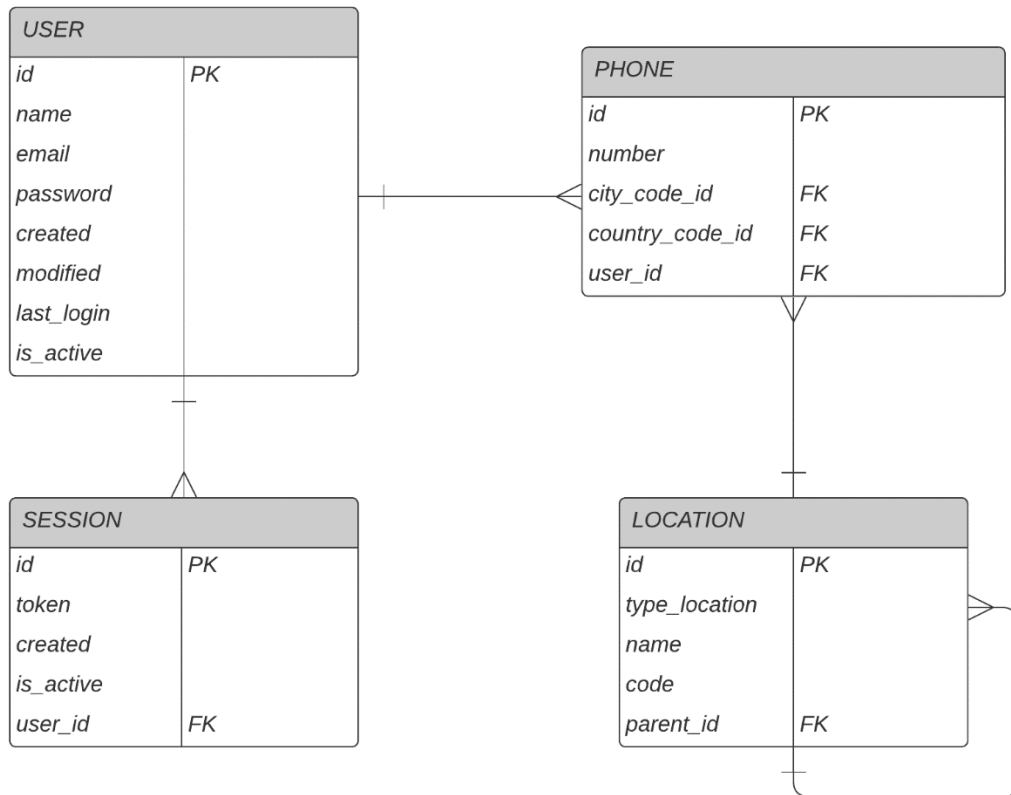


# Solución Evaluación: JAVA

## Diagrama de la solución

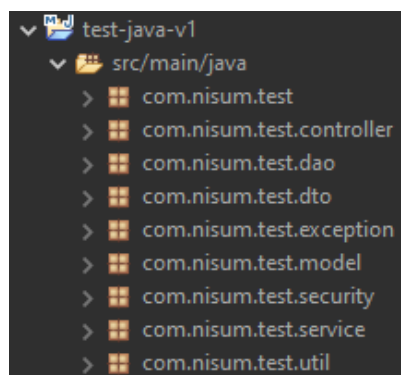
Con base en el problema descrito se identificaron cuatro entidades y sus relaciones y se elaboró el diagrama entidad relación presenta a continuación.



**Figura 1. Diagrama Entidad Relación**

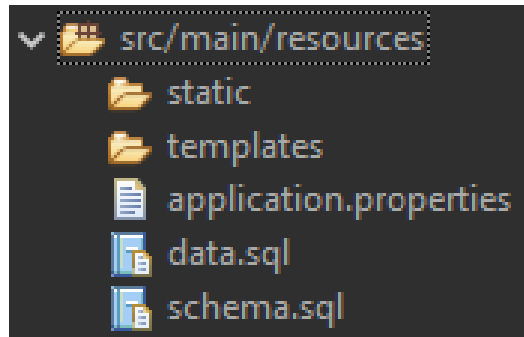
## Arquitectura

Para el desarrollo de la solución se implementó el patrón arquitectural multicapa, en el cual se definieron las siguientes capas.



**Figura 2. Capas creadas en el proyecto**

Como recursos adicionales se utilizo la base de datos H2, como fue solicitado y se crearon los archivos de **schema.sql** y **data.sql** donde se encuentra los scripts de creación de las tablas y de inserción de datos iniciales respectivamente.



**Figura 3. Archivos para creación de tablas y cargue de datos iniciales**

## Tecnologías Utilizadas

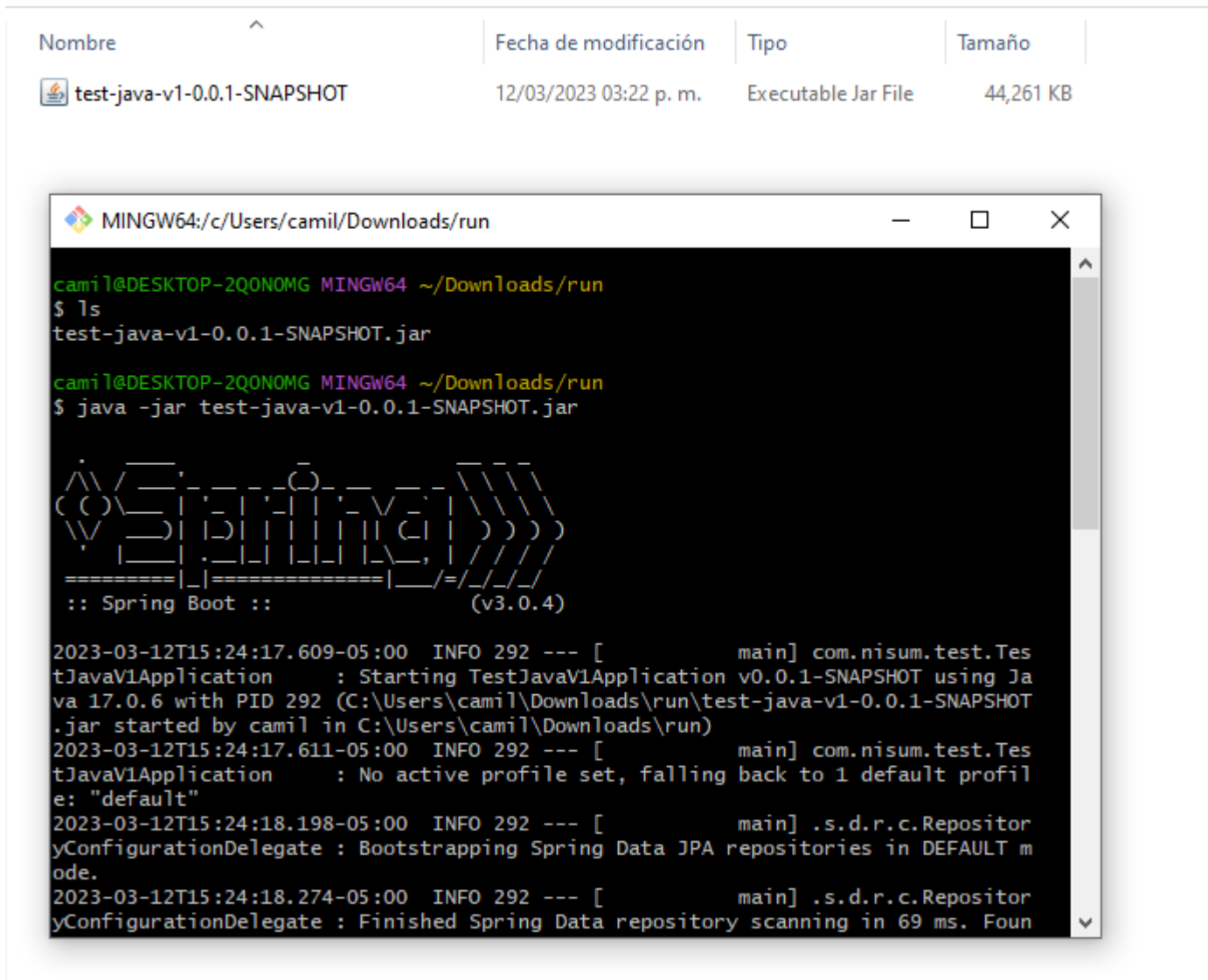
Para el desarrollo de la solución se utilizaron las siguientes tecnologías:

- Java 17.0.6
- Spring Boot 3.0.4
- Base de datos H2
- Maven
- Hibernate
- JWT
- Junit
- Postman

## PRUEBA

Para probar la aplicación realizada se debe contar con java 17. En la carpeta donde se almacene el archivo que se adjunta test-java-v1-0.0.1-SNAPSHOT.jar. Por La línea de comandos ejecutar el comando:

```
java -jar test-java-v1-0.0.1-SNAPSHOT
```



**Figura 4. Ejecución de la aplicación**

Una vez iniciada la aplicación, con ayuda de un sistema de pruebas (**Postman**) se pueden realizar las respectivas pruebas al siguiente endpoint.

<http://localhost:8080/register>

Dado que los servicios fueron asegurados se debe utilizar el siguiente token inicial:

eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJjYXV1bG8uZGVsZ2Fkb2NAZ21haWwY29tIiwiaXNjbG9jaXQzMzI2MDQwODQ4MjYxLCJuYXZ1IjoidGFyNTY0UE9zc2QqIn0.V-F-QgwsveZkygI4UWmlIJ5XCYC6QEiqT6R6v7lzfQVM



# Pruebas de validación

## 1. No ingreso de nombre de usuario

Se realiza la prueba sin nombre de usuario. Como se aprecia en el mensaje de error y el código de mensaje retornado, cumplen con lo solicitado en la prueba.

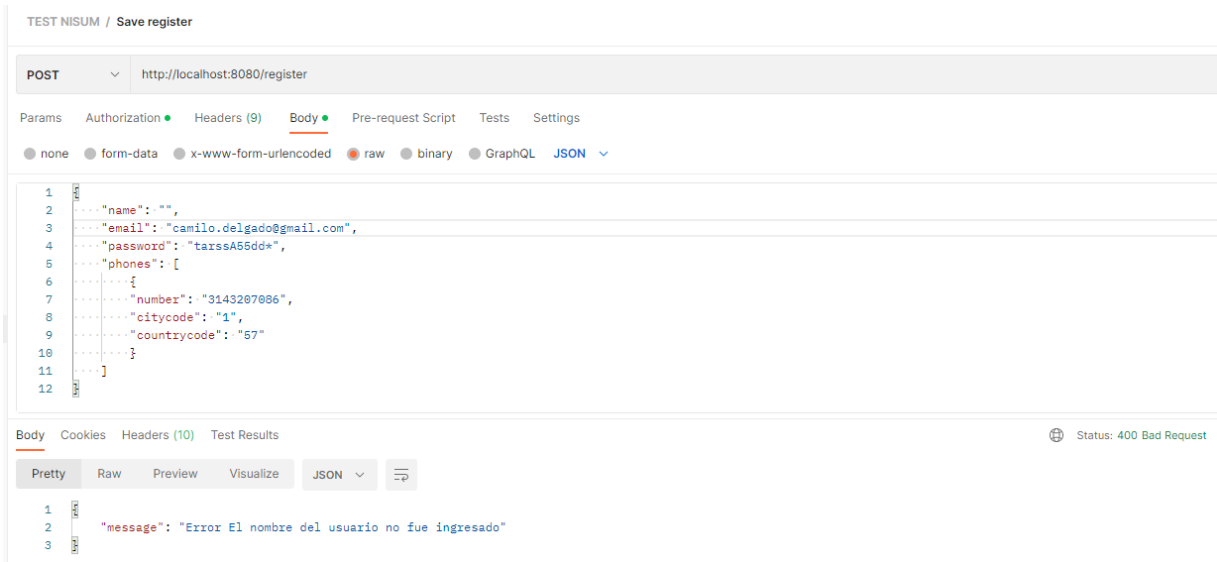


Figura 7. Prueba 2. Validación del ingreso del nombre del usuario

## 2. Si se trata de ingresar con un correo que ya está registrado

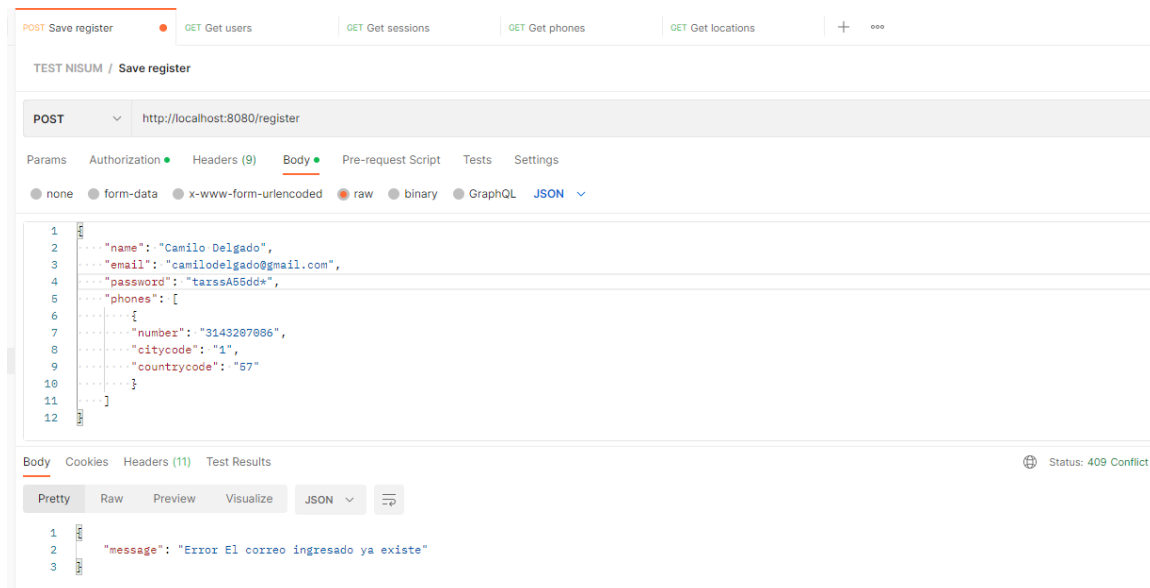


Figura 8. Prueba 3. Validación del correo existente

### 3. Validación formato del correo

#### Reglas

- Valores numéricos de 0 a 9.
- Mayúsculas y minúsculas de la **(a)** a la **(z)**
- Permite guion bajo, "\_", guion "-", y punto "."
- No puede iniciar ni terminar con punto (.)
- No permite puntos (..) consecutivos
- Tamaño máximo de 64 caracteres.

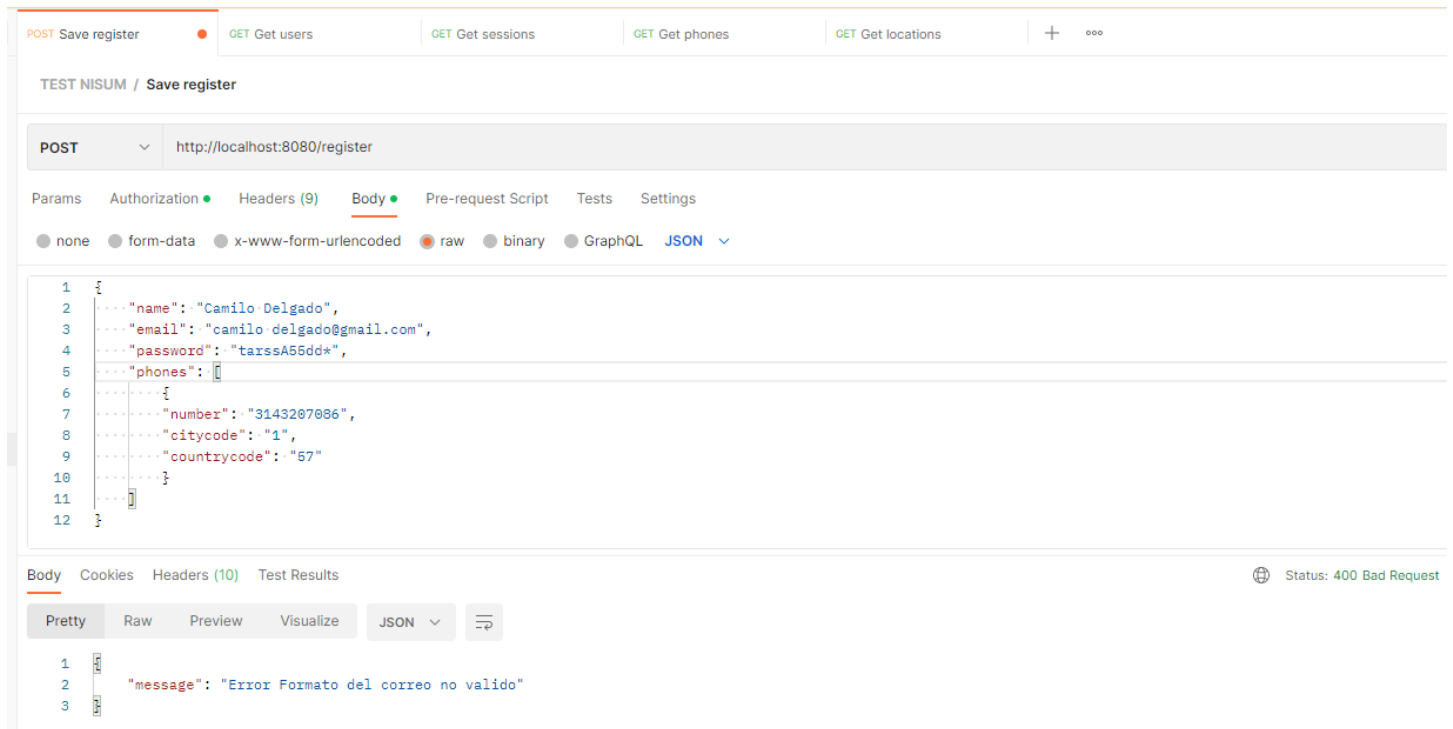


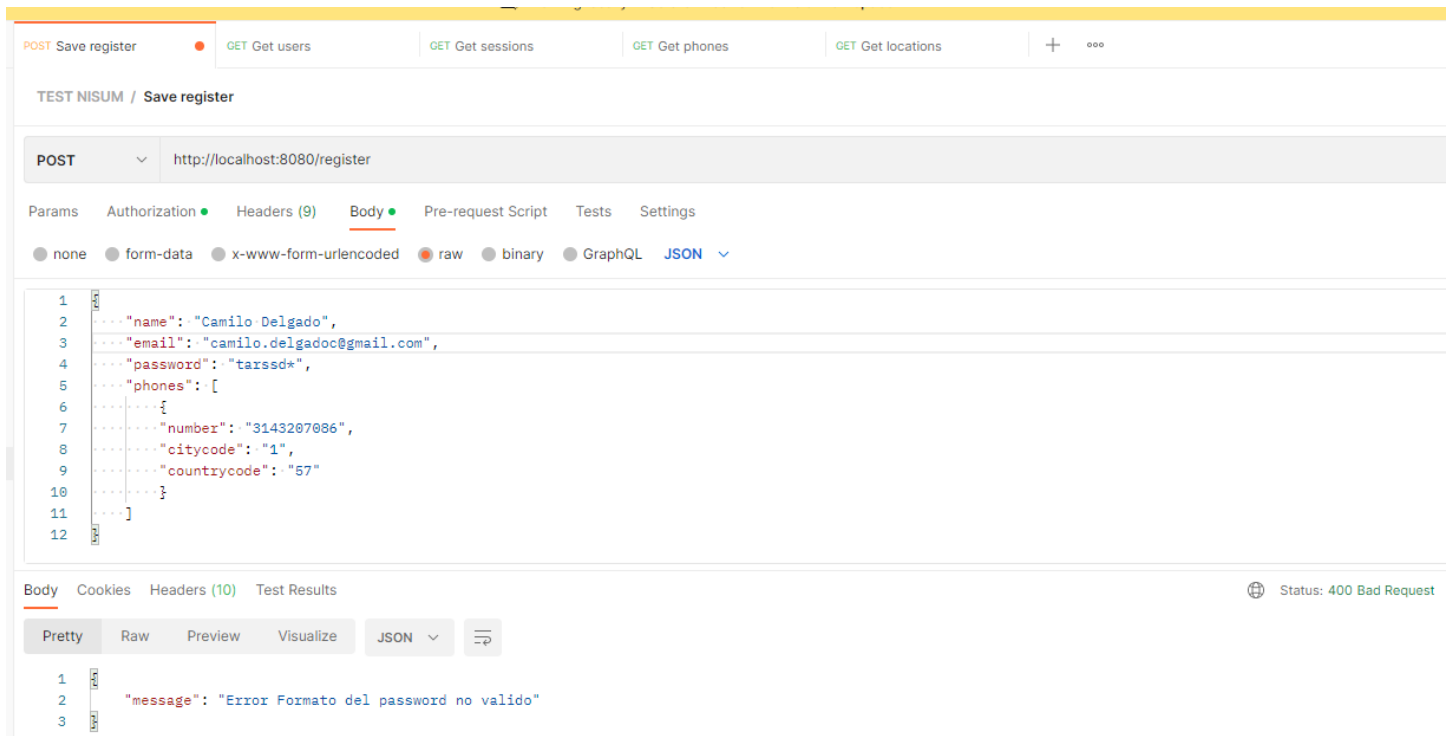
Figura 9. Prueba 4. Validación del formato de correo espacio en blanco

### 4. Validación formato de la contraseña

#### Reglas

El Password debe contener:

- Por lo menos un dígito [0-9].
- Por lo menos un carácter latino en minúscula [a-z].
- Por lo menos un carácter latino en Mayúscula [A-Z].
- Por lo menos un carácter especial como ! @ # & ( ).
- Longitud de entre 8 a máximo 20 caracteres.



**Figura 10. Prueba 5. Validación del formato de la contraseña falta número y letra mayúscula**

## 5. Validación de los códigos de área

Para la prueba se registraron unos códigos de área iniciales que se pueden consultar en el archivo **data.sql**

### Países

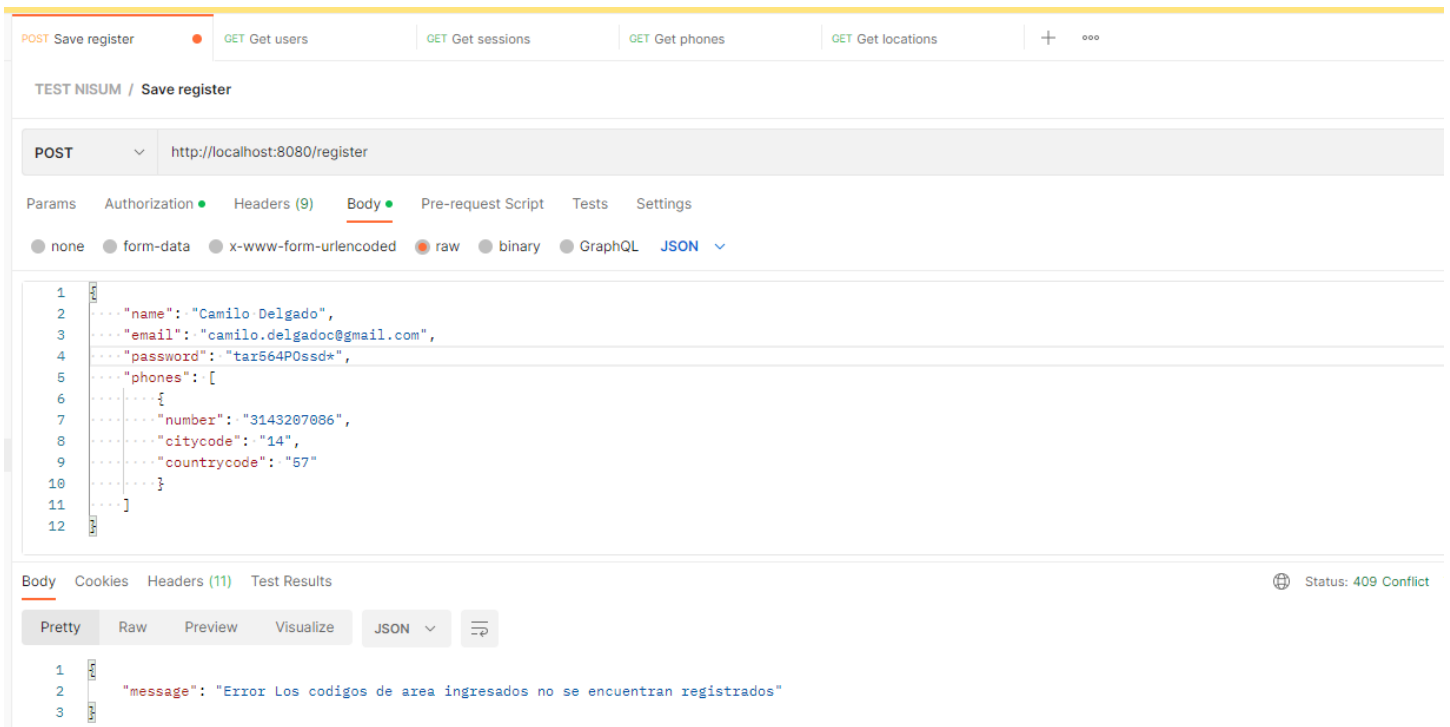
- 'Perú', '51'
- 'Argentina', '54'
- 'Brasil', '55'
- 'Chile', '56'
- 'Colombia', '57'

### Ciudades de Colombia

- 'Bogotá', '1'
- 'Cali', '2',
- 'Medellín', '4',
- 'Barranquilla', '5'
- 'Bucaramanga', '7'

Para la prueba solo se parametrizaron las 5 ciudades de Colombia, cualquier otro valor distinto de 57 + 1 o 2 o 4 o 5 o 7

Producirá un error.

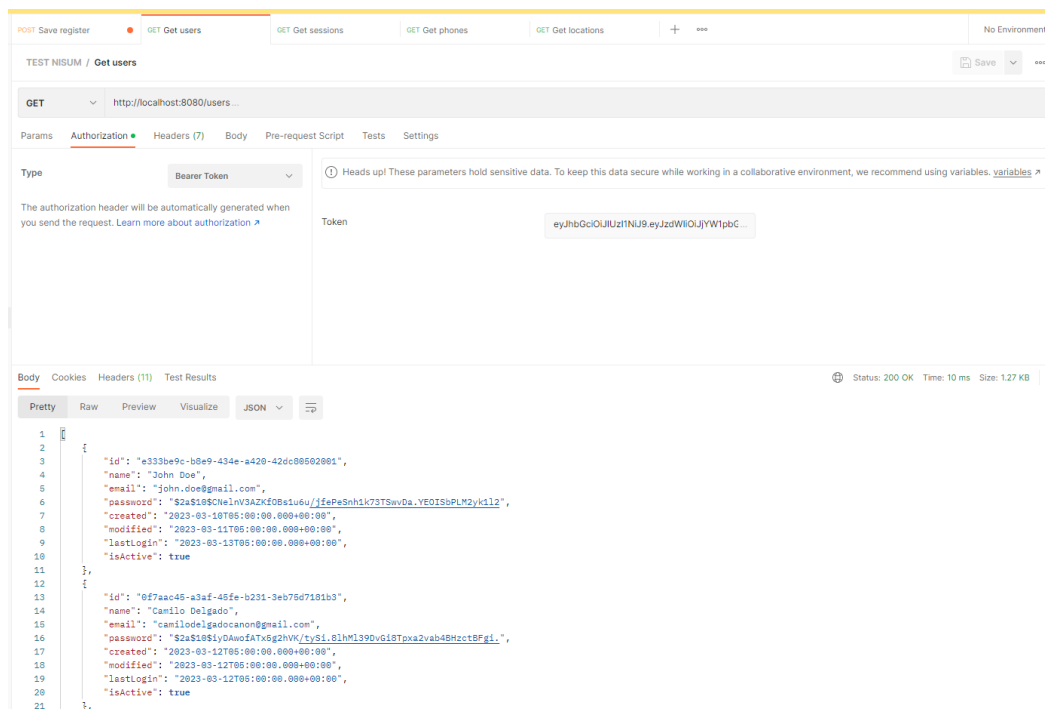


**Figura 11. Prueba 6. Validación del formato de la contraseña falta número y letra mayúscula**

Adicionalmente como parte extra a la prueba, se crearon los siguientes endpoint con los cuales se pueden validar que los datos sean registrados en las tablas, para consultar estos endpoints se debe usar el token adjunto o se puede usar el token que genera la prueba de ingreso del usuario

## Consulta de todos los usuarios

<http://localhost:8080/users>





# Consulta de todas las sesiones (tabla donde se almacenan los tokens generados)

<http://localhost:8080/sessions>

POST Save registerGET Get usersGET Get sessionsGET Get phonesGET Get locations+ ...No Environment

TEST NISUM / Get sessions

GEThttp://localhost:8080/sessionsSend

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettingsCookies

TypeBearer Token

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. variables

The authorization header will be automatically generated when you send the request. Learn more about authorization

TokeneyJhbGciOiJIUzI1Ni99.eyJzdWI0IjY1Y1pbC...

BodyCookiesHeaders (11)Test Results

Status: 200 OKTime: 14 msSize: 2.11 KBSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "id": "a333be9c-b8e9-434e-c420-42dc08084001",
3   "token": "eyJhbGciOiJIUzI1Ni99.eyJzdWI0IjY1Y1pbC...",
4   "created": "2023-03-10T05:00:00.000+00:00",
5   "isActive": true,
6   "user": {
7     "id": "a333be9c-b8e9-434e-c420-42dc08082001",
8     "name": "John Doe",
9     "email": "john.doe@gmail.com",
10    "password": "$2a$10$CwlnV3AZXf08s1u6u/ffePeShtk73T8wDa.YE01SbPLM2yk112",
11    "created": "2023-03-10T05:00:00.000+00:00",
12    "modified": "2023-03-11T05:00:00.000+00:00",
13    "lastLogin": "2023-03-13T05:00:00.000+00:00",
14    "isActive": true
15  },
16 },
17 },
18 },
19 {
20   "id": "cbal73a8-29e7-4b17-a83b-86dc687772b5",
21   "token": "eyJhbGciOiJIUzI1Ni99.eyJzdWI0IjY1Y1pbC...",
22   "created": "2023-03-12T05:00:00.000+00:00",
23   "isActive": true,
24   "user": {
25     "id": "0ef7aac45-a3af-46fe-b231-3eb75d7181b3",
26     "name": "Camilln fnlaxdn".
27   }
28 }
```

# Consulta de todos los números telefónicos registrados

<http://localhost:8080/phones>

POST Save registerGET Get usersGET Get sessionsGET Get phonesGET Get locations+ ...No Environment

TEST NISUM / Get phones

GEThttp://localhost:8080/phones

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

TypeBearer Token

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. variables

The authorization header will be automatically generated when you send the request. Learn more about authorization

TokeneyJhbGciOiJIUzI1Ni99.eyJzdWI0IjY1Y1pbC...

BodyCookiesHeaders (11)Test Results

Status: 200 OKTime: 11 msSize: 564 B

PrettyRawPreviewVisualizeJSON

```
1 [
2   {
3     "citycode": "1",
4     "countrycode": "57",
5     "number": "3178285"
6   },
7   {
8     "citycode": "1",
9     "countrycode": "57",
10    "number": "3601212014"
11  },
12  {
13    "citycode": "4",
14    "countrycode": "57",
15    "number": "3144407086"
16  },
17  {
18    "citycode": "1",
19    "countrycode": "57",
20    "number": "3143207086"
21  }
22 ]
```

# Consulta de todas las locaciones registradas (códigos de área país, ciudad)

<http://localhost:8080/locations>

POST Save registerGET Get usersGET Get sessionsGET Get phonesGET Get locations+ \*\*\*

TEST NISUM / Get locations

GEThttp://localhost:8080/locations ...

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

noneform-data x-www-form-urlencodedrawbinaryGraphQL

This request does not have a body

BodyCookiesHeaders (11)Test Results

PrettyRawPreviewVisualizeJSON

123456789101112131415161718192021222324252627

```
{
  "type": "COUNTRY",
  "code": "51",
  "name": "Perú"
},
{
  "type": "COUNTRY",
  "code": "54",
  "name": "Argentina"
},
{
  "type": "COUNTRY",
  "code": "55",
  "name": "Brasil"
},
{
  "type": "COUNTRY",
  "code": "56",
  "name": "Chile"
},
{
  "type": "COUNTRY",
  "code": "57",
  "name": "Colombia"
}
```

Status: 200 OK

GRACIAS