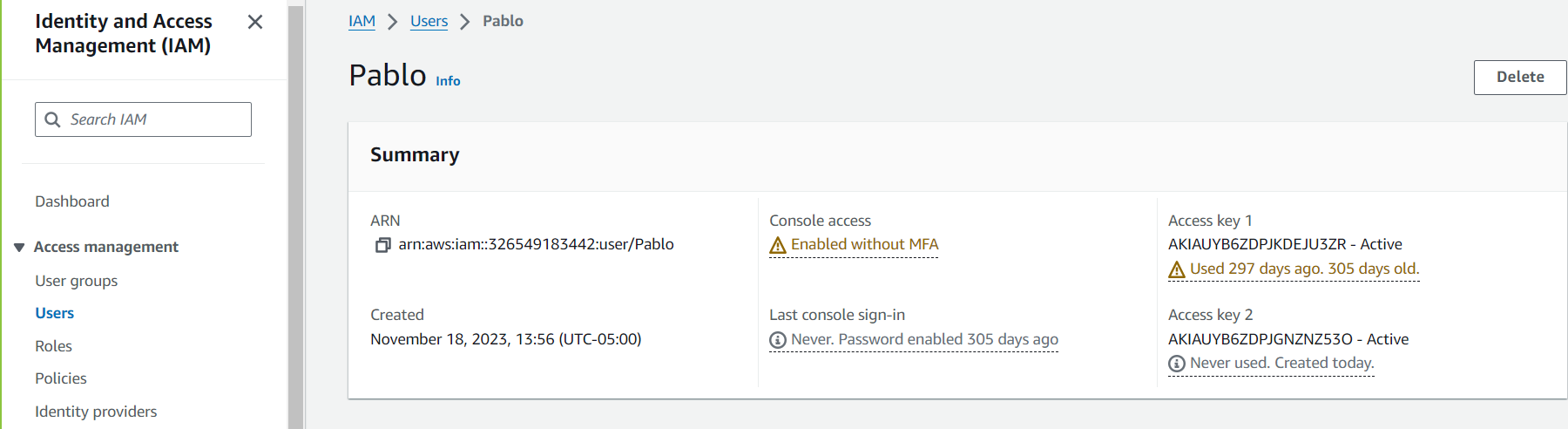
**Configuración**

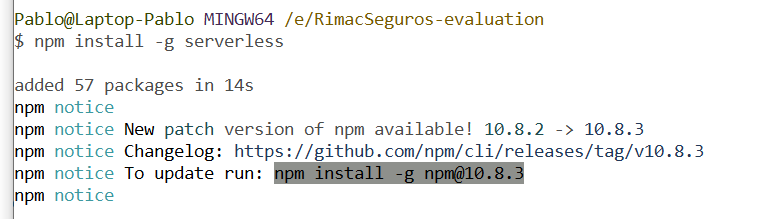
Se debe tener un usuario de aws para el acceso y permisos al despliegue del proyecto.



Configurar el acceso a aws desde el ordenador local. Se debe crear el profile “serverlessUser” para que serverless tenga la conexión hacia aws.

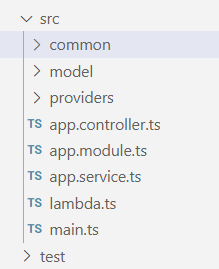


Instalar la librería serverless de manera global.

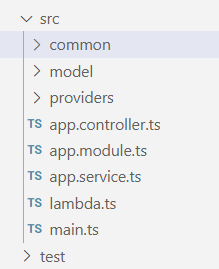


**Estructura del proyecto**

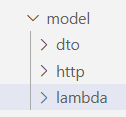
El proyecto se encuentra distribuido de la siguiente manera:



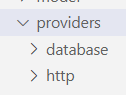
La carpeta common: Contiene las funcionalidades extras como manejo de excepción, validaciones, alguna funcionalidad utilitaria, etc.



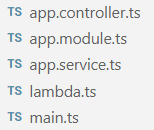
La carpeta model: Contiene los modelos de datos que permite el transporte de datos a lo largo del proceso de ejecución.



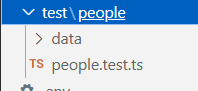
La carpeta providers: Contiene la funcionalidad de acceso a fuentes externas. En este caso uso de api y dase datos.



Las carpetas principales: Contiene los archivos necesarios de acuerdo a la estructura nestjs. Adicionalmente, el archivo lambda.ts contiene la inicialización del lambda aws. El archivo main.ts contiene la configuración del swagger.

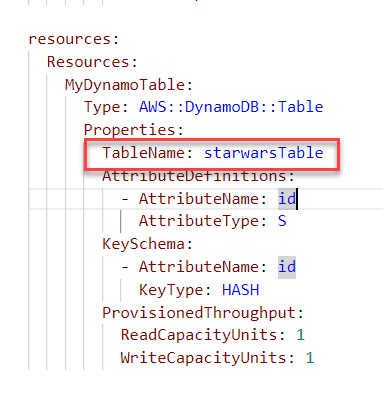


La carpeta test: Contiene los archivos test para las pruebas unitarias.



El archivo serverless.yml: contiene el código para la construcción de los recursos aws a través de cloudformation. Tomar en cuenta las variables de entornos.

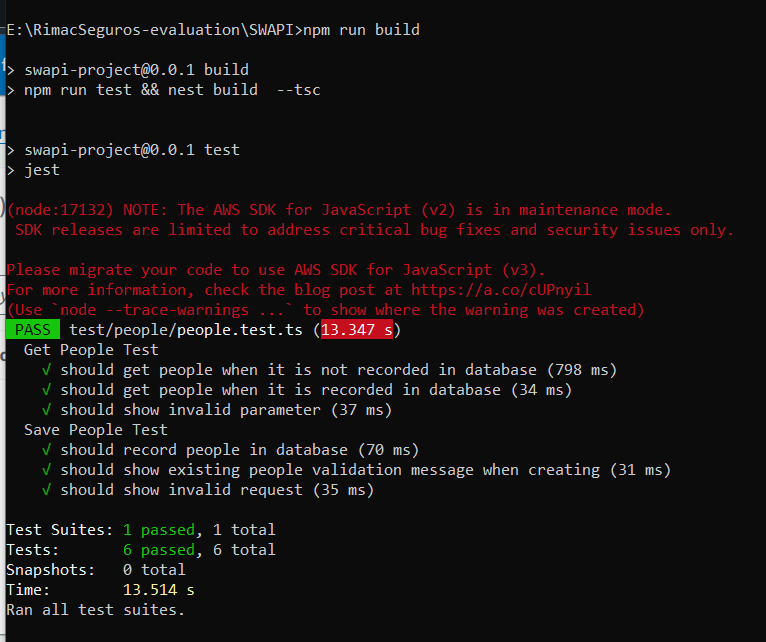




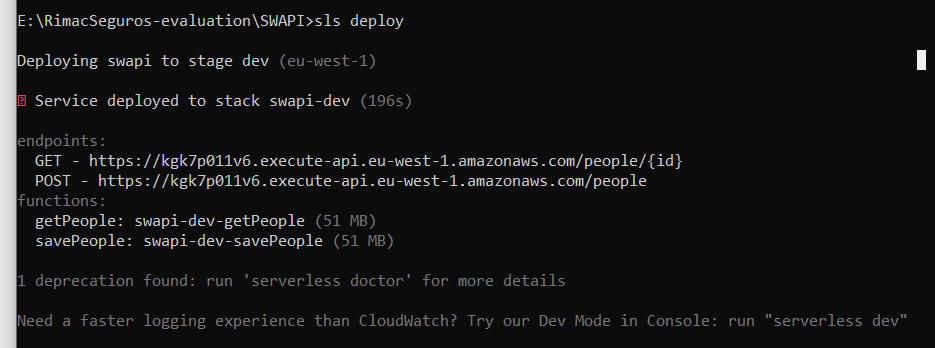
**Publicación del proyecto**

Descargar el proyecto del repositorio github <https://github.com/pccapa/SWAPI> y ejecutar el comando “npm install” para descargar las librerías.

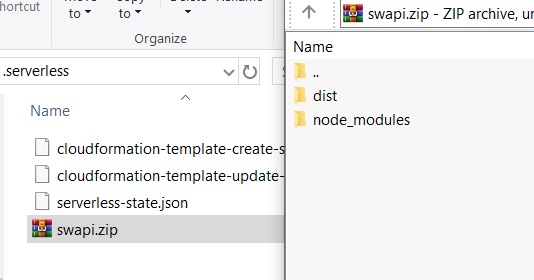
Seguidamente ejecutar el comando “npm run build”. Ello ejecuta los test unitarios y realiza la compilación del proyecto.



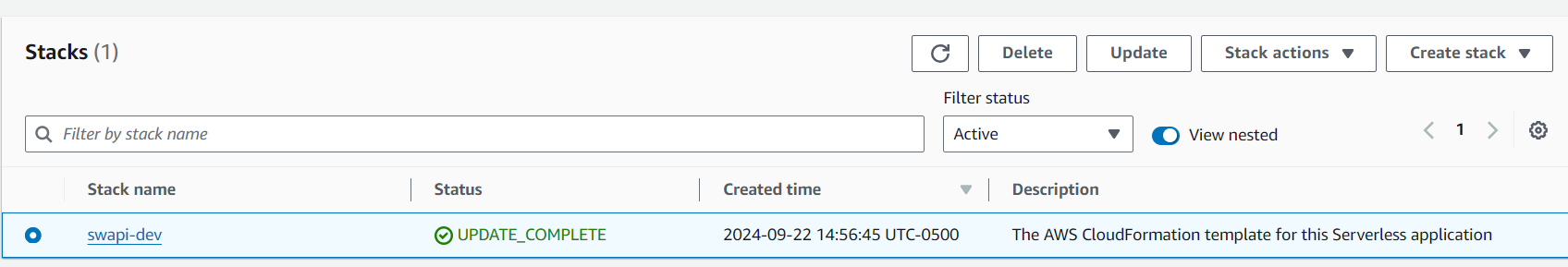
Luego ejecutamos el comando “sls deploy” y se genera la infraestructura en aws mostrando los endpoints.

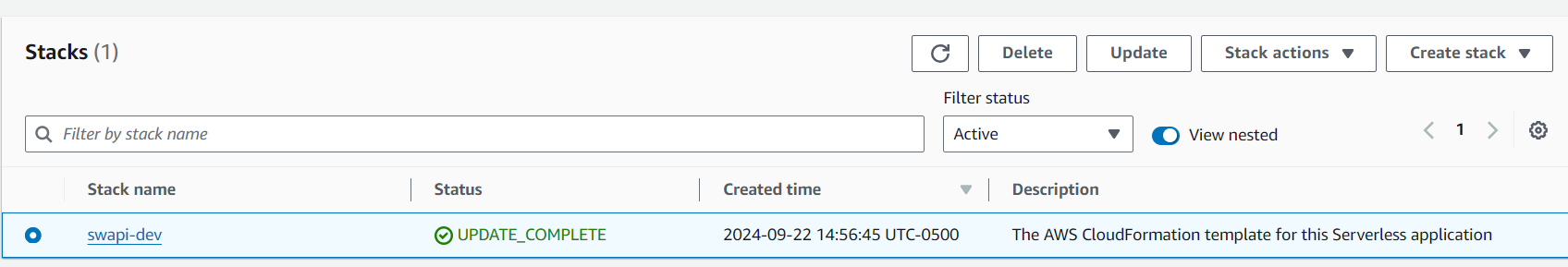


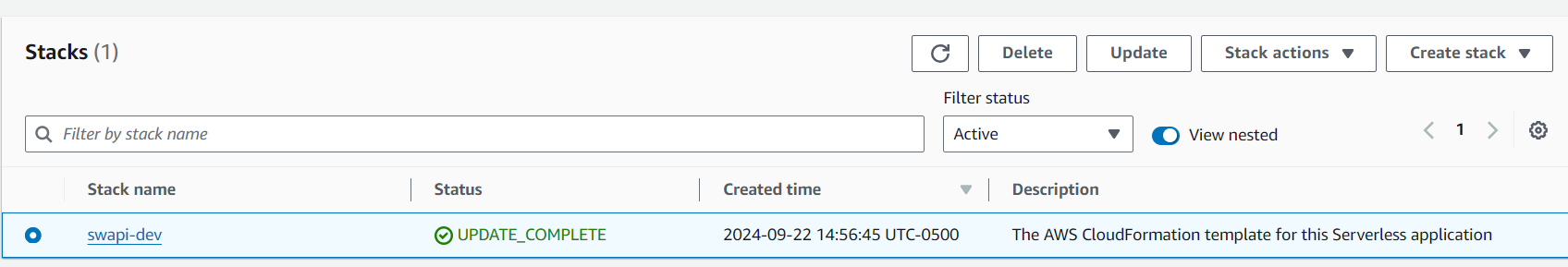
Dentro del código fuente podemos ver que en la carpeta .serverless se generó un archivo zipeado donde contiene los compilados.

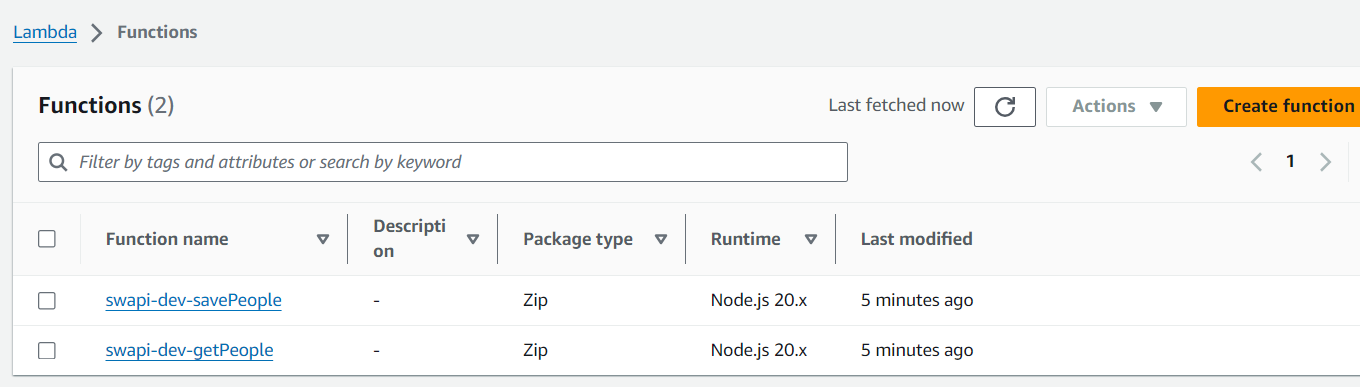


En el cloudformation del aws se generó el stack infraestructura.





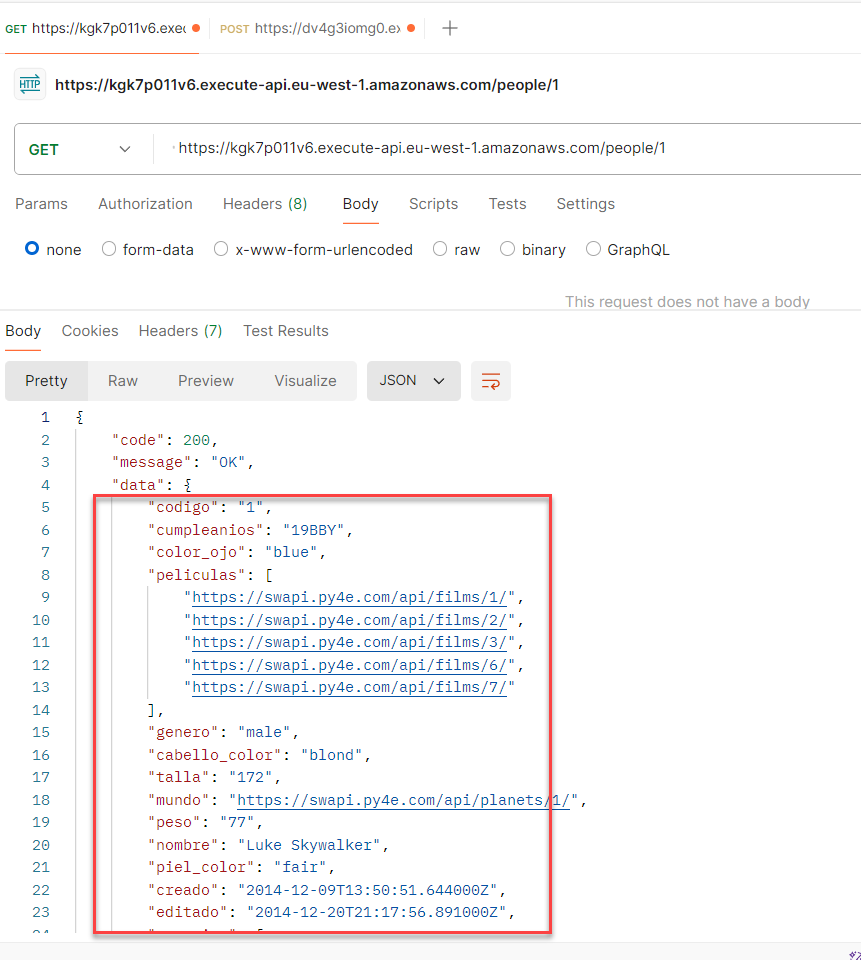


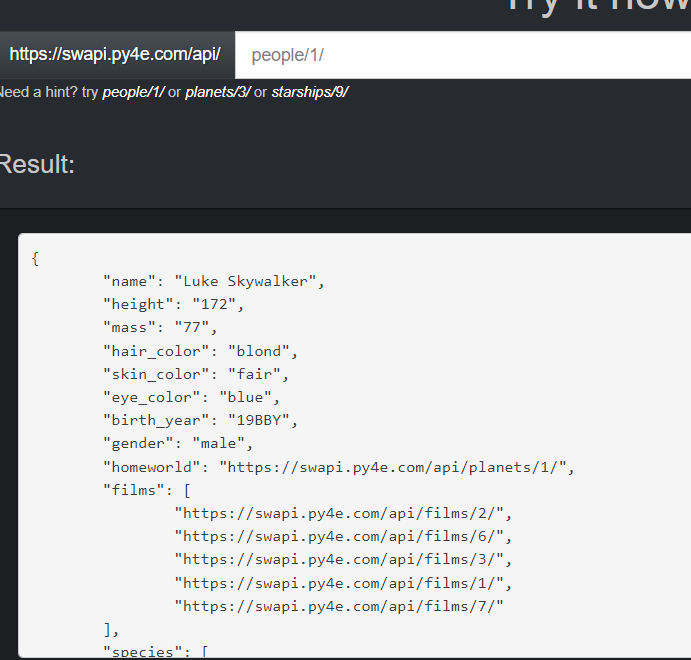


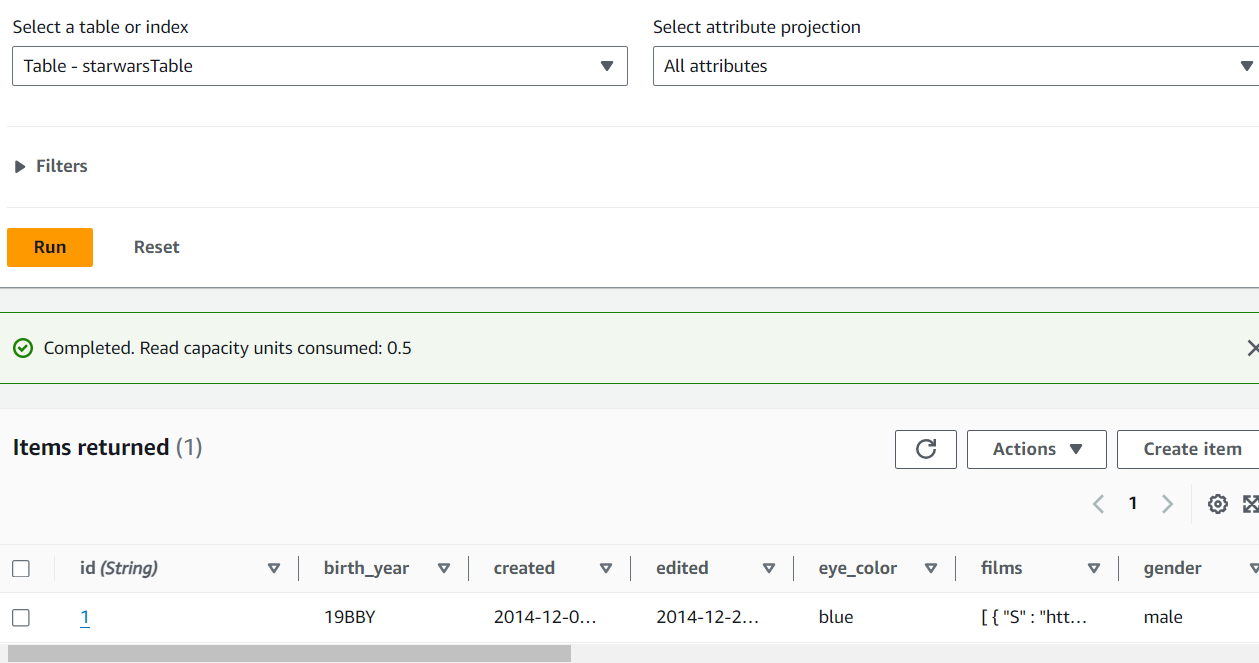
Entonces, ejecutamos el endpoint “persona” [https://kgk7p011v6.execute-api.eu-west-1.amazonaws.com/people/{id}](https://kgk7p011v6.execute-api.eu-west-1.amazonaws.com/people/%7bid%7d)

Internamente verifica si en la base de datos existe el registro con el código 1. Si existe en la base de datos, devuelve el registro. De lo contrario, obtiene el registro desde el api de SWAPI, graba en la base de datos y luego es mostrado en el response.

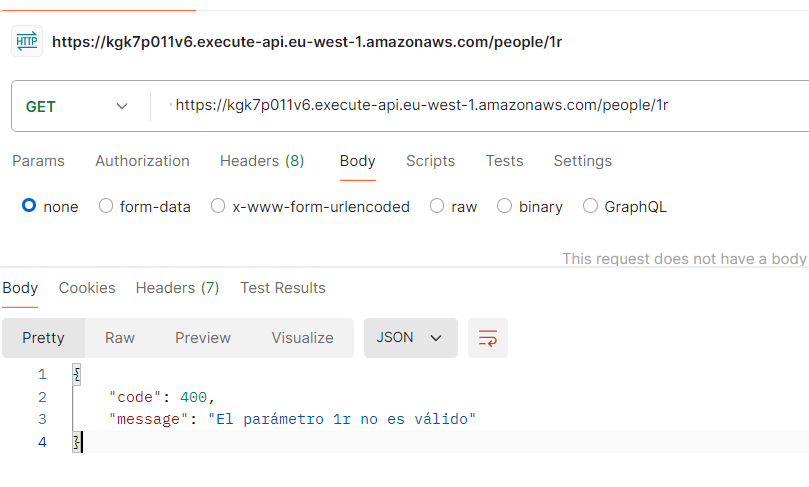
Los campos del api SWAPI son guardados en ingles en la base de datos, pero en la visualización para el usuario es mostrado en español.



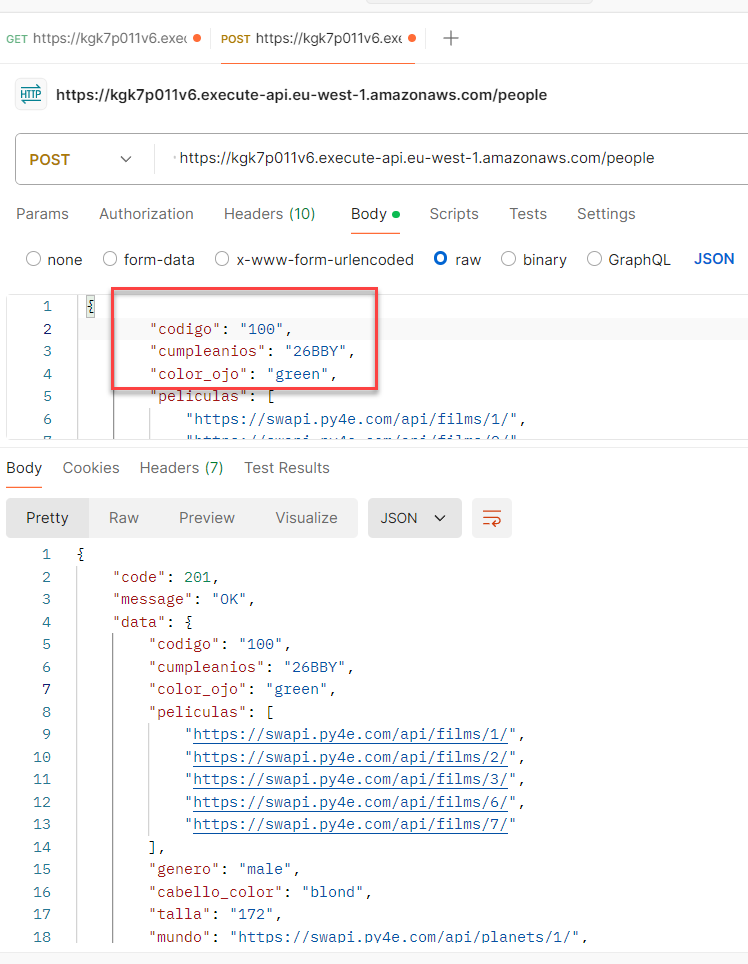


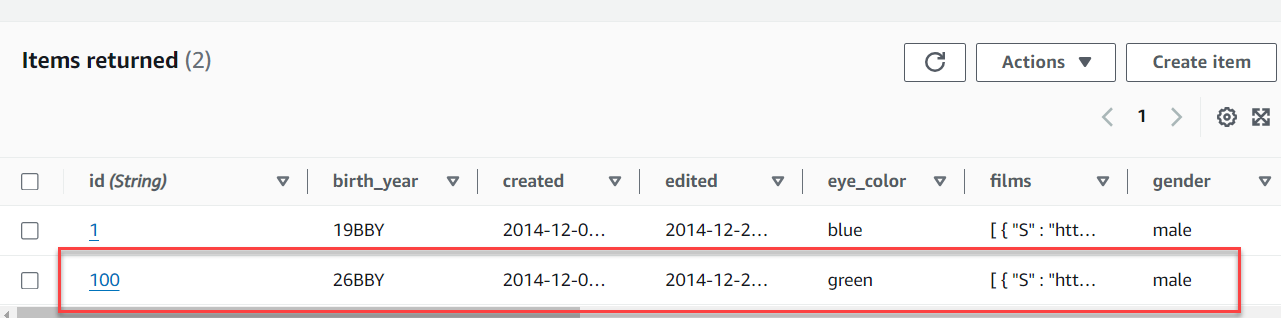


Si se ingresa mal el parámetro, muestra la validación correspondiente.

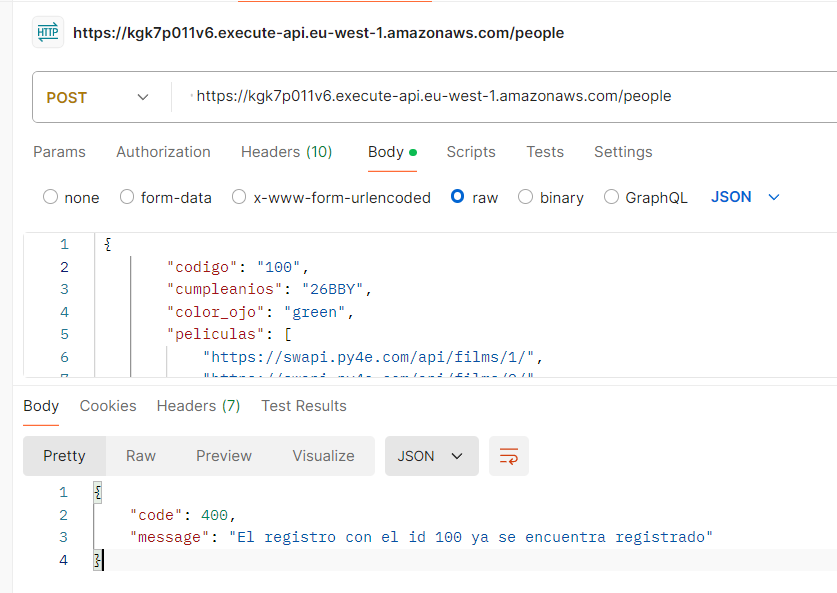


Ahora, con el endpoint <https://kgk7p011v6.execute-api.eu-west-1.amazonaws.com/people> se puede grabar un registro “persona” de la siguiente manera. Tomando en cuenta el registro obtenido anteriormente, podemos modificar su información y guardarlo. Internamente verifica si la información no existe en la base de datos, lo guarda.





Si la información existe en la base de datos (se valida con el codigo) muestra el mensaje de validación.



Por último, si el registro “persona” contiene una estructura incorrecta, muestra el siguiente mensaje de validación.

