1 Practice 15.1. Databases with Node.js and MongoDB

> This practice consists of the first of a series of practices, consisting of a Book's review application. Each practice will be based on the previous one.
>
> Once done, do the submission via GitHub Classroom as detailed in the end of the practice.

1.1 Application overview

In this app we want to do a book's review API. In the database we will store a list of books and each book will contain a list of reviews.

In this practice we will create the database and operate with some data.

1.2 Exercise 1. Creating the project

Clone the project repository task:

1. You need a GitHub account. Create one if you don't have any.
2. Click on the assignment invitation link. Enter your GitHub credentials and authorize GitHub Classroom to access your GiHub account.
3. Select your name from the "Join the Classroom" list.
4. Accept the assignment.
5. Go to your assignment repository. Copy the repository URL (http or ssh), that is in the green "Code" button.
6. In your computer, clone the project on your practices folder:

```
git clone your_repo_url
```

7. Enter into your project folder and initialize your project with npm:

```
npm init
```

8. Start coding.

1.3 Exercise 2. Populating the database

In order to having some data in the DB, you can execute the next query on your MongoDB console (you need MongoDB installed locally or on a Docker container, if you don't have it, review the Unit 15 annex):

```javascript
db.books.insertMany([
  {
    "isbn": "A22334",
    "title": "The Lathe Of Heaven",
    "author": "Ursula K. Le Guin",
    "publisher": "Scribner",
    "year": 2023,
    "reviews": [
      {
        "date": ISODate('2024-01-04T17:49:36.143Z'),
        "mark": 4,
        "text": "A rare and powerful synthesis of poetry and science, reason
        ↪   and emotion.",
        "author": "The New York Times"
      },
      {
        "date": ISODate('2023-01-08T00:00:00.000Z'),
        "mark": 3,
        "text": "Gracefully developed...extremely inventive",
        "author": "Newsweek"
      }
    ]
  },
  {
    "isbn": "B33445",
    "title": "Lazarillo de Tormes",
    "publisher": "Alfaguara",
    "year": 2010,
    "reviews": []
  },
  {
    "isbn": "C55667",
    "title": "The Lord of The Rings",
    "author": "J.R.R. Tolkien",
    "publisher": "Clarion Books",
    "year": 2019,
```

```
    "reviews": [
      {
        "mark": 2,
        "text": "Amazing story of courage and friendship",
        "author": "J. Goodell",
        "date": ISODate('2024-01-04T22:28:11.231Z')


      }
    ]
  }
])
```

1.4 Exercise 3. Models

Install Mongoose in your App. Then create the next schema for book:

- isbn: String, required, unique, minimum length 6 characters.
- title: String, required.
- author: String.
- publisher: String.
- year: Number, between 1000 and 3000.
- reviews: array:

    – date: type date, default value current date.
    – mark: Number, between 1 and 5, default value 1.
    – text: String.
    – author: String, required.

Trim all the data too.

1.5 Exercise 4. Select queries

Create a script called `select_test.js` with the next queries:

- The list of all the books with the reviews for each book (not only the id).
- All the data for one book (including the reviews) selected by its `id`.
- All the data for one book selected by its ISBN.
- A list of all the books, with all the data, searched by title, case insensitive. For example if the value of the search string is `'the'`, the query must return *'The Lathe Of Heaven'* and *'The Lord of The Rings'*. Tip: use regular expressions.

Print the results and errors on the console, preceding each query by its description (for instance *'Find by ISBN result: …'*)

1.6 Exercise 5. Insert queries

Create a script called `insert_test.js` that inserts 2 new books, one with all the fields and 2 reviews and the other one without reviews and with some blank (not required) fields.

Print the results and errors on the console.

1.7 Exercise 6. Update queries

Create a script called `update_test.js`. On it do 2 update queries, searching by the book's id:

- Modify some data of a book.
- Add a new review to an existing book (tip: use `$push` instead of `$set`).

Print the results and errors on the console.

1.8 Exercise 7. Delete queries

Create a script called `delete_test.js`. On it delete a book searching by its id.

1.9 How to submit to GitHub Classroom

Once you finish the task, make a commit with the comment "PRACTICE 15.1 SUBMISSION COMMIT" and push it to GitHub. It's recommended to tag your commit with the tag "Practice_15.1".

Before that, you can do the commits and pushes you want. If you change your code after your submission commit, make another commit and push with the same text in the message adding the corrections you've done.

If you have any doubt in your task, you can push your code and ask me by email what's your problem. It will make it easier for both the solutions of code issues.