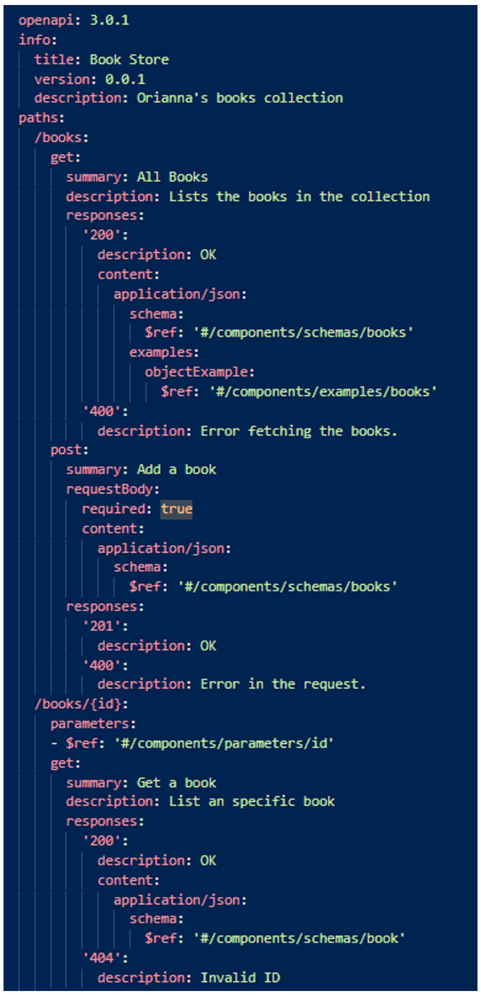
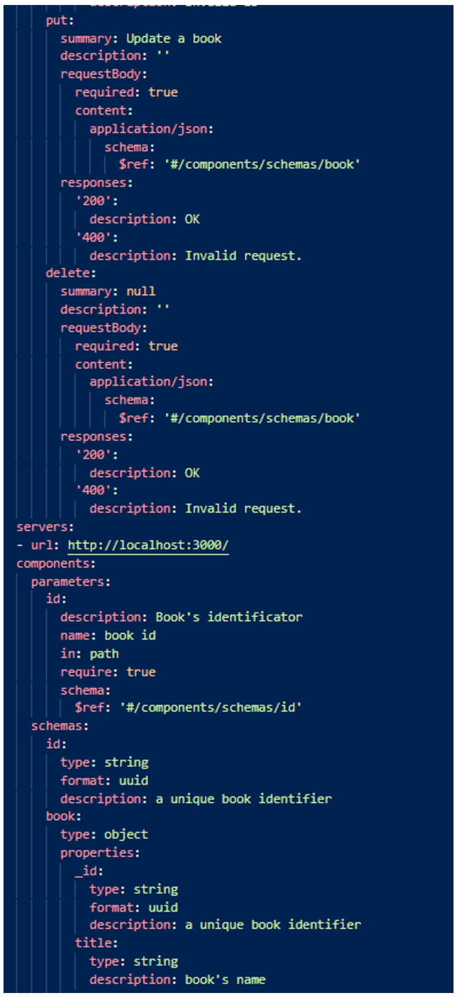
T3: EJ OpenAPI

jueves, 15 de mayo de 2025 22:35

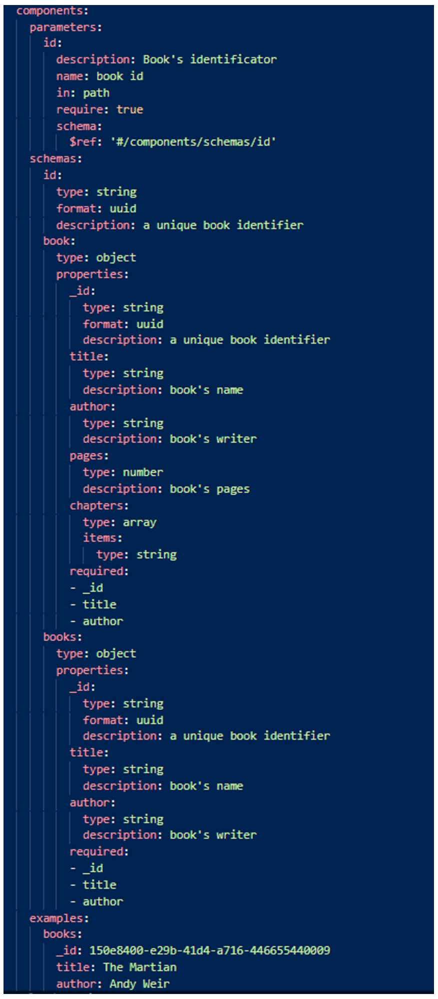
Ejemplo de un doc OpenAPI:



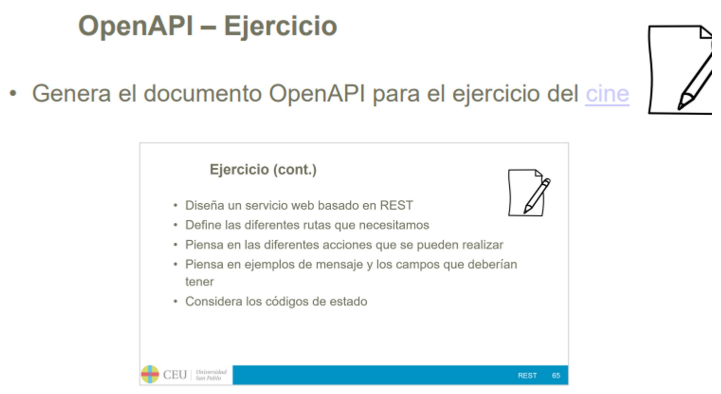
```yaml
openapi: 3.0.1
info:
  title: Book Store
  version: 0.0.1
  description: Orianna's books collection
paths:
  /books:
    get:
      summary: All Books
      description: Lists the books in the collection
      responses:
        '200':
          description: OK
          content:
            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
```



```yaml
    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 identificator
      name: book id
      in: path
      require: 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
```



```yaml
components:
  parameters:
    id:
      description: Book's identificator
      name: book id
      in: path
      require: 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
```



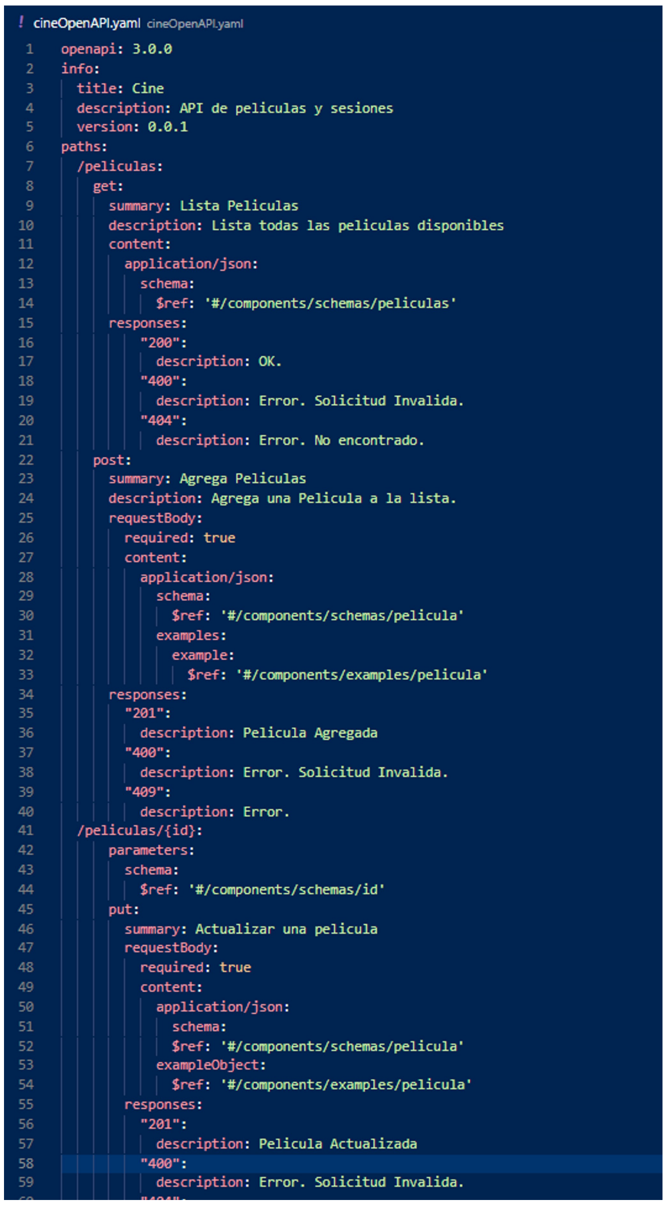
OpenAPI – Ejercicio

- Genera el documento OpenAPI para el ejercicio del cine

Ejercicio (cont.)

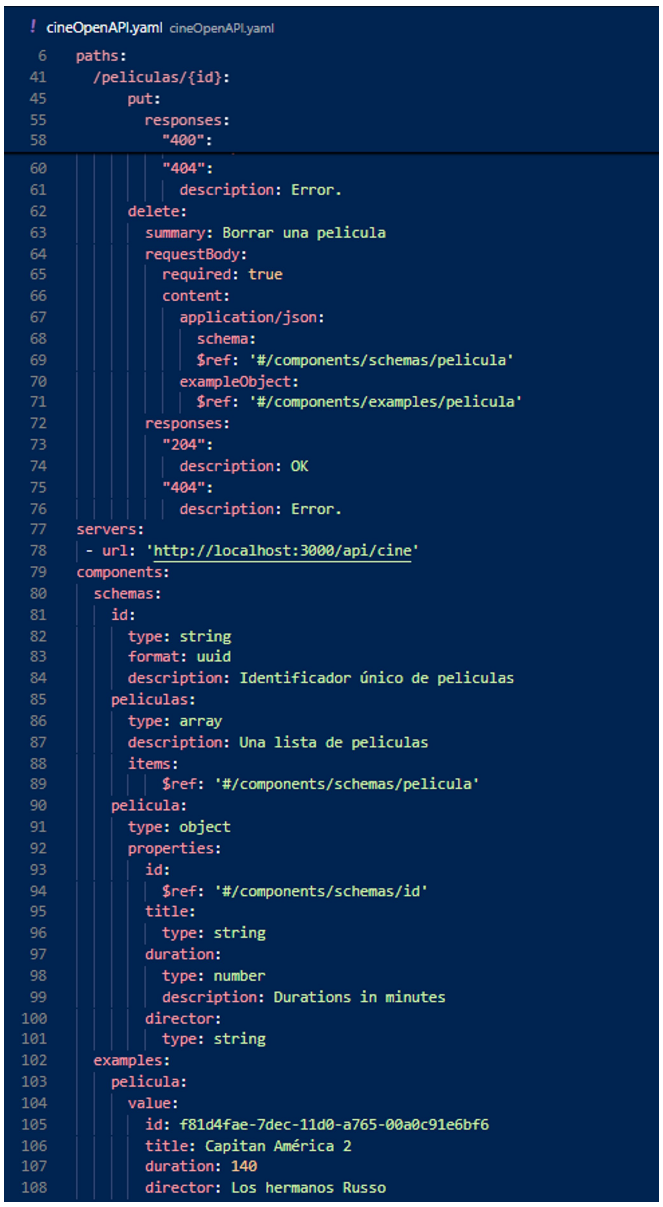
- Diseña un servicio web basado en REST
- Define las diferentes rutas que necesitamos
- Piensa en las diferentes acciones que se pueden realizar
- Piensa en ejemplos de mensaje y los campos que deberían tener
- Considera los códigos de estado

CEU Universidad San Pablo REST 65



! cineOpenAPI.yaml cineOpenAPI.yaml

```yaml
openapi: 3.0.0
info:
  title: Cine
  description: API de peliculas y sesiones
  version: 0.0.1
paths:
  /peliculas:
    get:
      summary: Lista Peliculas
      description: Lista todas las peliculas disponibles
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/peliculas'
      responses:
          "200":
            description: OK.
          "400":
            description: Error. Solicitud Invalida.
          "404":
            description: Error. No encontrado.
    post:
      summary: Agrega Peliculas
      description: Agrega una Pelicula a la lista.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/pelicula'
            examples:
              example:
                $ref: '#/components/examples/pelicula'
      responses:
        "201":
          description: Pelicula Agregada
        "400":
          description: Error. Solicitud Invalida.
        "409":
          description: Error.
  /peliculas/{id}:
      parameters:
        schema:
          $ref: '#/components/schemas/id'
      put:
        summary: Actualizar una pelicula
        requestBody:
          required: true
          content:
            application/json:
              schema:
              $ref: '#/components/schemas/pelicula'
            exampleObject:
              $ref: '#/components/examples/pelicula'
        responses:
          "201":
            description: Pelicula Actualizada
          "400":
            description: Error. Solicitud Invalida.
```



! cineOpenAPI.yaml cineOpenAPI.yaml

```yaml
paths:
  /peliculas/{id}:
      put:
        responses:
          "400":
          "404":
            description: Error.
      delete:
        summary: Borrar una pelicula
        requestBody:
          required: true
          content:
            application/json:
              schema:
              $ref: '#/components/schemas/pelicula'
            exampleObject:
              $ref: '#/components/examples/pelicula'
        responses:
          "204":
            description: OK
          "404":
            description: Error.
servers:
  - url: 'http://localhost:3000/api/cine'
components:
  schemas:
    id:
      type: string
      format: uuid
      description: Identificador único de peliculas
    peliculas:
      type: array
      description: Una lista de peliculas
      items:
          $ref: '#/components/schemas/pelicula'
    pelicula:
      type: object
      properties:
        id:
          $ref: '#/components/schemas/id'
        title:
          type: string
        duration:
          type: number
          description: Durations in minutes
        director:
          type: string
  examples:
    pelicula:
      value:
        id: f81d4fae-7dec-11d0-a765-00a0c91e6bf6
        title: Capitan América 2
        duration: 140
        director: Los hermanos Russo
```