UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS
APLICACIONES OPEN SOURCE (SI729)
2023-1

Se le encarga el desarrollo de una API REST que permita las siguientes funcionalidades.

Requisitos funcionales

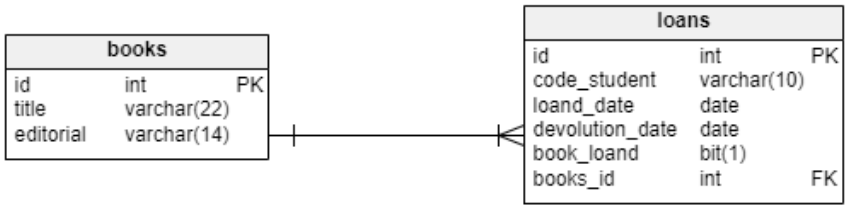
- Registro de información
  - Registro de libros (POST)
  - Registrar préstamo de libro. (POST)
- Listados de información
  - Listado de todos los libros (GET)
  - Listado de libros por editorial (GET)
  - Listado de préstamos por código de alumno (GET)

Atributos de las clases / Columnas de las tablas

La relación es de uno a muchos desde Book (Libro) a Loan (Préstamo)

Nota: bookLoan atributo que permite saber si el libro se prestó (true)

Clases	Tablas
Book	books
====	=========
private Long id	id
private String title	title
private String editorial	editorial
Loan	loans
====	=====
private Long id	id
private String codeStudent	code_student
private LoalDate loanDate	loan_date
private LoalDate devolutionDate	devolution_date
private boolean bookLoan	book_loan
private Book book	book_id



Reglas de negocio.

Book

====

- Validar que los atributos del book cumplan las siguientes restricciones
    - Cuando no se ingrese un title se debe mostrar el mensaje “El título del libro debe ser obligatorio”
    - Cuando se ingrese un title que excede los 22 caracteres se debe mostrar el mensaje “El título del libro no debe exceder los 22 caracteres”
    - Cuando no se ingrese una editorial se debe mostrar el mensaje “La editorial del libro debe ser obligatorio”
    - Cuando se ingrese una editorial que excede los 14 caracteres se debe mostrar el mensaje “La editorial del libro no debe exceder los 14 caracteres”.

- No se debe permitir el registro de un book con el mismo title y editorial
    - Cuando se trate de registrar un book con un title y editorial que ya existe se debe mostrar el siguiente mensaje “No se puede registrar el libro porque existe uno con el mismo título y editorial”.

Loan

====

- Para realizar el registro de préstamo de libros solo se debe enviar el codeStudent y book (id). El valor de loanDate y devolutionDate se obtiene del sistema. En el caso de bookLoan debe tener por defecto un valor de true.
- No se debe permitir el registro de un préstamo de libro con el mismo codeStudent book(id) y que bookLoan sea true. En caso se intente registrar un prestamos que no cumpla esta regla de negocio se debe mostrar el mensaje “El préstamo del libro con código XXXX no es posible porque ya fue prestado por el alumno YYYY”. Donde XXXX debe reemplazarlo por el title del libro y YYYY por el código del alumno.
- Validar que los atributos del Loan cumplan las siguientes restricciones

- Cuando no se ingrese un codeStudent se debe mostrar el mensaje “El código del alumno debe ser obligatorio”
- Cuando se ingrese un codeStudent menor de 10 caracteres se debe mostrar el mensaje “El código del alumno debe tener 10 caracteres”

Requisitos no funcionales

- Puede utilizar Postman o Swagger para realizar la prueba del API REST.
- Debe crear clases personalizadas de excepciones para los siguientes casos
  - Recurso no encontrado: ResourceNotFoundException
  - Validación de atributos: ValidationException
  - Gestor de excepciones (Handler): ControllerExceptionHandler
  - La información del error debe ser representada mediante una clase de nombre ErrorMessage, el cual debe tener la siguiente información:
    private int statusCode;
    private String message;

    private String description;
    private LocalDateTime timestamp;

- El nombre de los endpoints (url) a utilizar en los controllers debe ser
  - La ruta base de los endpoints debe ser /api/library/v1
  - La ruta para el registro de libros debe ser /api/library/v1/books
  - La ruta para el listado de libros debe ser /api/library/v1/books
  - La ruta para el listado de libros por editorial debe ser /api/library/v1/books/ filterByEditorial
  - La ruta para el registro de prestamo de libro /api/library/v1/books/{id}/loans
  - La ruta para el listado de préstamos por código de alumno /api/library/v1/books/filterByCodeStudent

1. Registro de libros (POST)

Escenario 1: registro de libro exitoso

Request: http://localhost:8080/api/library/v1/books

```
{
   "title":"Spring Data JPA",
   "editorial": "Baeldung"
}
```

Response:

```
{
  "id": 1,
   "title": "Spring Data JPA",
   "editorial": "Baeldung"
}
```

Escenario 2: registro de un libro con un title y editorial que ya existe

Request: http://localhost:8080/api/library/v1/books

```
{
   "title":"Spring Data JPA",
   "editorial": "Baeldung"
}
```

Response:

```
{
   "statusCode": 400,
   "message": "No se puede registrar el libro porque existe uno con el mismo título y
              editorial",
   "description": "uri=/api/library/v1/books",
   "timestamp": "29-11-2022 05:35:40"
```

}

Escenario 3: registro de un libro sin enviar un valor de title o editorial

Request: http://localhost:8080/api/library/v1/books

{

"title":"",

"editorial": "Baeldung"

}

{

"title":"Spring Data JPA ",

"editorial": ""

}

Response

{

"statusCode": 400,

"message": "El título del libro debe ser obligatorio",

"description": "uri=/api/library/v1/books",

"timestamp": "29-11-2022 05:38:03"

}

{

"statusCode": 400,

"message": "La editorial del libro debe ser obligatorio",

"description": "uri=/api/library/v1/books",

"timestamp": "29-11-2022 05:43:48"

}

2. Registro de préstamo de libro (POST)

Escenario 1: registro de préstamo de libro exitoso

Request: http://localhost:8080/api/library/v1/books/1/loans

{

"codeStudent":"2022222333",

"book":{

"id":1

}

}

Response:

{

"id": 4,

"codeStudent": "2022222333",

"loanDate": "2022-11-29",

"devolutionDate": "2022-12-02",

"bookLoan": true

}

Escenario 2: registro de préstamo de libro con un código de estudiante y id de book duplicado

Request: http://localhost:8080/api/library/v1/books/1/loans

{

"codeStudent":"2022222333",

"book":{

"id":1

}

}

Response

{

"statusCode": 400,

"message": "El prestamo del libro Spring Data JPA no es posible porque ya fue prestado por el alumno 2022222333",

"description": "uri=/api/library/v1/books/1/loans",

"timestamp": "29-11-2022 05:46:39"

}

Escenario 3: registro de préstamo de libro sin enviar valor de código de estudiante

Request: http://localhost:8080/api/library/v1/books/1/loans

```
{
  "codeStudent":"",
  "book":{
    "id":1
  }
}
```

Response

```
{
  "statusCode": 400,
```

```
    "message": "El código del alumno debe ser obligatorio",
    "description": "uri=/api/library/v1/books/1/loans",
    "timestamp": "29-11-2022 05:48:36"
}
```


3. Listado de libros (GET)
   Request: http://localhost:8080/api/library/v1/books

   Response


```
[
  {
    "id": 1,
    "title": "Spring Data JPA",
    "editorial": "Baeldung"
  },
  {
    "id": 2,
    "title": "Angular",
    "editorial": "Baeldung"
  },
  {
    "id": 3,
    "title": "Angular",
    "editorial": "Pivotal"
  }
]
```


4. Listado de libros por editorial (GET)
   Request:
   http://localhost:8080/api/library/v1/books/filterByEditorial?editorial=Baeldung

   Response


```
[
  {
    "id": 1,
    "title": "Spring Data JPA",
```

```
    "editorial": "Baeldung"
  },
  {
    "id": 2,
    "title": "Angular",
    "editorial": "Baeldung"
  }
]
```


5. *Listado de prestamos por código de alumno (GET)*
   Request:
   http://localhost:8080/api/library/v1/loans/filterByCodeStudent?codeStudent=2022222333

   Response


```
[
 {
    "id": 4,
    "codeStudent": "2022222333",
    "loanDate": "2022-11-29",
    "devolutionDate": "2022-12-02",
    "bookLoan": true
  },
  {
    "id": 5,
    "codeStudent": "2022222333",
    "loanDate": "2022-11-29",
    "devolutionDate": "2022-12-02",
    "bookLoan": true
  }
]
```