Books API

Used environement :

- Windows machine with Windows 10
- Cygwin for the console (php cli 7.1)
- HTTP Server : PHP built-in server
- DB server : Sqlite3 (pdo driver necessary)

Some php extensions are necessary or it won't work:

- mbstrings
- the sqlite and pdosqlite modules
- php-soap
- php-simplexml
- php-xml

Database location:

- The database is in **data/books.db**
- The dump is in **data/books.sql**

The Module is in:

/module/Book/

To launch the http server:

From the root folder: php -S 0.0.0.0:8080 -t public public/index.php

1 RESTful, JSON-RPC or SOAP-XML

NB : Initially there was only a Restful service, then I added the possibility to send JSON-RPC and SOAP-XML request.

Whether you use a service or another, only the Controller part changes slightly, the model(s) remain the same. All the client requests have been tested with Curl from the command line.

No HTML/PHP pages or views were used.

2 Test Requests

If you type the 2 following commands in a terminal, you'll get the same result:

Restful request:

```
curl -s  -H  "Accept: application/json"   -i -X GET
http://localhost:8080/book/from/1999/to/2008/minrating/1
```

JSON-RPC request:

```
curl -s -i -H "Accept: application/json" --request GET  --data '{"params":[{"from":"1999",
"to":"2008", "minrating":"1"}]}'  http://localhost:8080/rpcroute
```

A) Example of requests for the REST service:

Search for specific ISBN

```
curl -s  -H "Accept: application/json"  http://localhost:8080/book/isbn/1111111111111  | python
-mjson.tool
```

Search books released between 2000 and 2018

```
curl -s  -H "Accept: application/json"  http://localhost:8080/book/from/2000/to/2018 | python
-mjson.tool
```

Search books with author having "lor" in their names and a rating superior to 7

```
curl -s  -H "Accept: application/json"   http://localhost:8080/book/author/lor/minrating/8 |
python -mjson.tool
```

B) Example of requests for the JSON-RPC service

Search books published between 1999 and 2008

```
curl -s -H "Accept: application/json" --request GET   --data '{"params":[{"from":"1999", "to":"2008", "minrating":"1"}]}'  http://localhost:8080/rpcroute | python -mjson.tool
```

C) Example of requests for the SOAP-XML service

NB: The request is contained in a file called “soap_request.xml”, located at the root of the project.

Search books published between 1999 and 2005 (what the current xml file does):

```
curl -i -H "Content-Type: text/xml;charset=UTF-8" --header "SOAPAction:urn:GetBooks" --data @soap_request.xml http://localhost:8080/soap
```

3 Methods Allowed

If the request is valid but no book is found, it just returns an empty object. And although none of the restful parameters are mandatory, they need to be in the right order.

There is no "id" to provide in the parameters, so the traditionnal get/update/delete methods are never called. The getList / updateList / deleteList are called instead. The controllers only accept the GET method. Any other method returns a 405:

```
curl -s  -i -X PUT http://localhost:8080/book
```

```
curl -s  -i -X POST http://localhost:8080/book
```

4 Unit Tests

NB: The Unit Tests are only for the RESTful service.

They check the following:

1. Checks if route can be reached and send a success response code.
2. Checks if the response is always in proper Json (either full or empty)
3. Checks if the POST / PUT / PATCH / DELETE request properly return a 405 response.
4. Checks if a broken / non valid route returns a 404 as it should.
5. Checks if a GET request with no parameter properly returns json array with the key "nodata".

To start the test, from the root of the project, run : **./vendor/bin/phpunit**