**Ejercicio 3**

**Técnicas para generar las pruebas**

Ejercicio 1

En el ejercicio 1 - API temperatura, cree una tabla en Excel para escribir los casos de prueba, los atributos de los casos de prueba son:

**ID**: Identificador de los casos de prueba.

**Descripción**: Oración corta para describir el objetivo de la prueba.

**Datos**: HTTP Request URL y precondiciones.

**Pasos**: Lista de acciones que describen el caso de prueba.

**Resultado** **esperado**: Descripción del comportamiento esperado del caso de prueba.

Nota: no fue incluido el atributo método porque era GET para todos los casos de prueba.

Realicé 11 casos de pruebas manuales, en donde abarqué todos los escenarios posibles positivos que me den como respuesta un status code = 200 OK y luego abarqué todos los casos posibles negativos que me den como respuesta un status code = 400 Bad Request y 500 Internal Server Error, también tuve en cuenta el caso de prueba para la respuesta “País y Ciudad incorrectos” en el escribí uno de los tantos ejemplos de país y ciudad incorrectos.

Ejercicio 2

En el ejercicio 2 – Automatización de API, la API es la siguiente: [**https://jsonplaceholder.typicode.com**](https://jsonplaceholder.typicode.com), a la cual accedí a través del Google Chrome para investigar un poco más sobre los recursos, rutas, guías, etc.

A partir de una investigación de los distintos endopoints, los recursos seleccionados fueron los siguientes 3:

**Posts**

**Comments**

**Albums**

Luego cree en un Excel con 3 tablas para los casos de pruebas de los respectivos endopoints elegidos.

Para el endpoint **Posts** realicé 8 casos de pruebas de los cuales los primeros 5 fueron utilizando el método GET en donde describí los escenarios positivos que dan como respuesta un status code 200 OK y una estructura de los posts válida y también validé que el tiempo de respuesta sea menor a 500ms. Luego describí uno de los posibles escenarios positivos en donde se valida que trae correctamente un post en específico. También tuve en cuenta el escenario negativo en donde se espera como respuesta un 404 Not Found cuando se escribe una URL incorrecta.

Luego utilicé para los 3 casos de pruebas restantes el método POST, PUT y DELETE, para validar la creación correcta de un nuevo post, la actualización correcta de un post en específico y la eliminación de un post en específico.

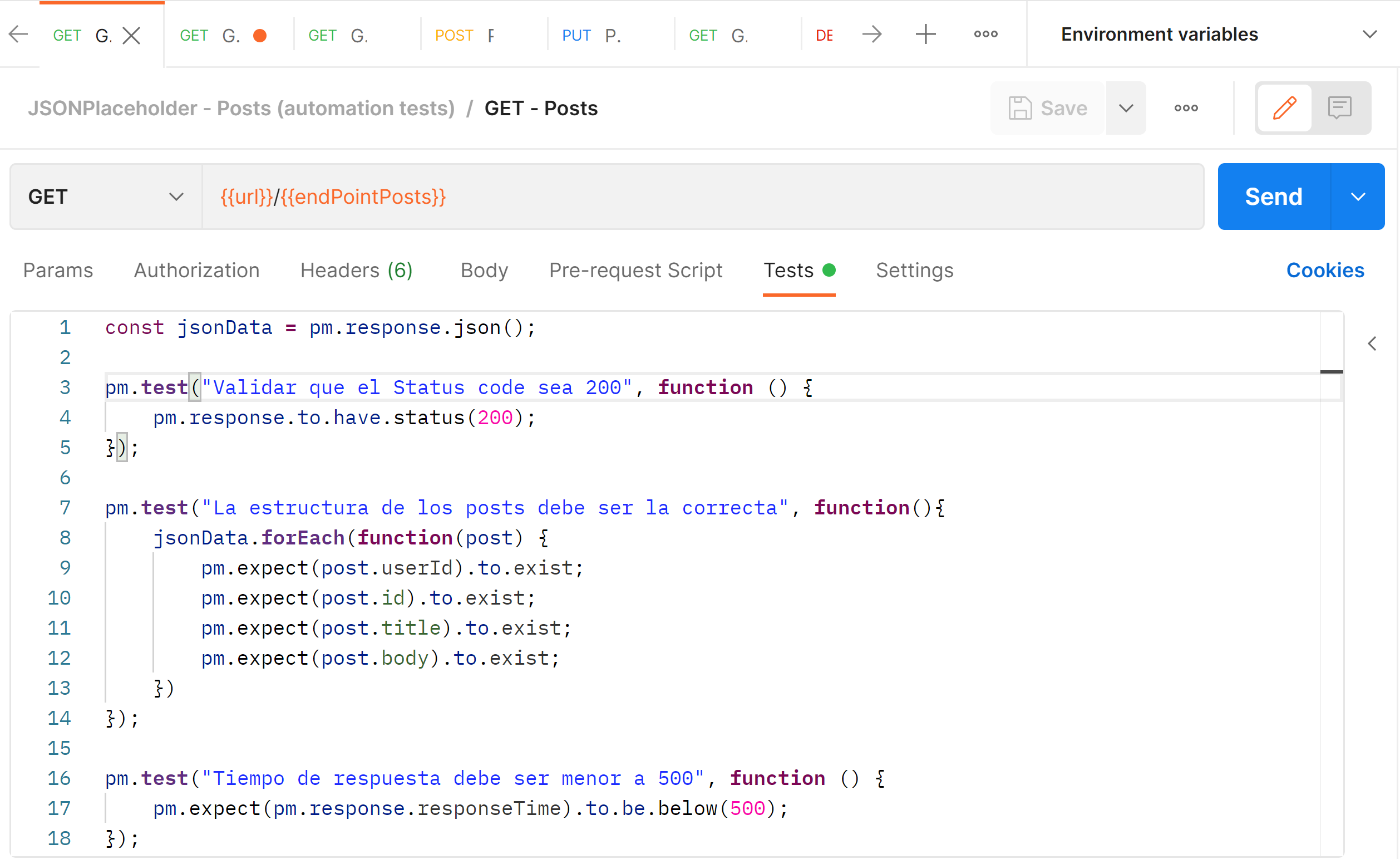
Para los endopint **Comments** y **Album** realicé los mismos primeros 5 test con el método Get que cree para Post y les agregué un caso de prueba para ambos utilizando el método POST para validar la correcta creación de un comment y un almbum.

Todos estos casos de prueba los ejecuté en Postman creando una colección para cada endpoint y adentro los diferentes requests. También cree las variables de entorno correspondientes.

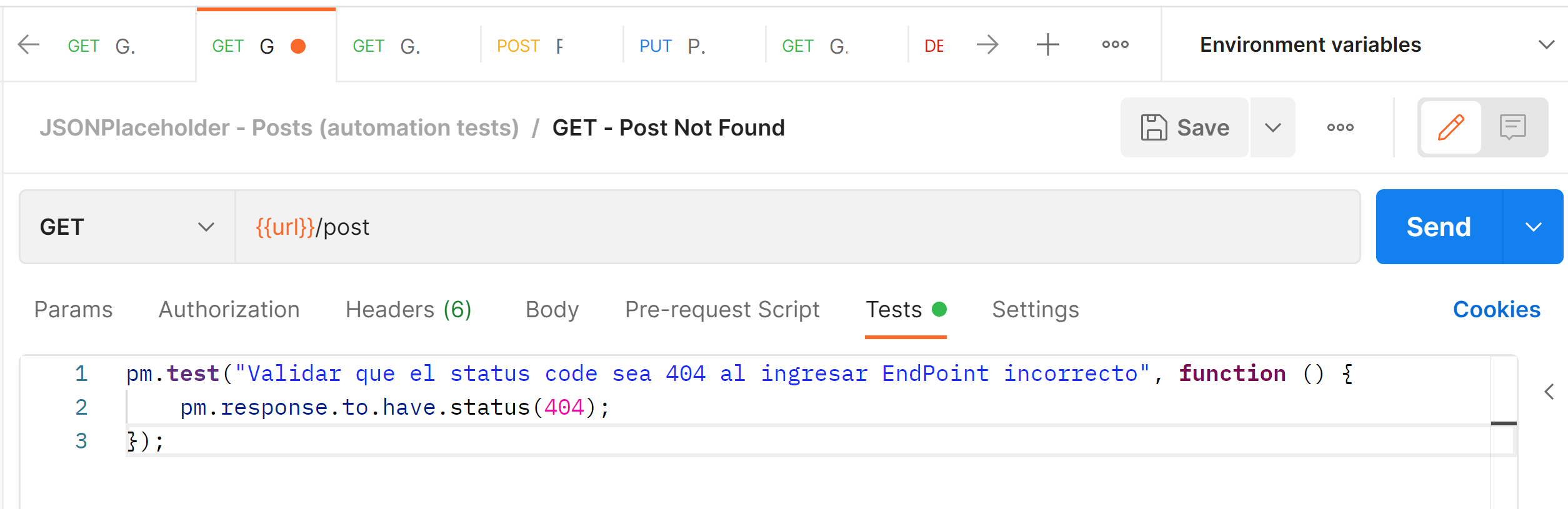
Finalmente automaticé los 8 casos de prueba del endopoint **Post**, para automatizar los casos de pruebas utilicé el lenguaje JavaScript dentro de Postman en la pestaña Test.

Cree 8 Scripts automatizados:

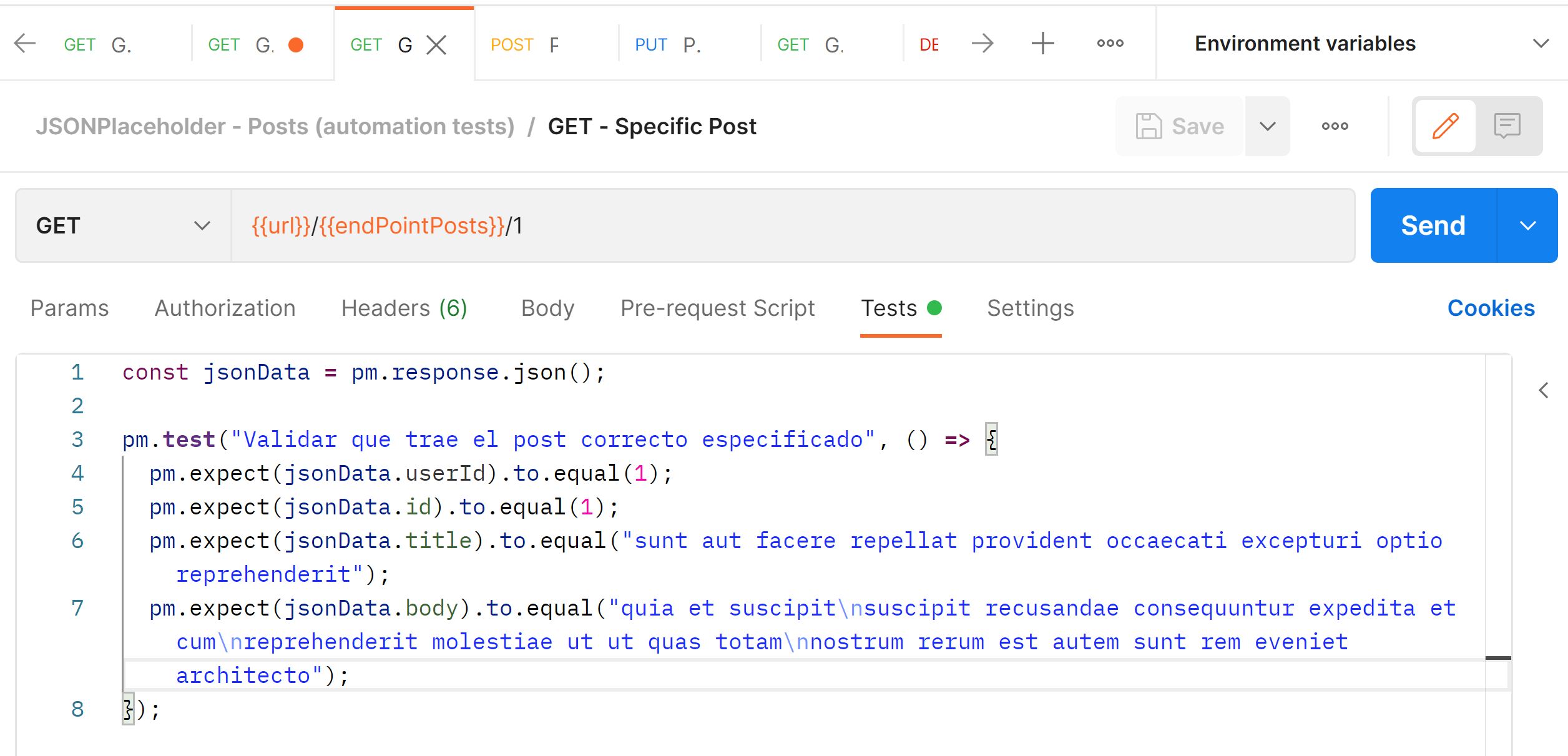
1. Status
2. La estructura del objeto
3. Tiempo de respuesta



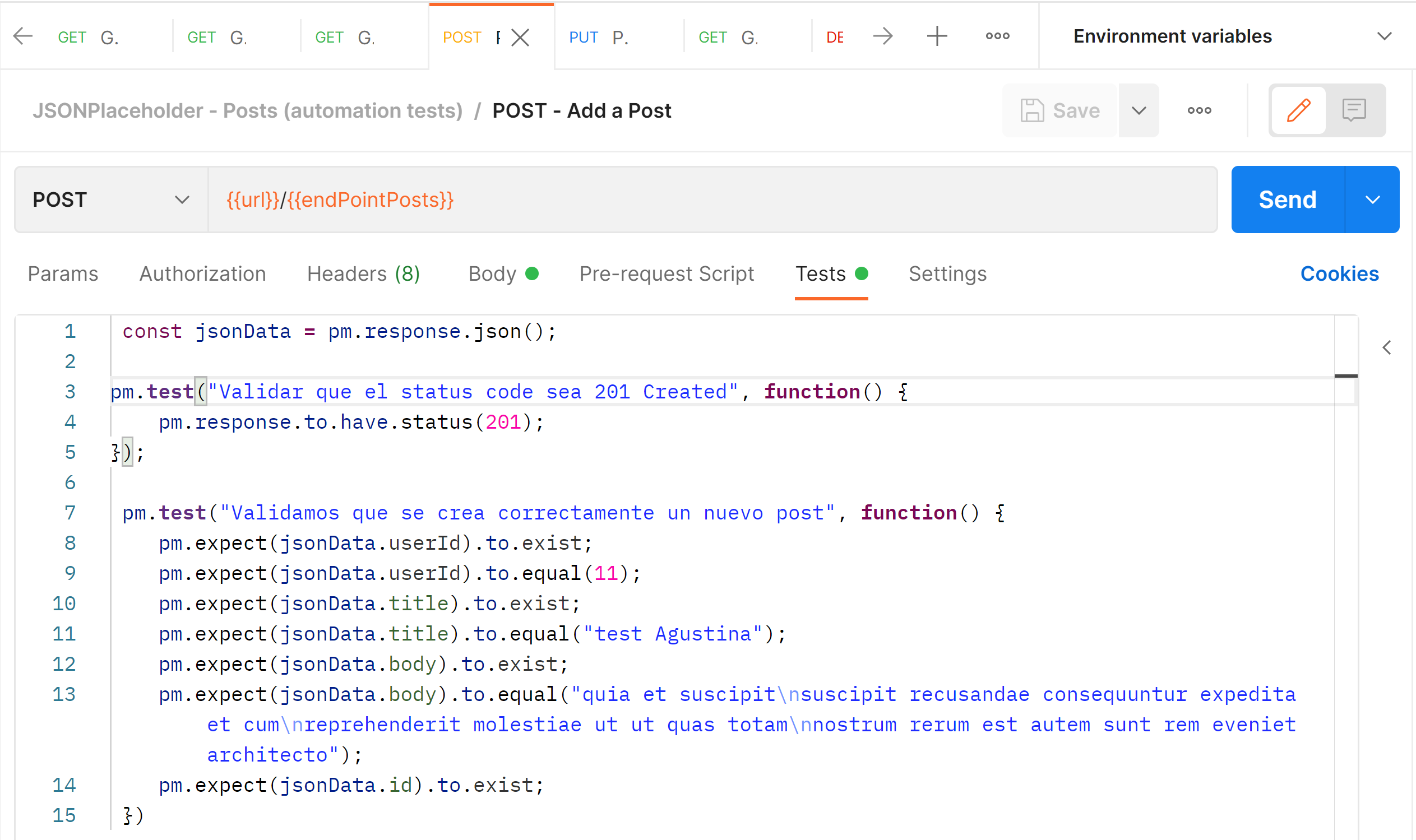
1. EndPoint incorrecto 404 Not Found



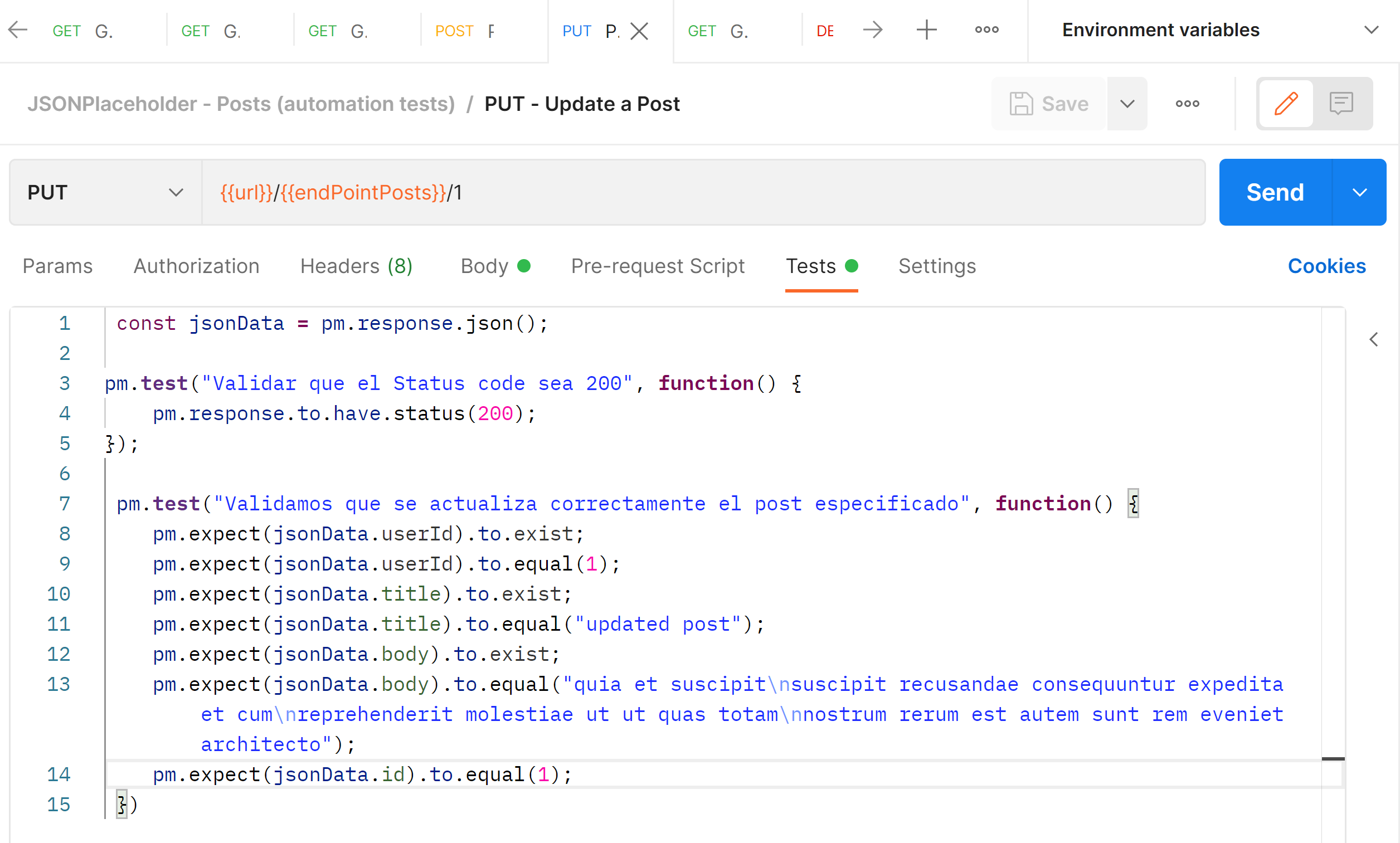
1. Objeto en específico



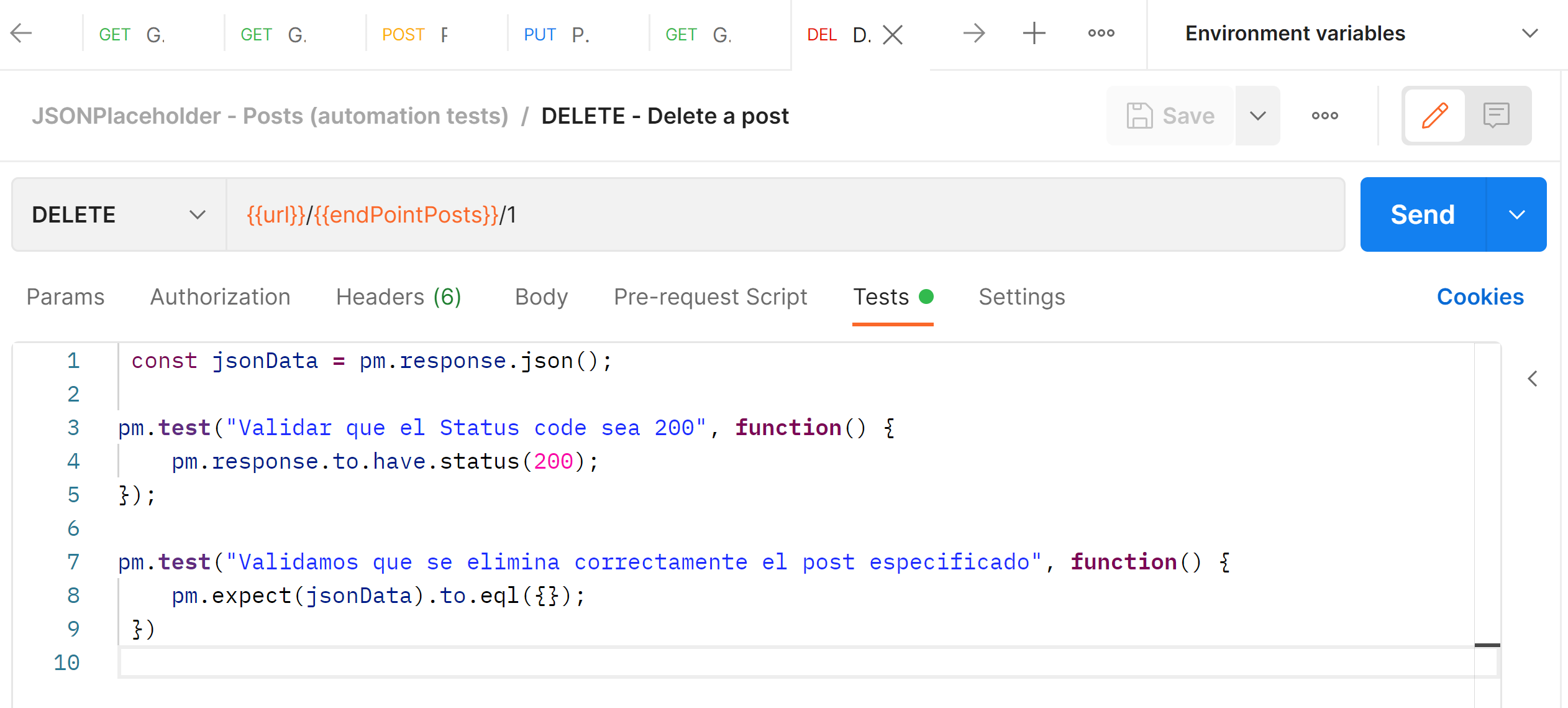
1. Agregar un post (se comprueba que el id se genere automáticamente verificando que exista)



1. Actualizar un post



1. Eliminar un post



Luego ejecuté todos los test de los request de la colección JSONPlaceholder – Posts (automation tests) y pude observar los resultados:

