Canonical Technical Exercise

Step 1: Describe the expected user experience

This should include all CLI commands and their options, think of this as writing the help or manual page of your software ahead of time, including documented examples of user interactions.

▼ Find

> `find`

▼ Find a book

> `find -`

Flags:

`-isbn` (required) - isbn of the book

`-all` (optional) - show all books

▼ Find a collection

> `find -c`

Flags:

`-n` (optional) - collection name

`-all` (optional) - show all collections

```
# Find Harry Potter and the Deathly Hallows via its ISBN
find -b -isbn 9780545010221
# Find all books
find -b -all
# Find a collection named Favourites
find -c -n "Favourites"
```

▼ Add

> `add`

▼ Add a book

> `add -b`

Flags:

`-isbn` (required) - isbn of book

`-t` (required) - title of book

`-a` (required) - author of book

`-pd` (required) - published date

`-e` (optional) - edition

`-g` (optional) - genre

`-d` (optional) - description

Add a collection

> `add -c`

Flags:

`-n` (required) - name of collection

`-d` (optional) - description

```
# Add Harry Potter and the Deathly Hallows to the Book Database
add -b -isbn 9780545010222 -t "Harry Potter and the Deathly Hallows" -a "JK Rowli
ng" -pd "2007-07-01"

# Add a new Favourites collection to the database
add -c -n "Favourites" -d "My favourite books"
```

Update

> `update`

Update book

> `update -b`

Flags:

`-isbn` (required) - isbn of book

`-t` (optional) - title of book

`-a` (optional) - author of book

`-pd` (optional) - published date

`-e` (optional) - edition

`-g` (optional) - genre

`-d` (optional) - description

Update collection

> `update -c`

Flags:

`-n` (required) - name of collection

`-isbn` (required) - isbn of book to add to collection

```
# Update a books information
update -b -isbn 9780545010222 -t "Harry Potter and the Deathly Hallows" -a "JK Ro
```

```
wling" -pd "2007-07-01" -e 1 -g "Young Adult"

# Add Harry Potter and the Deathly Hallows to your favourites
update -c -n "Favourites" -isbn 9780545010222
```

▼ Delete

> `delete`

▼ Delete book

> `delete -b`

Flags:

`-isbn` (required) - isbn of book

▼ Delete collection

> `delete -c`

Flags:

`-n` (required) - name of collection

```
# Delete Harry Potter and the Deathly Hallows
delete -b -isbn 9780545010222

# Delete the favourites collection
delete -c -n "Favourites"
```

Step 2: Describe the expected REST API

This should include the documentation of the REST API, all supported methods and what they'll be doing, examples of input and output data and what query parameters will be supported and what they'll do. See for example the actual REST API documentation of LXD.

Retrieve a given book

Using a book's unique isbn, retrieve a book and it's given metadata. Alternatively with no isbn provided, retrieve a list of all books.

Request

▼ GET /books

`isbn` string

Response

200

```
{
        "title": "Harry Potter and the Deathly Hallows",
        "author": "JK Rowling",
        "published_date": "January 21, 2007",
        "edition": 1,
        "genre": "Fantasy",
        "description": "It's no longer safe for Harry at Hogwarts, so he and his best
}
```


400


```
{
    "status": "Error",
    "status_code": 400
}
```


Add a book

Upload a book and it’s given metadata to the system.

Request

▼ **POST /books**

`isbn` string (required)

`title` string (required)

`author` string (required)

`published_date` string (required)

`edition` integer

`genre` array of strings

`description` string

Response

201


```
{
    "status": "Created",
    "status_code": 201
}
```


400

```
{
    "status": "Error",
    "status_code": 400
}
```

Delete a book

Delete a book using it’s unique isbn and remove it from any collections it belongs to

Request

▼ **DELETE /books**

isbn string (required)

Response

204

```
{
    "status": "Success",
    "status_code": 204
}
```

400

```
{
    "status": "Error",
    "status_code": 400
}
```

Retrieve a collection

Given the name of a collection, retrieve a list of all of the book titles in the given collection or a list of all collections and their descriptions.

Request

▼ **GET /collections**

collection_name string

Response

200

```
{
    "Favourites": {
        "isbn": "9780545010222",
        "title": "Harry Potter and the Deathly Hallows",
```

```
        "author": "JK Rowling",
        "published_date": "2001-01-07",
        "edition": 1,
        "genre": "Fantasy",
        "description": "It's no longer safe for Harry at Hogwarts, so he and his best
    }
}
```

400

```
{
    "status": "Error",
    "status_code": 400
}
```

Delete a collection

Using the provided name, the collection will be deleted (but not the books!)

Request

▼ DELETE /collections

`collection_name` string (required)

Response

204

```
{
    "status": "Success",
    "status_code": 204
}
```

400

```
{
    "status": "Error",
    "status_code": 400
}
```

Create a collection

Create an empty collection with a given unique name.

Request

▼ POST /collections

`collection_name` string (required)

Response

201

```
{
    "status": "Created",
    "status_code": 201
}
```

400

```
{
    "status": "Error",
    "status_code": 400
}
```

Update a collection

Add book to a given collection by inputting the collection name and an isbn of the book you're wanting to add. Similarly you can remove a book by following the same steps and setting the to_delete parameter to true.

Request

▼ **PUT /collections**

`collection_name` string (required)

`isbn` string (required)

`to_delete` boolean *default: to_delete=False*

Response

202

```
{
    "status": "Success",
    "status_code": 202
}
```

400

```
{
    "status": "Error",
    "status_code": 400
}
```

Step 3: Describe the database structure

This should include all the tables you expect to use to store this data, for each table, all its expected columns, column types, relations, constraints and any index you'd like to add. You may write this directly in SQL or use tables or text as an alternative way to describe what you're going for.

Books

ISBN	Title	Author	Published Date	Edition	Genre	Description
PK VARCHAR(255)	VARCHAR(255)	VARCHAR(255)	DATE	INT	VARCHAR(255)	VARCHAR(255)

Collections

Name	Description
PK VARCHAR(255)	VARCHAR(255)

Books_Collections

ISBN	Collection_name
PK FK VARCHAR(255)	**PK FK** VARCHAR(255)