

Plan de pruebas

1. Redacción de Historia de usuario

ID: US01

Título: Creación de Guía con servicio de Recaudo Contra Entrega

Descripción:

Cómo usuario del sistema de gestión de envíos, quiero poder crear una guía con el servicio de Recaudo contra entrega, para gestionar envíos en los que requieran un cobro al destinatario

Criterios de aceptación

1. La API debe permitir la creación exitosa de una guía cuando se proporciona una referencia de recaudo y un valor dentro del rango permitido (de \$1 a \$16'000.000). La respuesta de la API debe contener un código de remisión (*código_remision*), y el código de estado HTTP debe ser 200
2. Los campos "Referencia de recaudo" y el "Valor a recaudar" deben ser obligatorios. Si falta alguno de estos campos, la API debe devolver un error con código de estado 400, la respuesta debe incluir un mensaje indicando qué campo hace falta ingresar. Los mensajes que se deben parametrizar según sea el caso son:
 - Mensaje cuando falta el campo referenciaRecaudo: *"El campo referenciaRecaudo es requerido"*
 - Mensaje cuando falta el campo valorRecaudar: *"El campo valorRecaudar es requerido"*
 - Mensaje cuando faltan los dos campos: *"Los campos valorRecaudar y referenciaRecaudo son requeridos"*
3. El campo "Referencia de recaudo" debe permitir un máximo de 30 caracteres. Si el usuario ingresa una referencia que exceda este límite, la API debe responder con código *HTTP 400*, la respuesta debe incluir un mensaje claro, indicando el motivo por el cual no es posible crear la guía. Se debe parametrizar el siguiente mensaje: *"El campo referenciaRecaudo excede la cantidad de caracteres permitidos, 30"*
4. El campo "Valor a recaudar" debe estar dentro del rango permitido (\$1 y \$16'000.000):
 - Si el valor ingresado es menor a \$1, la API debe responder con código *HTTP 400*
 - Si el valor ingresado es mayor a \$16'000.000, la API debe responder con código *HTTP 400*.Se deben parametrizar los siguientes mensajes para cada caso: *"El campo valorRecaudar debe ser mayor a 1"* y *"El campo valorRecaudar debe ser menor o igual a 16000000"*
5. El campo "Valor a recaudar" debe permitir los valores mínimo y máximo
 - Si el valor ingresado es \$1, la API debe aceptarlo y responder con código *HTTP 200*
 - Si el valor ingresado es \$16000000, la API debe aceptarlo y responder con código *HTTP 200*
6. La API debe permitir la creación de la guía cuando se omiten campos opcionales, y responder con código *HTTP 200*
7. El campo "Referencia de recaudo" debe permitir el ingreso de caracteres especiales siempre que no se supere los 30 caracteres, la API debe responder con código *HTTP 200*
8. Después de crear una guía exitosamente, debe ser posible consultarla utilizando el endpoint de consulta

Escenarios

- TS01. Validar la creación de una guía con datos válidos
- TS02. Verificar la validación de campos obligatorios
- TS03. Comprobar la restricción de caracteres en el campo "Referencia recaudo"
- TS04. Verificar el rango permitido en el campo "Valor recaudar"
- TS05. Asegurar la aceptación de valores límite en el campo "Valor recaudar"
- TS06. Confirmar la creación de una guía con datos opcionales omitidos
- TS07. Validar la creación de una guía con caracteres especiales en el campo "Referencia recaudo"
- TS08. Validar almacenamiento y consulta de datos

2. PRUEBA MANUAL- API GENERACIÓN DE GUÍAS

Objetivo: Validar flujos principales del API de generación de guías en diferentes escenarios utilizando datos reales

Herramienta a usar para realizar las solicitudes manuales

- Postman

Casos de prueba basados en los escenarios definidos en la historia de usuario

ID: TC01

Título: Validar la creación exitosa de una guía con datos correctos

Descripción: Verificar que, al enviar una solicitud con todos los datos válidos, la API responde con un código de estado HTTP 200, almacena la guía correctamente y devuelve un código de remisión en la respuesta

Precondiciones: El servicio de la API está disponible

Datos de prueba:

- **Headers:** "Content-Type": "application/json"
- **Body:** `json {"referenciaRecaudo": "REF20240207XYZ123", "valorRecaudar": "4000000"}`

Pasos:

1. Enviar una solicitud *POST* al endpoint <https://apiv2-test.coordinadora.com/guias/cm-quias-ms/guia> con los datos de prueba
2. Validar que la API responde con código 200
3. Confirmar que la respuesta incluye un *código_remision* valido (no vacío)

Resultado esperado: la API debe responder con código de estado **200** y genera un *código_remision* con el siguiente cuerpo de respuesta:

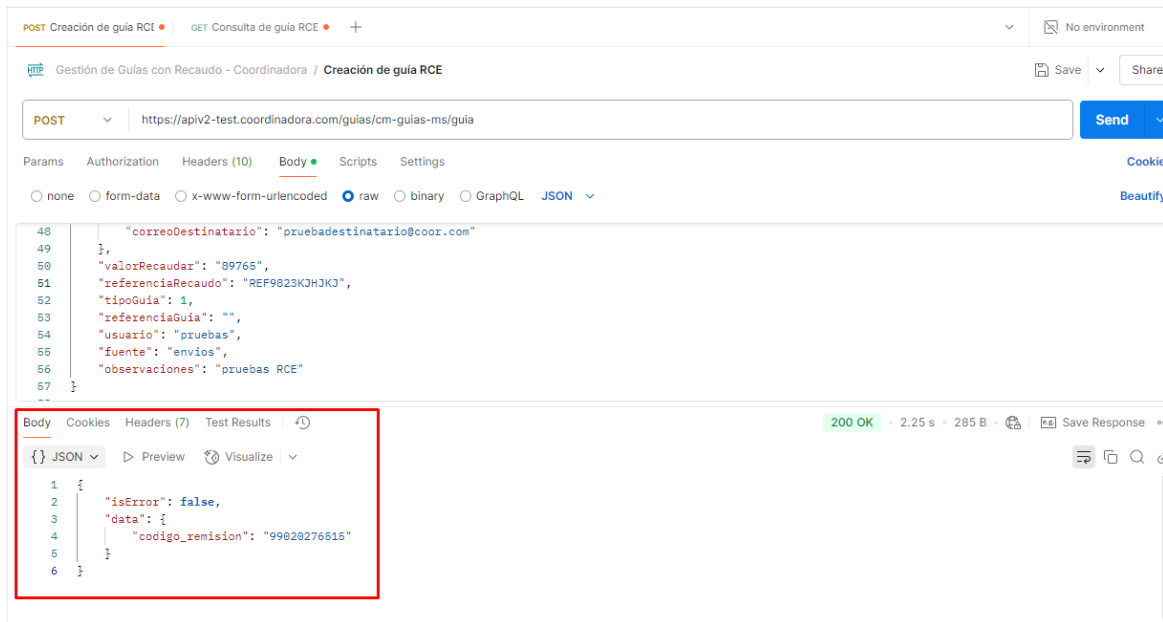
```
json {"isError": false,  
  "data": {  
    "codigo_remision": "aquí se mostrará el número de guía que se acaba de crear"}}
```

Resultado obtenido: La API no responde con código de estado **200** pero si genera un *código_remision* con el siguiente cuerpo de respuesta:

```
json {"isError": false,  
  "data": {  
    "codigo_remision": "aquí se mostrará el número de guía que se acaba de crear"  
  }}  
}}
```

Estado: **Fallido**

Evidencias:



ID: TC02

Título: Validar error al intentar crear una guía con el campo “referenciaRecaudo” vacío

Descripción: Validar que, si falta el campo “referenciaRecaudo”, la API devuelve código HTTP 400 y un mensaje de error claro

Precondiciones: El servicio de la API está disponible

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** json {no se ingresa nada en el campo referenciaRecaudo, “valorRecaudar”: “500000”}

Pasos:

1. Enviar una solicitud *POST* al endpoint <https://apiv2-test.coordinadora.com/quias/cm-quias-ms/quia> con los datos de prueba
2. Validar la respuesta de la API

Resultado esperado: la API debe responder con **400** y debe devolver el siguiente mensaje de error:

```
json {
  "isError": true,
  "message": {“Los valores de entrada no son correctos.”,
  "code": “BAD_MESSAGE”,
  "statusCode": 400,
  "cause": “El campo referenciaRecaudo es requerido”}
```

Resultado obtenido: la API responde con **400** con el siguiente mensaje de error:

```
json {
  "isError": true,
  "message": {“Los valores de entrada no son correctos.”,
  "code": “BAD_MESSAGE”,
  "statusCode": 400,
  "cause": “El campo referenciaRecaudo es requerido”}
```

Estado: Aprobado

Evidencias:

POST Creación de guía RCE • GET Consulta de guía RCE • +

HTTP Gestión de Guías con Recaudo - Coordinadora / Creación de guía RCE

Save Share

POST https://apiv2-test.coordinadora.com/guías/cm-guías-ms/guia Send

Params Authorization Headers (10) Body Scripts Settings Cookies Beautify

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

```
48     "correoDestinatario": "pruebadestinatario@coor.com"
49   },
50   "valorRecaudar": "89765",
51   "referenciaRecaudo": "",
52   "tipoGuia": 1,
53   "referenciaGuia": "",
54   "usuario": "pruebas",
55   "fuente": "envios",
56   "observaciones": "pruebas RCE"
57 }
--
```

Body Cookies Headers (7) Test Results 400 Bad Request • 276 ms • 392 B Save Response

JSON Preview Visualize

```
1 {
2   "isError": true,
3   "message": "Los valores de entrada no son correctos.",
4   "code": "BAD_REQUEST",
5   "statusCode": 400,
6   "cause": "El campo referenciaRecaudo es requerido"
7 }
```

ID: TC03

Título: Validar que, si falta el campo “valorRecaudar”, la API devuelve código HTTP 400 y un mensaje de error claro y conciso

Precondiciones: El servicio de la API está disponible

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** json {no se ingresa nada en el campo valorRecaudar, “referenciaRecaudo”: “REF1587”}

Pasos:

1. Enviar una solicitud *POST* al endpoint <https://apiv2-test.coordinadora.com/guías/cm-guías-ms/guia> con los datos de prueba
2. Validar la respuesta de la API

Resultado esperado: la API debe devolver un *HTTP 400* con un mensaje de error claro y conciso

Resultado obtenido: La API responde con *HTTP 400*, pero en el cuerpo de respuesta el mensaje no es muy claro
json {

“statusCode”: 400,

“error”: “Bad Request”,

“message”: “body.valorRecaudar should be number”}

Estado: **Fallido**

ID Bug relacionado: BUG-03

Evidencias:

REST client interface showing a POST request to `https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia`. The request body is a JSON object:

```
{  "indicativoDestinatario": "8/",  "celularDestinatario": "3216549825",  "correoDestinatario": "pruebadestinatario@coor.com",  "valorRecaudar": "",  "referenciaRecaudo": "REF20250208XYZ5487",  "tipoGuia": 1,  "referenciaGuia": "",  "usuario": "",  "fuente": "envios",  "observaciones": "prueba RCE"}
```

The response is a **400 Bad Request** with the following JSON body:

```
{  "statusCode": 400,  "error": "Bad Request",  "message": "body.valorRecaudar should be number"}
```

ID: TC04

Título: Validar que, si faltan los campos “valorRecaudar” y “referenciaRecaudo”, la API devuelve código HTTP 400 y un mensaje de error claro y conciso

Precondiciones: El servicio de la API está disponible

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** json {no se ingresa nada en los campos “valorRecaudar” y “referenciaRecaudo”}

Pasos:

1. Enviar una solicitud *POST* al endpoint <https://apiv2-test.coordinadora.com/guias/cm-quias-ms/quia> con los datos de prueba
2. Validar la respuesta de la API

Resultado esperado: la API debe devolver un *HTTP 400* con el siguiente cuerpo de respuesta:

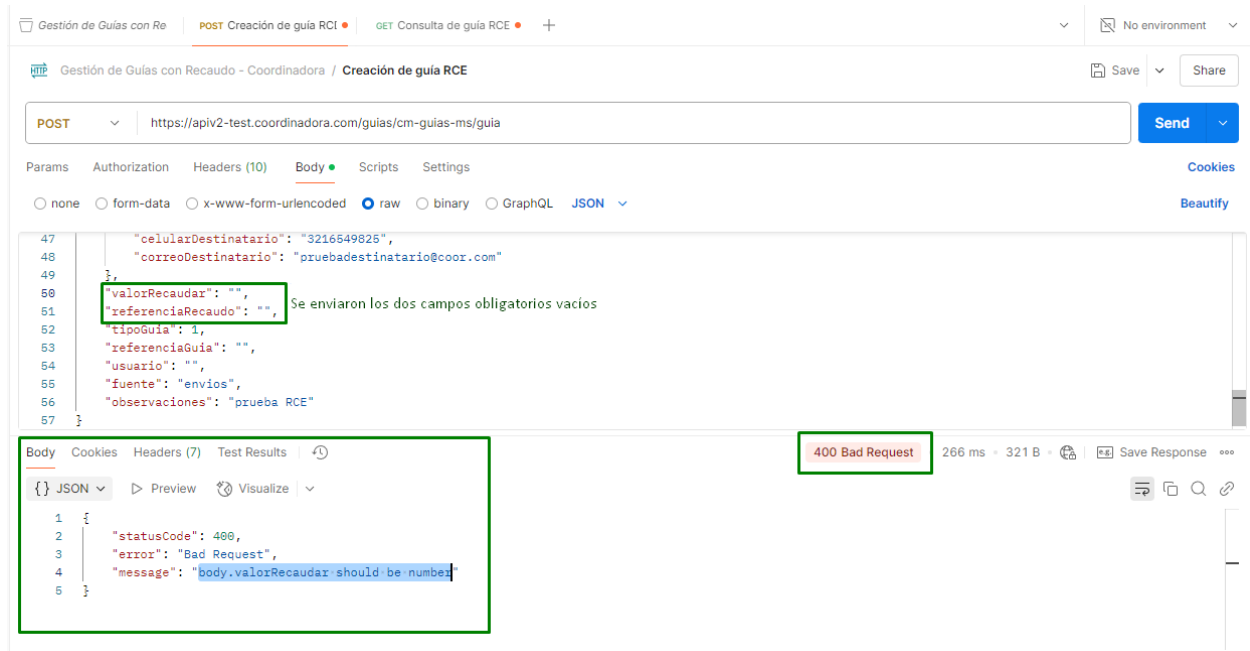
```
json {  "statusCode": 400,  "error": "Bad Request",  "message": "body.valorRecaudar should be number"}
```

Resultado obtenido: la API responde con *HTTP 400* con el siguiente cuerpo de respuesta:

```
json {  "statusCode": 400,  "error": "Bad Request",  "message": "body.valorRecaudar should be number"}
```

Estado: Aprobado

Evidencias:



ID: TC05

Título: Comprobar la restricción de caracteres en el campo “Referencia recaudo”

Descripción: Verificar que la API devuelva un error si usuario ingresa más de 30 caracteres en el campo “Referencia recaudo”

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** `json {`
 “Referencia recaudo”: “REF20250208XYZ741DEFGHIJKLMNOPQRSTUVWXYZ”,
 “valorRecaudar”: “80000”
}

Pasos:

1. Enviar una solicitud `POST` al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **400 Bad Request** con el siguiente cuerpo de respuesta:

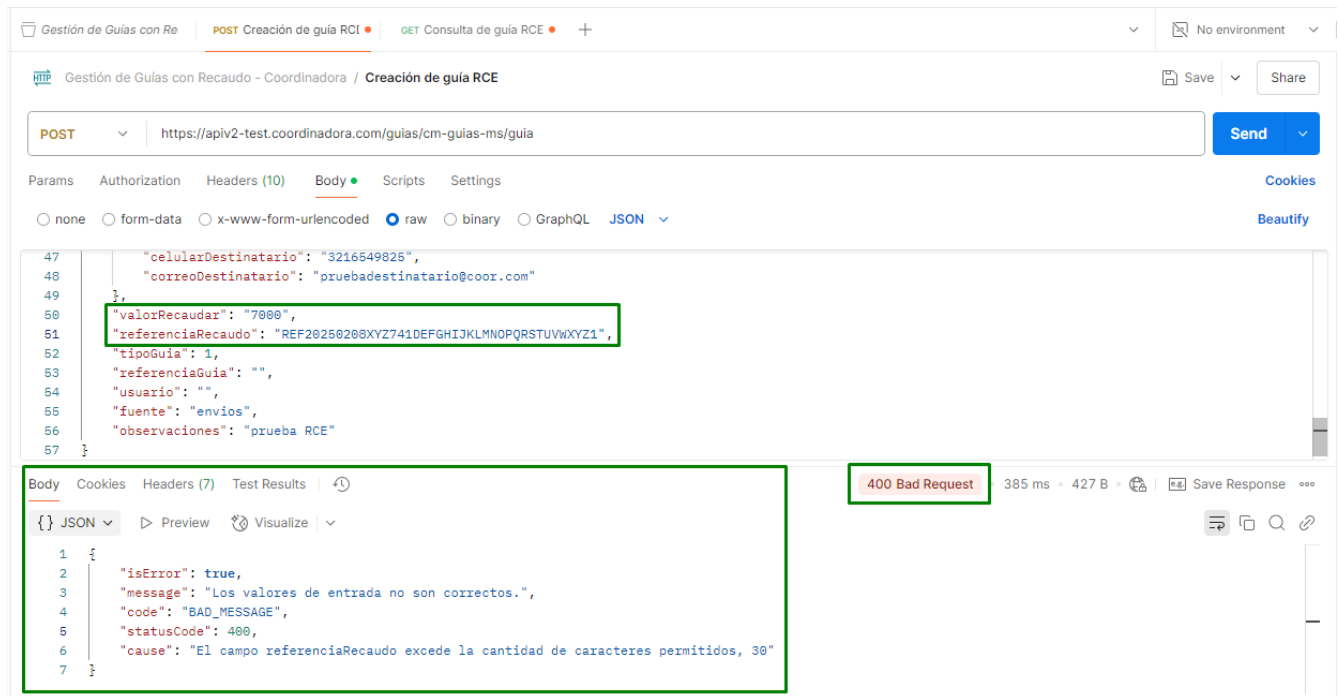
```
json {
  "isError": true,
  "message": { "Los valores de entrada no son correctos." },
  "code": "BAD_MESSAGE",
  "statusCode": 400,
  "cause": "El campo referenciaRecaudo excede la cantidad de caracteres permitidos, 30" }
```

Resultado obtenido: la API responde con **HTTP 400 Bad Request** y el siguiente cuerpo de respuesta:

```
json {
  "isError": true,
  "message": { "Los valores de entrada no son correctos." },
  "code": "BAD_MESSAGE",
  "statusCode": 400,
  "cause": "El campo referenciaRecaudo excede la cantidad de caracteres permitidos, 30" }
```

Estado: **Aprobado**

Evidencias:



ID: TC06

Título: Verificar que la API rechace valores menores a \$1 dentro del campo “Valor a recaudar”

Descripción: Validar que la API devuelva un error cuando el usuario intenta crear una guía con un valor a recaudar menor a \$1

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** `json {`
 “Referencia recaudo”: “REF20250208XYZ0022”,
 “valorRecaudar”: “0”`}`

Pasos:

1. Enviar una solicitud `POST` al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **400 Bad Request** con el siguiente cuerpo de respuesta:

```
{
  "isError": true,
  "message": "Los valores de entrada no son correctos.",
  "code": "BAD_MESSAGE",
  "statusCode": 400,
  "cause": "El campo valorRecaudar debe ser mayor a 1"
}
```

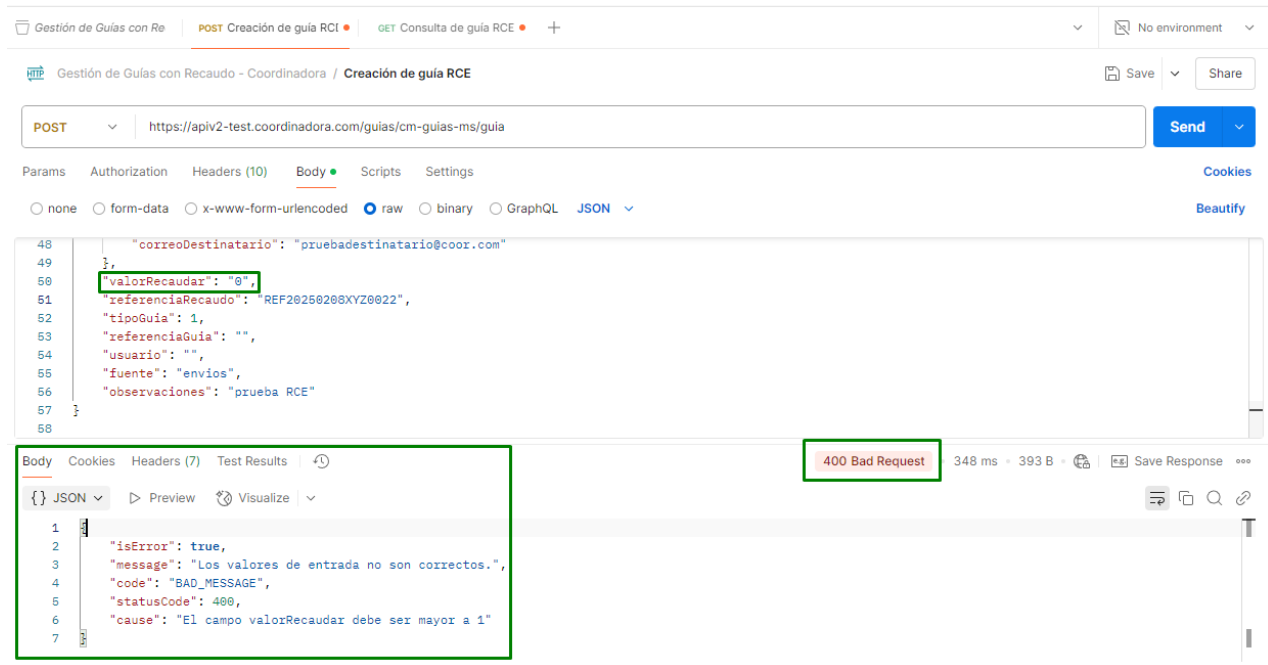
Resultado obtenido: la API responde con **HTTP 400 Bad Request** y el siguiente cuerpo de respuesta:

```
{
  "isError": true,
  "message": "Los valores de entrada no son correctos.",
  "code": "BAD_MESSAGE",
  "statusCode": 400,
  "cause": "El campo valorRecaudar debe ser mayor a 1"
}
```

}

Estado: **Aprobado**

Evidencias:



ID: TC07

Título: Verificar que la API rechace valores mayores a \$16'000.000 dentro del campo “Valor a recaudar”

Descripción: Validar que la API devuelva un error cuando el usuario intenta crear una guía con un valor a recaudar mayor a \$16'000.000

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** json {
“Referencia recaudo”: “REF20250208XYZ1024”,
“valorRecaudar”: “16'000.005”}

Pasos:

1. Enviar una solicitud **POST** al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **400 Bad Request** con el siguiente cuerpo de respuesta:

```
{  "isError": true,  "message": "Los valores de entrada no son correctos.",  "code": "BAD_MESSAGE",  "statusCode": 400,  "cause": "El campo valorRecaudar debe ser menor o igual a 16'000.000"}
```

Resultado obtenido: la API responde con **HTTP 400 Bad Request** y el siguiente cuerpo de respuesta:

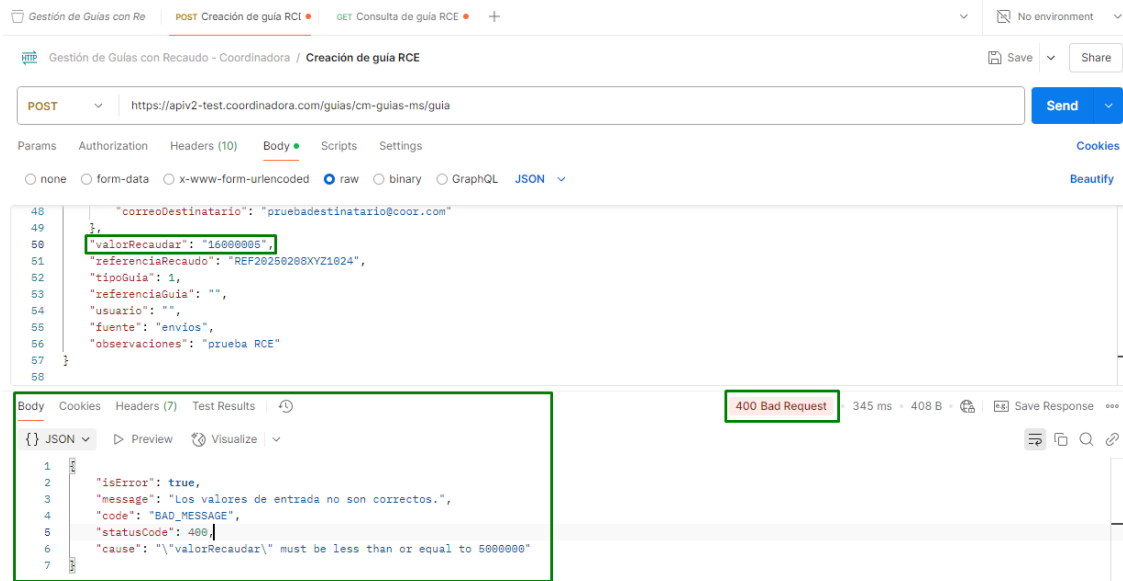
```
{  "isError": true,  "message": "Los valores de entrada no son correctos.",  "code": "BAD_MESSAGE",  "statusCode": 400,  "cause": "\"valorRecaudar\" must be less than or equal to 5000000"}
```


}

Estado: **No aprobado**

ID Bug relacionado: BUG-01

Evidencias:



ID: TC08

Título: Validar que la API acepte el valor mínimo permitido (\$1) en el campo “Valor recaudar”

Descripción: Verificar que la API permita la creación de una guía cuando el usuario ingresa el valor mínimo permitido (\$1) en el campo “Valor recaudar”

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** json {
 “Referencia recaudo”: “REF20250208XYZ5879”,
 “valorRecaudar”: “1”}

Pasos:

1. Enviar una solicitud *POST* al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **200 OK** con el siguiente cuerpo de respuesta:

```
{  
  "isError": false,  
  "data": {  
    "codigo_remision": " aquí se mostrará el número de guía que se acaba de crear "  
  }  
}
```

Resultado obtenido: la API responde con *HTTP 200 OK* y el siguiente cuerpo de respuesta:

```
{  
  "isError": false,  
  "data": {  
    "codigo_remision": " aquí se mostrará el número de guía que se acaba de crear "  
  }  
}
```

Estado: **Aprobado**

Evidencias:

The screenshot shows a REST client interface with a POST request to `https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia`. The request body is a JSON object with the following fields: `correoDestinatario`, `valorRecaudar` (highlighted with a green box), `referenciaRecaudo`, `tipoGuia`, `referenciaGuia`, `usuario`, `fuentes`, and `observaciones`. The response status is `200 OK` (highlighted with a green box), and the response body is a JSON object with `isError` set to `false` and `data` containing `codigo_remision` (highlighted with a green box).

```
48 {
49   "correoDestinatario": "pruebadestinatario@coor.com"
50   "valorRecaudar": "1",
51   "referenciaRecaudo": "REF20250208XYZ5879",
52   "tipoGuia": 1,
53   "referenciaGuia": "",
54   "usuario": "",
55   "fuentes": "envios",
56   "observaciones": "prueba RCE"
57 }
58
```

```
{
  "isError": false,
  "data": {
    "codigo_remision": "99020276349"
  }
}
```

ID: TC09

Título: Validar que la API acepte el valor máximo permitido (\$16'000.000) en el campo "Valor recaudar"

Descripción: Verificar que la API permita la creación de una guía cuando el usuario ingresa el valor mínimo permitido (\$16'000.000) en el campo "Valor recaudar"

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** "Content-Type": "application/json"
- **Body:** `json {`
"Referencia recaudo": "REF20250208XYZ71",
"valorRecaudar": "16000000"}
`}`

Pasos:

1. Enviar una solicitud `POST` al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **200 OK** con el siguiente cuerpo de respuesta:

```
{
  "isError": false,
  "data": {
    "codigo_remision": "aquí se mostrará el número de guía que se acaba de crear"
  }
}
```

Resultado obtenido: la API responde con `HTTP 400 Bad Request` y el siguiente cuerpo de respuesta:

```
{
  "isError": true,
  "message": "Los valores de entrada no son correctos.",
  "code": "BAD_MESSAGE",
  "statusCode": 400,
  "cause": "\"valorRecaudar\" must be less than or equal to 5000000"
}
```

Estado: No aprobado

ID Bug relacionado: BUG-01

Evidencias:

Gección de Guías con Re POST Creación de guía RCE GET Consulta de guía RCE

POST https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia

Params Authorization Headers (10) Body Scripts Settings

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

```
48 {
49   "correoDestinatario": "pruebadestinatario@coor.com"
50   "valorRecaudar": "16000000",
51   "referenciaRecaudo": "REF20250208XYZ5879",
52   "tipoGuia": 1,
53   "referenciaGuia": "",
54   "usuario": "",
55   "fuente": "envios",
56   "observaciones": "prueba RCE"
57 }
58
```

400 Bad Request 340 ms · 407 B Save Response

```
1 {
2   "isError": true,
3   "message": "Los valores de entrada no son correctos.",
4   "code": "BAD_MESSAGE",
5   "statusCode": 400,
6   "cause": "\"valorRecaudar\" must be less than or equal to 5000000"
7 }
```

ID: TC10

Título: Validar la creación de una guía cuando se omiten campos opcionales

Descripción: Verificar que la API permita la creación de una guía cuando el usuario no incluye campos opcionales en la solicitud

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** "Content-Type": "application/json"
- **Body:** json {
"Referencia recaudo": "REF20250208XYZ40588",
"valorRecaudar": "25000"
"observaciones": dejarlo vacío}

Pasos:

1. Enviar una solicitud POST al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **200 OK** con el siguiente cuerpo de respuesta:

```
{
  "isError": false,
  "data": {
    "codigo_remision": " aquí se mostrará el número de guía que se acaba de crear "
  }
}
```

Resultado obtenido: la API responde con **HTTP 400 Bad Request** y el siguiente cuerpo de respuesta:

```
{
  "isError": false,
  "data": {
    "codigo_remision": " aquí se mostrará el número de guía que se acaba de crear "
  }
}
```

Estado: Aprobado

Evidencias:

The screenshot shows a REST client interface with a POST request to `https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia`. The request body is a JSON object with the following fields: `correoDestinatario`, `valorRecaudar`, `referenciaRecaudo`, `tipoGuia`, `referenciaGuia`, `usuario`, `fuentes`, and `observaciones`. The response status is `200 OK` with a response time of 2.09 s and a size of 285 B. The response body is a JSON object with `isError` set to `false` and `data` containing `codigo_remision`.

```
POST https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia

{
  "correoDestinatario": "pruebadestinatario@coor.com",
  "valorRecaudar": "25000",
  "referenciaRecaudo": "REF20250588/*-+@!/?¿=)(/&.'!\"",
  "tipoGuia": 1,
  "referenciaGuia": "",
  "usuario": "pruebas",
  "fuentes": "envios",
  "observaciones": ""
}
```

```
{
  "isError": false,
  "data": {
    "codigo_remision": "99020276354"
  }
}
```

ID: TC11

Título: Validar la creación de una guía con caracteres especiales en el campo “Referencia recaudo”

Descripción: Verificar que la API permita la creación de una guía cuando el usuario ingresa caracteres especiales en el campo Referencia recaudo, siempre que se respete el límite de caracteres

Precondiciones: Tener acceso al API

Datos de prueba:

- **Headers:** “Content-Type”: “application/json”
- **Body:** `json {`
“Referencia recaudo”: “REF20250588/*-+@!/?¿=)(/&.'!”,
“valorRecaudar”: “25000”}

Pasos:

1. Enviar una solicitud `POST` al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Incluir los headers y el body con los datos de prueba
3. Validar la respuesta HTTP y el mensaje de error

Resultado esperado: la API debe devolver un **200 OK** con el siguiente cuerpo de respuesta:

```
{
  "isError": false,
  "data": {
    "codigo_remision": " aquí se mostrará el número de guía que se acaba de crear "
  }
}
```

Resultado obtenido: la API responde con `HTTP 400 Bad Request` y el siguiente cuerpo de respuesta:

```
{
  "isError": false,
  "data": {
    "codigo_remision": " aquí se mostrará el número de guía que se acaba de crear "
  }
}
```

Estado: Aprobado

Evidencias:

The screenshot shows a REST client interface with a POST request to `https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia`. The request body is a JSON object with the following fields: `correoDestinatario`, `valorRecaudar`, `referenciaRecaudo`, `tipoGuia`, `referenciaGuia`, `usuario`, `fuentes`, and `observaciones`. The `referenciaRecaudo` field is highlighted with a green box. The response status is `200 OK`, and the response body is a JSON object with `isError` set to `false` and `data` containing a `codigo_remision` value.

```
48 {
49   "correoDestinatario": "pruebadestinatario@coor.com",
50   "valorRecaudar": "25000",
51   "referenciaRecaudo": "REF20250288/*-+@!7&=) (/&*"/,
52   "tipoGuia": 1,
53   "referenciaGuia": "",
54   "usuario": "pruebas",
55   "fuentes": "envios",
56   "observaciones": "pruebas RCE"
57 }
58
```

```
1 {
2   "isError": false,
3   "data": {
4     "codigo_remision": "99020276356"
5   }
6 }
```

ID: TC12

Título: Verificar que el número de guía generado pueda ser consultado después de crearlo

Descripción: Validar que después de crear una guía, el número de guía generado pueda ser consultado correctamente mediante el endpoint de consulta y que el cuerpo de respuesta contenga únicamente un registro

Precondiciones: El usuario tiene acceso al API

Datos de prueba:

- **Headers:** "Content-Type": "application/json"
- **Body:** `json {`
"Referencia recaudo": "REF20250288",
"valorRecaudar": "98000"}

Pasos:

1. Enviar una solicitud `POST` al endpoint <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
2. Copiar el número de guía generado en la respuesta
3. Enviar una solicitud `GET` al endpoint de consulta por número de guía <https://apiv2-test.coordinadora.com/guias/cm-guias-consultas-ms/guia/aca se ingresa el número de guía generado>
4. Validar la respuesta contiene exactamente un registro en `data`
5. Confirmar que el código de estado HTTP es `200 OK`

Resultado esperado:

- La API debe devolver un HTTP **200 OK**
- El cuerpo de la respuesta debe contener exactamente un registro en `data`

Resultado obtenido: la API responde con `HTTP 200 Ok` y con un único registro para ese número de guía dentro de `data`

Estado: Aprobado

Evidencias:

Creación del número de guía

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

- URL:** `https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia`
- Method:** POST
- Body (JSON):**

```
{  "correoDestinatario": "pruebadestinatario@coor.com",  "valorRecaudar": "98000",  "referenciaRecaudo": "REF20250288",  "tipoGuia": 1,  "referenciaGuia": "",  "usuario": "pruebas",  "fuente": "envios",  "observaciones": "pruebas RCE"}
```
- Response:** 200 OK (3.21 s, 285 B)

```
{  "isError": false,  "data": {    "codigo_remision": "99020276357"  }}
```

BUGS ENCONTRADOS

ID: BUG-01

Título: No permite ingresar un valorRecaudar mayor a \$5'000.000 cuando debería permitir un valor hasta \$16'000.000

Descripción: Actualmente, la API rechaza valores mayores a \$5'000.000 en el campo *valorRecaudar*. Pero según los criterios de aceptación, debería aceptar un valor hasta \$16'000.000

Precondiciones: Tener acceso a la API

Test data:

- Endpoint: POST <https://apiv2-test.coordinadora.com/guias/cm-quias-ms/guia>
- Headers: "Content-Type": "application/json"
- Body de la solicitud:

```
{  "valorRecaudar": "16000000",  "referenciaRecaudo": "REF20250288",}
```

Pasos para reproducir el bug:

- Enviar una solicitud POST al endpoint con un valorRecaudar mayor a \$5'000.000

Resultado obtenido:

- La API devuelve un *400 Bad Request*,

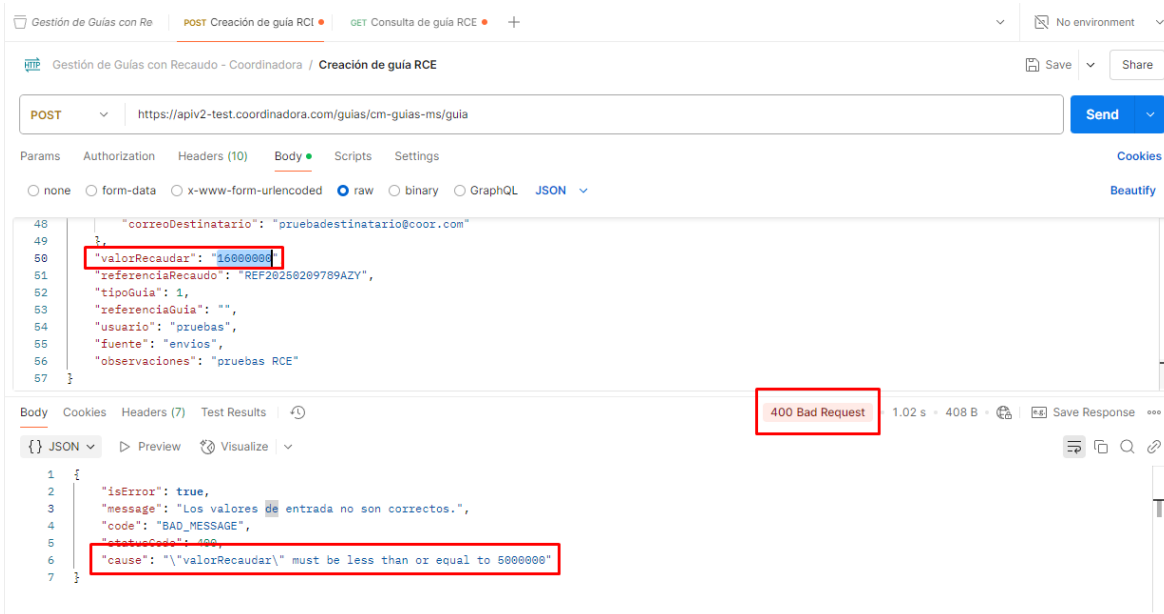
```
{  "isError": true,  "message": "Los valores de entrada no son correctos.",  "code": "BAD_MESSAGE",  "statusCode": 400,  "cause": "\"valorRecaudar\" must be less than or equal to 5000000"}
```
- La API no permite valores mayores a \$5'000.000, lo cual es incorrecto

Resultado esperado:

- La API debería aceptar valores hasta \$16'000.000 en el campo valorRecaudar

Prioridad: Alta

Evidencia:



ID: BUG-02

Título: Mensaje de error genérico cuando se rechaza un valorRecaudar mayor a \$5'000.000

Descripción:

- Cuando se envía un valorRecaudar superior a \$5'000.000, la API devuelve un mensaje poco claro
- El *message* es genérico y no explica el motivo del error, además el mensaje de error sale en inglés, cuando debería estar en español
- La razón real solo está en *cause*

Precondiciones: Tener acceso al API

Test data:

- Endpoint: POST <https://apiv2-test.coordinadora.com/guias/cm-quias-ms/guia>
- Body de la solicitud:

```
{  "valorRecaudar": "16000000",  "referenciaRecaudo": "REF20250288",}
```

Pasos:

- Enviar una solicitud POST al endpoint con un valorRecaudar mayor a \$5'000.000

Resultado obtenido:

- La API devuelve un mensaje de error poco claro:

```
{  "isError": true,  "message": "Los valores de entrada no son correctos.",  "code": "BAD_MESSAGE",  "statusCode": 400,  "cause": "\"valorRecaudar\" must be less than or equal to 5000000"}
```

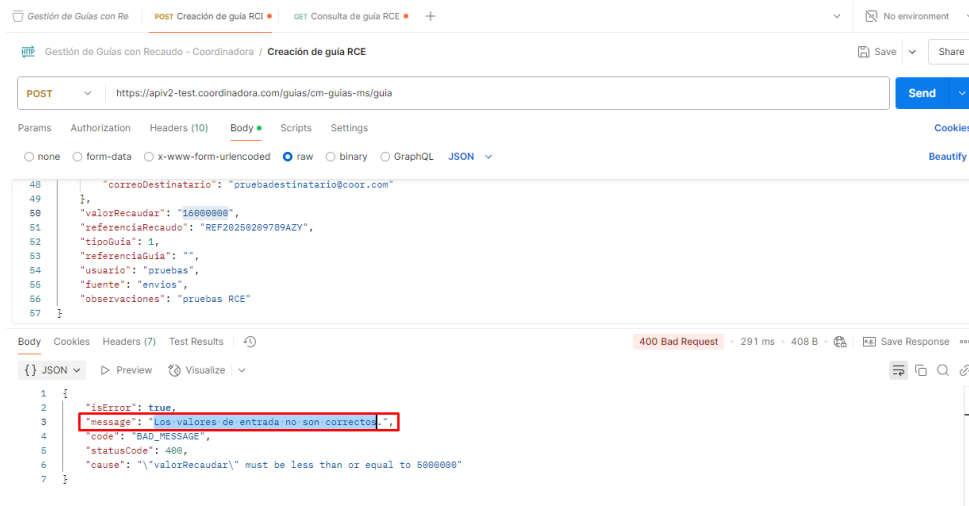
Resultado esperado:

- La API debería mostrar un *message* más claro para el usuario, por ejemplo:

```
{
  "isError": true,
  "message": "El valor a recaudar no puede ser mayor a 16000000",
  "code": "BAD_MESSAGE",
  "statusCode": 400,
  "cause": "El campo valorRecaudar debe ser menor o igual a 16'000.000"
}
```

Prioridad: Media ya que no afecta la funcionalidad, pero mejora la usabilidad de la API

Evidencia:



ID: BUG-03

Título: Mensaje de error incorrecto cuando el campo *valorRecaudar* y *referenciaRecaudo* están vacíos

Descripción: Cuando los dos campos obligatorios (*valorRecaudar* y *referenciaRecaudo*) se envían vacíos en la solicitud, la API responde con un *400 Bad Request*, pero el mensaje de error solo menciona que *valorRecaudar* debe ser número, ignorando el campo *referenciaRecaudo*. Además, el mensaje de error sale en inglés, cuando debería estar en español

Precondiciones: Tener acceso al API

Test data:

- Endpoint: POST <https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia>
- Body de la solicitud:

```
{
  "valorRecaudar": "",
  "referenciaRecaudo": "",
}
```

Pasos:

- Enviar una solicitud POST al endpoint con un *valorRecaudar* y *referenciaRecaudo* vacíos

Resultado obtenido:

- La API responde con un *400 Bad Request*, pero el mensaje de error solo menciona *valorRecaudar*, ignorando el campo *referenciaRecaudo*, y además el mensaje está en inglés

```
{
  "statusCode": 400,
  "error": "Bad Request",
  "message": "body.valorRecaudar should be number"
}
```

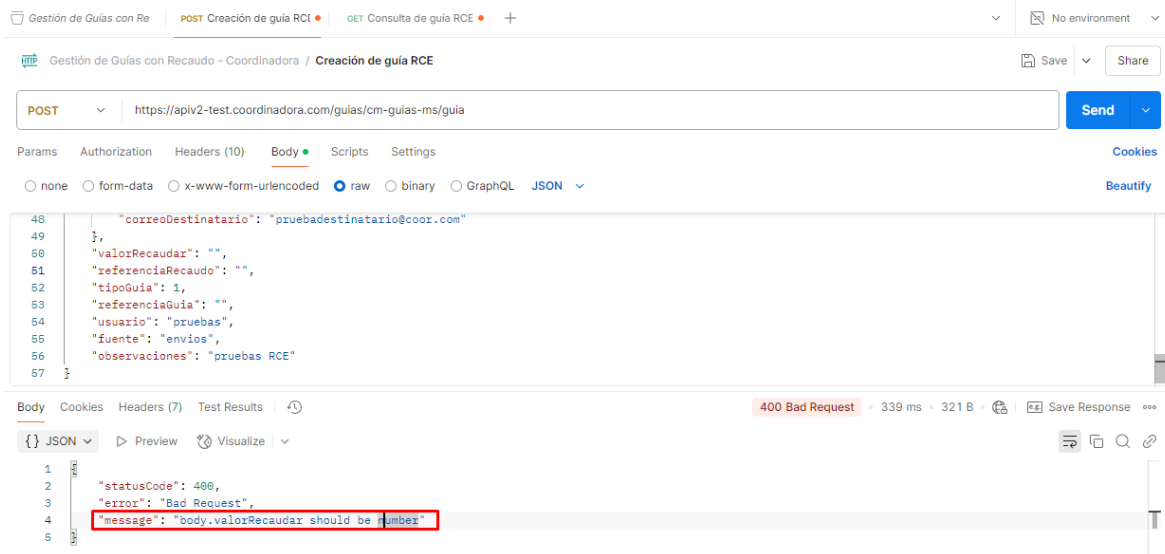

Resultado esperado:

- La API debería mostrar un *400 Bad Request* con un mensaje más claro y en español indicando que los campos son obligatorios, por ejemplo:

```
{
  "statusCode": 400,
  "error": "Bad Request",
  "message": "Los campos valorRecaudar y referenciaRecaudo son requeridos"
}
```

Prioridad: Media ya que no bloque la funcionalidad, pero afecta la claridad del error y la experiencia de usuario

Evidencia:



2. ESCENARIOS BDD

Feature: Validación de la API de creación de guías

Escenario: Validar la creación de una guía con datos válidos

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud con todos los campos obligatorios correctamente diligenciados

Entonces el sistema debe aceptar la solicitud y responder con un código de estado 200

Escenario: Validar error cuando referenciaRecaudo está vacío

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud con "Referencia recaudo" vacío

Entonces el sistema debe rechazar la solicitud con un código de estado 400

Escenario: Validar error cuando los campos obligatorios están vacíos

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud sin ingresar datos en los campos obligatorios "Valor a recaudar y Referencia recaudo"

Entonces el sistema debe rechazar la solicitud con un código de estado 400

Escenario: Validar restricción de caracteres en el campo "Referencia recaudo"

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud con un valor en el campo "referenciaRecaudo" que excede la cantidad máxima de caracteres permitidos

Entonces el sistema debe rechazar la solicitud con un código de estado 400

Y debe mostrar un mensaje de error indicando la cantidad máxima de caracteres permitida

Escenario: Verificar el rango permitido en el campo “Valor a recaudar”

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud con un valor fuera del rango permitido en el campo “Valor a Recaudar”

Entonces el sistema debe rechazar la solicitud y responder con un código de estado 400

Y debe mostrar un mensaje de error indicando el rango permitido

Escenario: Validación de carga y respuesta en límite de datos

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud con el valor mínimo permitido en el campo “Valor a recaudar”

Entonces el sistema debe aceptar la solicitud y responder con un código de estado 200

Escenario: Manejo de concurrencia en creación de guías

Dado que múltiples usuarios están creando guías simultáneamente

Cuando todos los usuarios envían solicitudes válidas al endpoint

Entonces el sistema debe responder consistentemente con un tiempo promedio inferior a 500ms

Y no debe generar errores relacionados con la concurrencia

Escenario: Validar la respuesta del sistema ante caracteres especiales en el campo “Referencia de recaudo”

Dado que soy un usuario autenticado en el sistema de guías

Cuando envío una solicitud con caracteres especiales en el campo “Referencia recaudo”

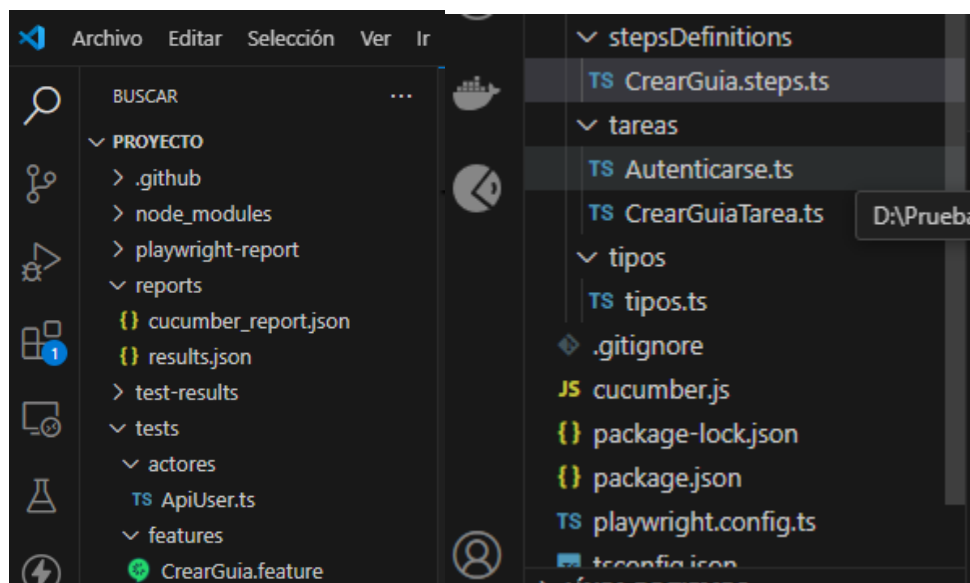
Entonces el sistema debe aceptar la solicitud y responder con un código de estado 200

3. Prueba funcional con Playwright y Screenplay

Descripción Breve del Proyecto:

Automatización de pruebas para la API de creación de guías utilizando Playwright y el patrón Screenplay implementado con Serenity/JS. El proyecto fue estructurado de manera modular, utilizando actores, tareas para interactuar con la API y validar respuestas. Los escenarios de prueba se definieron usando Gherkin en formato BDD, permitiendo una descripción clara y entendible de los comportamientos esperados.

- Estructura del proyecto



```
TS CrearGuia.steps.ts x JS cucumber.js TS Autenticarse.ts TS ApiUser.ts TS tipos.ts
tests > stepsDefinitions > TS CrearGuia.steps.ts > When('envío una solicitud con todos los campos obligatorios correctamente diligenciados') callback > datosDePrueba > referenc
12 let resultadoAPIError: RespuestaError
13 let resultadoAPIVacios: RespuestaCamposVacios
14
15 Given('que soy un usuario autenticado en el sistema de guías', async function() {
16   api = await request.newContext()
17   apiuser = new ApiUser(api)
18   //Simulación de login
19   await apiuser.attemptsTo(Autenticarse())
20 })
21
22 When('envío una solicitud con todos los campos obligatorios correctamente diligenciados', async function() {
23   const datosDePrueba = {
24     referenciaRecaudo: 'REF87654897',
25     valorRecaudar: '8800'
26   }
27
28   console.log('Datos de prueba', datosDePrueba)
29   const actor = {}; // Acá se proporciona un actor vacío, ya que no se necesita en este caso
30   resultadoAPIExitosa = await CrearGuiaTarea(actor, datosDePrueba) as RespuestaExitosa
31 })
32
```

PROBLEMAS SALIDA RESULTADOS DE LA PRUEBA TERMINAL PUERTOS POSTMAN CONSOLE PLAYWRIGHT COMENTARIOS ... powershell + v [icon] ... ^ x

✓ Usuario autenticado correctamente.

=====

.Solicitud enviada con referenciaRecaudo vacío: { referenciaRecaudo: '', valorRecaudar: '8800' }

.Prueba exitosa, se recibió el código de estado 400

Respuesta de la API: {

 isError: true,

 message: 'Los valores de entrada no son correctos.',

 code: 'BAD_REQUEST',

 statusCode: 400,

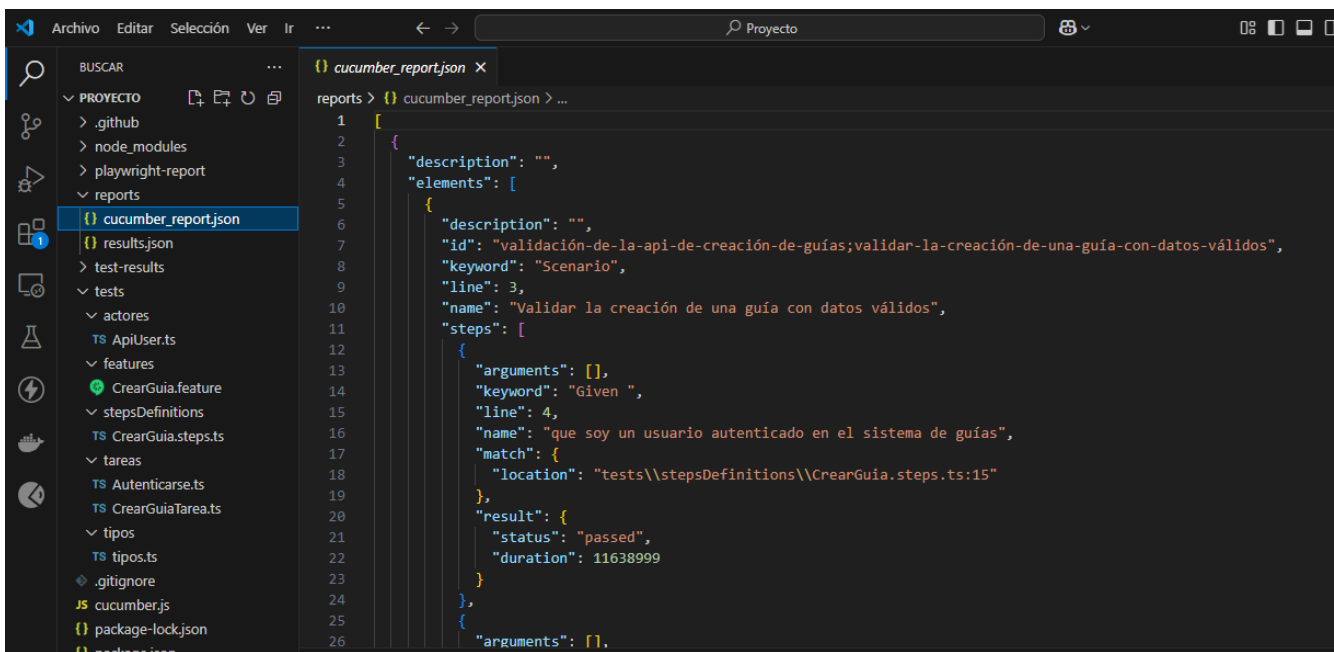
 cause: 'El campo referenciaRecaudo es requerido'

Lín. 24, col. 36 Espacios: 2 UTF-8 CRLF () TypeScript Go Live [icon] II Ninja Prettier [icon]

```
TS ApiUser.ts x
tests > actores > TS ApiUser.ts > ...
3 export class ApiUser {
20 }
21
22 async attemptsTo(...tasks: ((apiUser: ApiUser) => Promise<any>)[]) {
23   for (const task of tasks) {
24     await task(this);
25   }
26 }
27 }
28
```

```
TS CrearGuiaTarea.ts X
tests > tareas > TS CrearGuiaTarea.ts > CrearGuiaTarea
1  import { RespuestaExitosa } from '../tipos/tipos'; // Se importa el tipo definido para la respuesta
2  import { RespuestaError } from '../tipos/tipos'; // Se importa el tipo definido para la respuesta
3  import fetch from 'node-fetch';
4  import { RespuestaCamposVacios } from '../tipos/tipos'; // Se importa el tipo definido para la respuesta
5
6
7  export async function CrearGuiaTarea(actor: any, datosDePrueba: { referenciaRecaudo: string, valorRecaudar: string }
8  const url = 'https://apiv2-test.coordinadora.com/guias/cm-guias-ms/guia';
9
10 const headers = {
11   'Content-Type': 'application/json',
12 };
13 const body = JSON.stringify({
14   ...datosDePrueba,
15   identificacion: '890904713',
16   divisionCliente: '00',
17   idProceso: 100001,
18   codigoPais: 170,
19   valoracion: '200000',
20   tipoCuenta: 3,
21   contenido: 'reloj',
```

Se generan reportes automáticos



```
Archivo  Editar  Selección  Ver  Ir  ...  Proyecto
BUSCAR
PROYECTO
  > .github
  > node_modules
  > playwright-report
  > reports
  {} cucumber_report.json
  {} results.json
  > test-results
  > tests
    > actores
    TS ApiUser.ts
    > features
    ● CrearGuia.feature
    > stepsDefinitions
    TS CrearGuia.steps.ts
    > tareas
    TS Autenticarse.ts
    TS CrearGuiaTarea.ts
    > tipos
    TS tipos.ts
    > .gitignore
    JS cucumber.js
    {} package-lock.json
    {} package.json

cucumber_report.json > ...
1  reports > {} cucumber_report.json > ...
2  [
3    {
4      "description": "",
5      "elements": [
6        {
7          "description": "",
8          "id": "validación-de-la-api-de-creación-de-guias;validar-la-creación-de-una-guia-con-datos-válidos",
9          "keyword": "Scenario",
10         "line": 3,
11         "name": "Validar la creación de una guía con datos válidos",
12         "steps": [
13           {
14             "arguments": [],
15             "keyword": "Given ",
16             "line": 4,
17             "name": "que soy un usuario autenticado en el sistema de guías",
18             "match": {
19               "location": "tests\\stepsDefinitions\\CrearGuia.steps.ts:15"
20             },
21             "result": {
22               "status": "passed",
23               "duration": 11638999
24             }
25           },
26           {
27             "arguments": [],
```

```

✓ Usuario autenticado correctamente.
=====
.Datos de prueba { referenciaRecaudo: 'REF87654897', valorRecaudar: '8800' }
.Respuesta de la API: { isError: false, data: { codigo_remision: '99020276606' } }
Número de guía creado: 99020276606
Prueba exitosa
=====
✓ Usuario autenticado correctamente.
=====
.Solicitud enviada con referenciaRecaudo vacío: { referenciaRecaudo: '', valorRecaudar: '8800' }
.Prueba exitosa, se recibió el código de estado 400
Respuesta de la API: {
  isError: true,
  message: 'Los valores de entrada no son correctos.',
  code: 'BAD_REQUEST',
  statusCode: 400,
  cause: 'El campo referenciaRecaudo es requerido'
}

```

4. Pruebas de carga

Realizar pruebas de carga y estrés sobre la API: URL: <https://apiv2-test.coordinadora.com/guias/cm-guias-consultas-ms/guia/99020012725>

Se busca evaluar el rendimiento del sistema bajo diferentes condiciones de carga, identificando su límite máximo y comportamiento ante un aumento gradual de usuarios y solicitudes

1. Configuración del entorno

1.1. Herramientas utilizadas

- Se utilizará la herramienta JMeter
- Sistema operativo: Windows 10
- Java 23.0.1

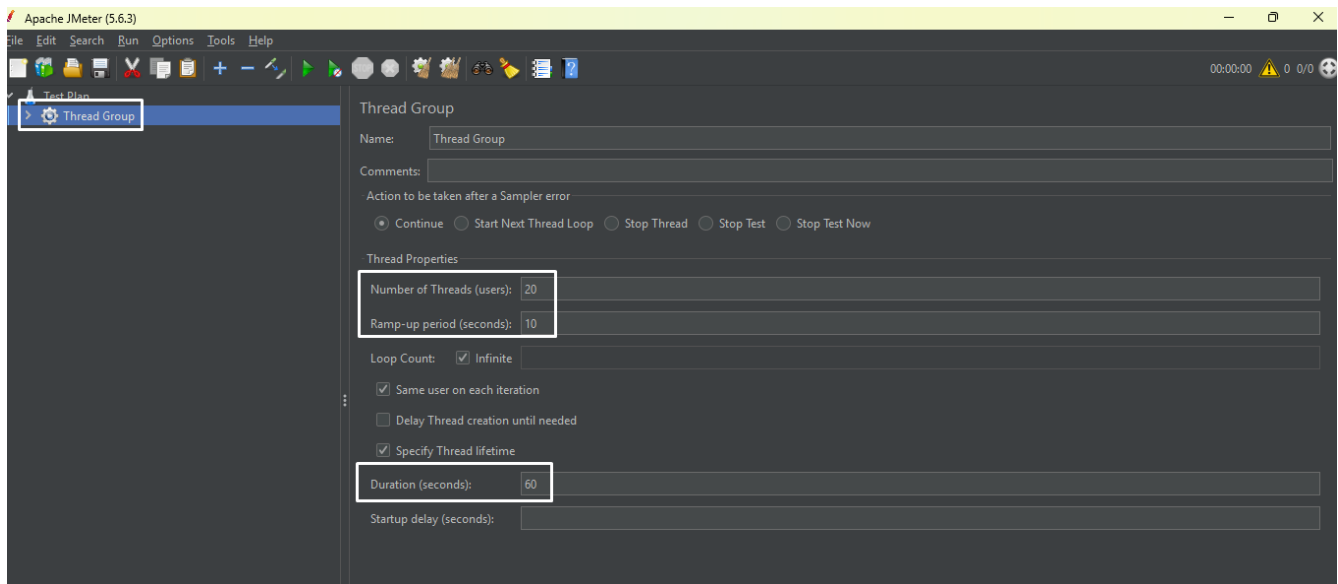
2. Configuración de las pruebas

2.1. Prueba de carga:

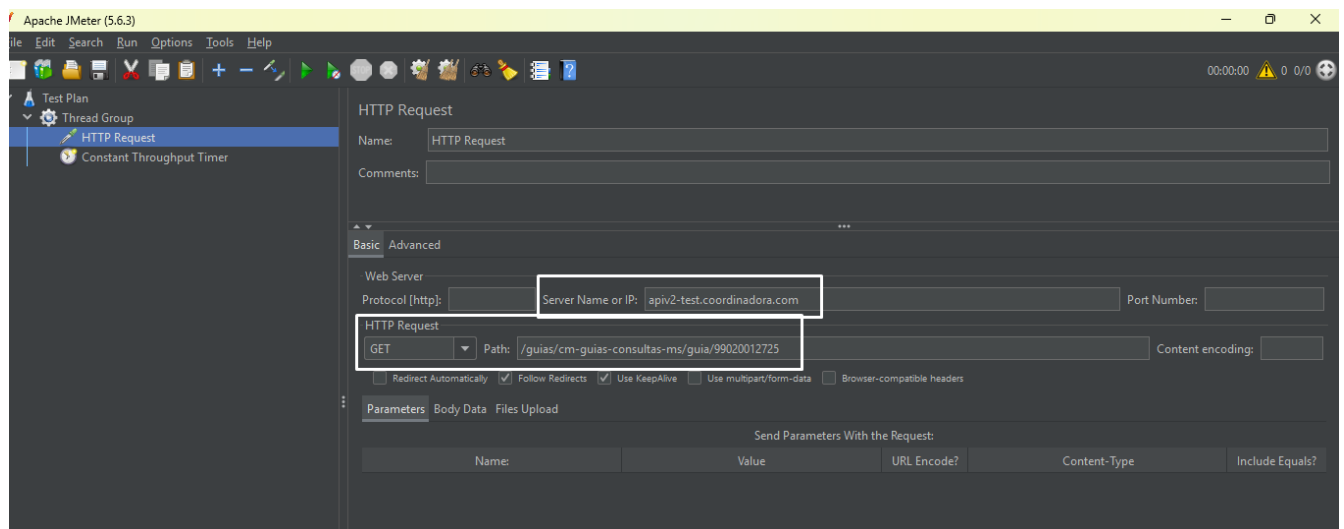
- Usuarios simultáneos: 20
- Frecuencia: 2 solicitudes/segundo
- Duración: 1 minuto
- Total esperado: 100 solicitudes

Configuración en JMeter:

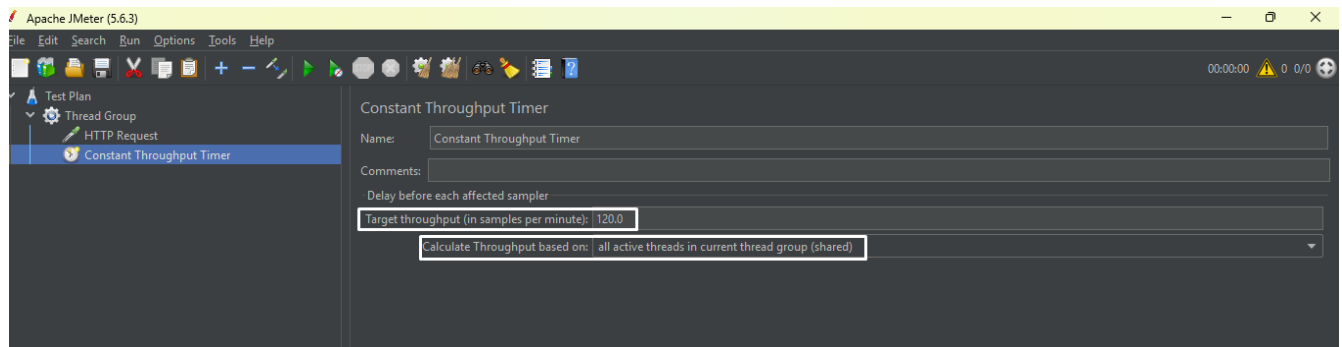
- Se debe crear un Test Plan y agregar un Thread Group con los siguientes datos:
 - Threads: 20
 - Ramp-up: 10
 - Duration: 60



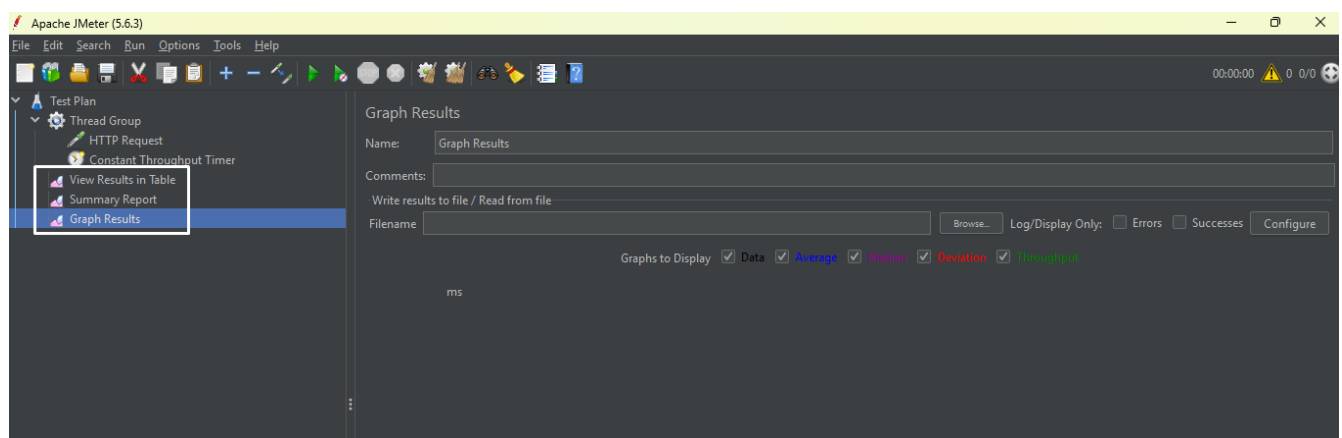
- Agregar un HTTP Request (Simulación de peticiones) con los siguientes datos en el campo correspondiente:
 - Method: GET
 - Server: apiv2-test.coordinadora.com
 - /guias/cm-guias-consultas-ms/guia/99020012725



3. Agregar el Throughout Timer



4. Se agregan Listeners para ver los resultados



5. Ejecución de pruebas

- Prueba de carga ejecutada a las 04:14:05.823 am del 10 de febrero de 2025
- Prueba de e

ANALISIS DE RESULTADOS

Prueba de carga

- **Objetivo:** ver como el sistema responde bajo una carga de 20 usuario simultaneos y 100 solicitudes en total
- **Resultados esperados:**
 - Tiempo de respuesta promedio ≤ 500 ms
 - Tasa de éxito $\geq 99\%$: se espera que el 99% de las solicitudes se completen con éxito sin errores
 - Uso de recursos: CPU $\leq 70\%$, memoria $\leq 80\%$
- **Resultado obtenido:**
 - Total de solicitudes: Se realizaron un total de 243 solicitudes durante la prueba
 - Tiempo de respuesta promedio: 322 ms
 - Tiempo total para procesar todas las solicitudes: 2675 ms
 - Errores: No se registraron errores durante la ejecución

