

T3: EJ OpenAPI

jueves, 15 de mayo de 2025 22:35

Ejemplo de un doc OpenAPI:

```
openapi: 3.0.1
info:
  title: Book Store
  version: 0.0.1
  description: Orianna's books collection
paths:
  /books:
    get:
      summary: All Books
      description: Lists the books in the collection
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/books'
              examples:
                objectExample:
                  $ref: '#/components/examples/books'
        '400':
          description: Error fetching the books.
    post:
      summary: Add a book
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/books'
      responses:
        '201':
          description: OK
        '400':
          description: Error in the request.
  /books/{id}:
    parameters:
      - $ref: '#/components/parameters/id'
    get:
      summary: Get a book
      description: List an specific book
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/book'
        '404':
          description: Invalid ID
```

```
    put:
      summary: Update a book
      description: ''
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/book'
      responses:
        '200':
          description: OK
        '400':
          description: Invalid request.
    delete:
      summary: null
      description: ''
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/book'
      responses:
        '200':
          description: OK
        '400':
          description: Invalid request.
servers:
  - url: http://localhost:3000/
components:
  parameters:
    id:
      description: Book's identifier
      name: book id
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/id'
  schemas:
    id:
      type: string
      format: uuid
      description: a unique book identifier
    book:
      type: object
      properties:
        _id:
          type: string
          format: uuid
          description: a unique book identifier
        title:
          type: string
          description: book's name
```

```
  components:
    parameters:
      id:
        description: Book's identifier
        name: book id
        in: path
        required: true
        schema:
          $ref: '#/components/schemas/id'
    schemas:
      id:
        type: string
        format: uuid
        description: a unique book identifier
      book:
        type: object
        properties:
          _id:
            type: string
            format: uuid
            description: a unique book identifier
          title:
            type: string
            description: book's name
          author:
            type: string
            description: book's writer
          pages:
            type: number
            description: book's pages
          chapters:
            type: array
            items:
              type: string
            required:
              - _id
              - title
              - author
      books:
        type: object
        properties:
          _id:
            type: string
            format: uuid
            description: a unique book identifier
          title:
            type: string
            description: book's name
          author:
            type: string
            description: book's writer
          required:
            - _id
            - title
            - author
    examples:
      books:
        _id: 150e8400-e29b-41d4-a716-446655440009
        title: The Martian
        author: Andy Weir
```

OpenAPI – Ejercicio

- Genera el documento OpenAPI para el ejercicio del [cine](#)



Ejercicio (cont.)

- Diseña un servicio web basado en REST
- Define las diferentes rutas que necesitamos
- Piensa en las diferentes acciones que se pueden realizar
- Piensa en ejemplos de mensaje y los campos que deberían tener
- Considera los códigos de estado

CEU | Universidad San Pablo | REST | 66

```

! cineOpenAPI.yaml cineOpenAPI.yaml
1  openapi: 3.0.0
2  info:
3    title: Cine
4    description: API de películas y sesiones
5    version: 0.0.1
6  paths:
7    /películas:
8      get:
9        summary: Lista Películas
10       description: Lista todas las películas disponibles
11       content:
12         application/json:
13           schema:
14             $ref: '#/components/schemas/películas'
15       responses:
16         "200":
17           description: OK.
18         "400":
19           description: Error. Solicitud Invalida.
20         "404":
21           description: Error. No encontrado.
22     post:
23       summary: Agrega Películas
24       description: Agrega una Película a la lista.
25       requestBody:
26         required: true
27         content:
28           application/json:
29             schema:
30               $ref: '#/components/schemas/película'
31             examples:
32               example:
33                 $ref: '#/components/examples/película'
34       responses:
35         "201":
36           description: Película Agregada
37         "400":
38           description: Error. Solicitud Invalida.
39         "409":
40           description: Error.
41     /películas/{id}:
42       parameters:
43         schema:
44           $ref: '#/components/schemas/id'
45       put:
46         summary: Actualizar una película
47         requestBody:
48           required: true
49           content:
50             application/json:
51               schema:
52                 $ref: '#/components/schemas/película'
53               exampleObject:
54                 $ref: '#/components/examples/película'
55       responses:
56         "201":
57           description: Película Actualizada
58         "400":
59           description: Error. Solicitud Invalida.
60         "404":
61           description: Error.

```

```

! cineOpenAPI.yaml cineOpenAPI.yaml
6  paths:
41    /películas/{id}:
45      put:
55        responses:
58          "400":
60            description: Error.
61      delete:
62        summary: Borrar una película
63        requestBody:
64          required: true
65        content:
66          application/json:
67            schema:
68              $ref: '#/components/schemas/película'
69            exampleObject:
70              $ref: '#/components/examples/película'
71        responses:
72          "204":
73            description: OK
74          "404":
75            description: Error.
76  servers:
77    - url: 'http://localhost:3000/api/cine'
78  components:
79    schemas:
80      id:
81        type: string
82        format: uuid
83        description: Identificador único de películas
84      películas:
85        type: array
86        description: Una lista de películas
87        items:
88          $ref: '#/components/schemas/película'
89      película:
90        type: object
91        properties:
92          id:
93            $ref: '#/components/schemas/id'
94          title:
95            type: string
96          duration:
97            type: number
98            description: Durations in minutes
99          director:
100            type: string
101    examples:
102      película:
103        value:
104          id: f81d4fae-7dec-11d0-a765-00a0c91e6bf6
105          title: Capitan América 2
106          duration: 140
107          director: Los hermanos Russo
108

```