044",
5   "message": "Respuesta exitosa!",
6   "response": [
7     {
8       "name": "Julián David Velasquez",
9       "idStudent": 1,
10      "city": "Tokyo",
11      "dateAssign": "2023-10-25 03:00:00.333",
12      "area": "Geografia",
13      "statusExam": "ACTIVE"
14    }
15  ]
16
```

5. Luego se envían las respuestas del examen indicando (idStudent, idExam).



```
HTTP TECHNICAL_TEST_EXAM / EXAMS / CREATE_ANSWERS

POST {{HOST}}/exam/answer

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

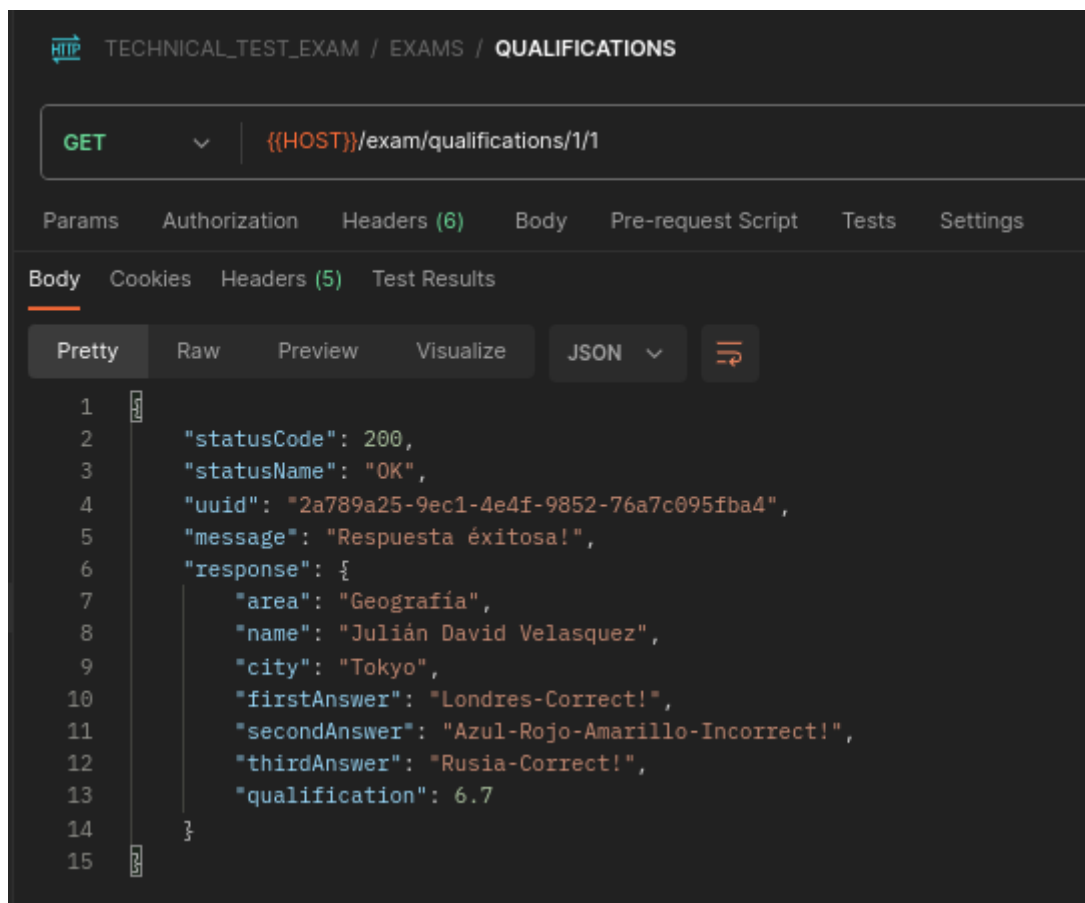
1

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

1
2   "statusCode": 201,
3   "statusName": "CREATED",
4   "uuid": "7595388f-32bc-47ee-a249-ed699b170b74",
5   "message": "Respuestas almacenadas con éxito!",
6   "response": {
7     "firstAnswer": "Londres-Correct!",
8     "secondAnswer": "Azul-Rojo-Amarillo-Incorrect!",
9     "thirdAnswer": "Rusia-Correct!",
10    "qualification": 66.6,
11    "idExam": 1,
12    "idStudent": 1
13  }
14
```

6. Se consulta las respuestas del examen, pasando (idStudent, idExam), pero por variable url



The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `{{HOST}}/exam/qualifications/1/1`
- Response Status:** 200 OK
- Response Body (JSON):**

```
{  "statusCode": 200,  "statusName": "OK",  "uuid": "2a789a25-9ec1-4e4f-9852-76a7c095fba4",  "message": "Respuesta exitosa!",  "response": {    "area": "Geografía",    "name": "Julián David Velasquez",    "city": "Tokyo",    "firstAnswer": "Londres-Correct!",    "secondAnswer": "Azul-Rojo-Amarillo-Incorrect!",    "thirdAnswer": "Rusia-Correct!",    "qualification": 6.7  }}
```