



TECHNOLOGIES JS

COMPTE RENDU DU TP: BOOK READING TRACKER

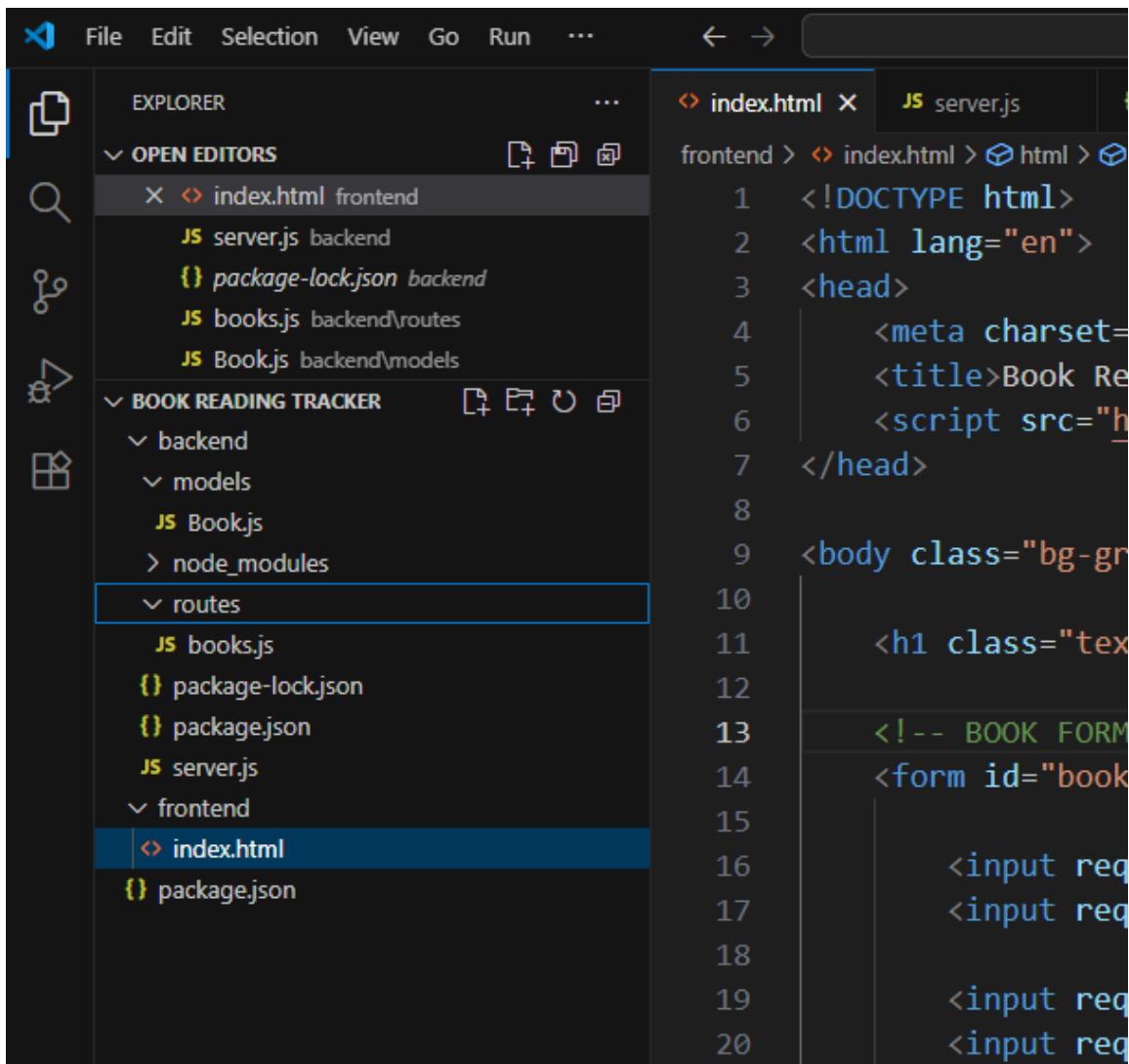
Encadré par : Pr. Amal OURDOU

Réalisé par : Houda Rebbouh

TP: Book Reading tracker

- Create an HTML file where we have a form that register new books (use tailwindCSS to style it)
- Each book have a title(string), author(string), number of pages(number), status(String Enum), price (number), number of pages read (number < number of pages), format(String Enum), suggested by(string), finished(boolean).
- Status can have one of the following values: Read, Re-read, DNF, Currently reading, Returned Unread, Want to read
- Format can have one of the following values: Print, PDF, Ebook, AudioBook
- Be default finished is equal to 0, the finished value will change to 1 automatically when number of pages read is euqal to number of pages
- Create a class book have the following methods: a constructor, currentlyAt, deleteBook
- The book class should be its own module.
- Create a web page where we can track our reading by listing books and showing the percentage of reading for each book, and a global section where you can see the total amount of book read and the amount of pages
- The books are stored in MongoDB

PROJECT STRUCTURE :



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure:
 - OPEN EDITORS:** index.html (frontend), server.js (backend), package-lock.json (backend), books.js (backend\routes), Book.js (backend\models).
 - BOOK READING TRACKER:** backend (models: Book.js), node_modules, routes (books.js, package-lock.json, package.json, server.js), frontend (index.html, package.json).
- Code Editor (Right):** The file `index.html` is open, showing the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Book Reader</title>
<script src="https://unpkg.com/tailwindcss@^2/dist/tailwind.min.js"></script>
</head>
<body class="bg-gray-100">
<h1 class="text-center text-3xl font-bold mb-4">Book Reader</h1>
<!-- BOOK FORM -->
<form id="book-form" method="POST" action="/add">
<input type="text" name="title" placeholder="Title" required="required" />
<input type="text" name="author" placeholder="Author" required="required" />
<input type="number" name="pages" placeholder="Pages" required="required" />
<input type="number" name="read" placeholder="Read" required="required" />
<input type="text" name="format" placeholder="Format" required="required" />
<input type="text" name="suggested_by" placeholder="Suggested By" required="required" />
<input type="checkbox" name="finished" checked="checked" /> Finished
<button type="submit" class="bg-blue-500 text-white font-bold py-2 px-4 rounded">Add Book</button>
</form>
</body>
```

Le projet est organisé en deux parties principales : un backend et un frontend. Le backend contient le fichier server.js, qui démarre le serveur Express, configure la connexion à MongoDB et charge les routes. Le dossier models/ contient les schémas Mongoose, notamment Book.js, qui définit la structure des livres dans la base de données. Le dossier routes/ contient les fichiers de routes Express, comme books.js, qui gère les opérations CRUD liées aux livres. Le fichier package.json gère les dépendances du backend. De l'autre côté, le dossier frontend/ contient index.html, qui représente l'interface utilisateur : formulaire d'ajout de livres, affichage de la liste, et intégration de TailwindCSS, associé à un script JavaScript pour communiquer avec l'API backend.

INTERFACE UTILISATEUR :

The screenshot shows a web browser window displaying the 'Book Reading Tracker' application. The URL in the address bar is 127.0.0.1:5500/frontend/index.html. The page features a header with the application name and several input fields for adding a book: Title, Author, Total Pages, Pages Read, Select Status (with a dropdown arrow), Select Format (with a dropdown arrow), Suggested by, and Price. A large blue button labeled 'Add Book' is positioned below these fields. Below the form, a message states 'Books finished: 0 | Total pages read: 40'. A book entry for 'harry potter' by jk rowling is listed, showing it is currently being read. A progress bar indicates 40.0% read (40/100). A red 'Delete' button is located next to the book entry. The browser interface includes standard navigation buttons, a search bar, and a notification for a new Chrome update.

DEMARRAGE DE MONGODB :

```
Administateur : Windows PowerShell
+ CategoryInfo          : InvalidArgument : () [Set-Location], ParameterBindingException
+ FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.PowerShell.Commands.SetLocationCommand

PS C:\Windows\system32> cd "C:\Program Files\MongoDB\Server\8.2\bin"
PS C:\Program Files\MongoDB\Server\8.2\bin> ./mongosh.exe
./mongosh.exe : Le terme «./mongosh.exe» n'est pas reconnu comme nom d'applet de commande, fonction, fichier de script ou programme exécutable. Vérifiez l'orthographe du nom, ou si un chemin d'accès existe, vérifiez que le chemin d'accès est correct et réessayez.
Au caractère Ligne:1 : 1
+ ./mongosh.exe
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (./mongosh.exe:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Program Files\MongoDB\Server\8.2\bin> .\mongod.exe
{"t":{"$date":"2025-11-09T14:28:10.836+01:00"}, "s": "I", "c": "-", "id": 8991200, "ctx": "thread1", "msg": "Shuffling initializers", "attr": {"seed": 3204579644}}
{"t":{"$date":"2025-11-09T14:28:10.969+01:00"}, "s": "I", "c": "CONTROL", "id": 97374, "ctx": "thread1", "msg": "Automatically disabling TLS 1.0 and TLS 1.1, to force-enable TLS 1.1 specify --sslDisabledProtocols 'TLS1_0'; to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2025-11-09T14:28:10.978+01:00"}, "s": "I", "c": "NETWORK", "id": 4915701, "ctx": "thread1", "msg": "Initialized wire specification", "attr": {"spec": {"incomingExternalClient": {"minWireVersion": 0, "maxWireVersion": 27}, "incomingInternalClient": {"minWireVersion": 0, "maxWireVersion": 27}, "outgoing": {"minWireVersion": 6, "maxWireVersion": 27}, "isInternalClient": true}}}
{"t":{"$date":"2025-11-09T14:28:10.987+01:00"}, "s": "I", "c": "CONTROL", "id": 5945603, "ctx": "thread1", "msg": "Multi threading initialized"}
{"t":{"$date":"2025-11-09T14:28:10.988+01:00"}, "s": "I", "c": "CONTROL", "id": 4615611, "ctx": "initandlisten", "msg": "MongoDB starting", "attr": {"pid": 17040, "port": 27017, "dbPath": "/data/db", "architecture": "64-bit", "host": "DESKTOP-8AJFV93"}}
{"t":{"$date":"2025-11-09T14:28:10.988+01:00"}, "s": "I", "c": "CONTROL", "id": 23398, "ctx": "initandlisten", "msg": "Target operating system minimum version", "attr": {"targetMinOS": "Windows 7/Windows Server 2008 R2"}}
{"t":{"$date":"2025-11-09T14:28:10.989+01:00"}, "s": "I", "c": "CONTROL", "id": 23403, "ctx": "initandlisten", "msg": "Build Info", "attr": {"buildInfo": {"version": "8.2.1", "gitVersion": "3312bdcf28aa65f5930005e21c2cb130f648b8c3", "modules": [], "allocator": "tcmalloc-gperf", "environment": {"distmod": "windows"}}}}
{"t":{"$date":"2025-11-09T14:28:10.990+01:00"}, "s": "I", "c": "CONTROL", "id": 51765, "ctx": "initandlisten", "msg": "Operating System", "attr": {"os": {"name": "Microsoft Windows Workstation (build 9200)", "version": "6.2 (build 9200)"}}, "attr": {"relatedParameters": ["tcpFastOpenServer", "tcpFastOpenClient", "tcpFastOpenQueueSize"]}}
{"t":{"$date":"2025-11-09T14:28:11.006+01:00"}, "s": "E", "c": "CONTROL", "id": 20557, "ctx": "initandlisten", "msg": "DBException in initAndListen, terminating", "attr": {"error": "NonExistentPath: Data directory \\data\\db not found. Create the missing directory or specify another path using (1) the --dbpath command line option, or (2) by adding the 'storage.dbPath' option in the configuration file."}}
{"t":{"$date":"2025-11-09T14:28:11.006+01:00"}, "s": "I", "c": "REPL", "id": 4784900, "ctx": "initandlisten", "msg": "Stepping down the ReplicationCoordinator for shutdown", "attr": {"waitTimeMillis": 15000}}
{"t":{"$date":"2025-11-09T14:28:11.014+01:00"}, "s": "I", "c": "REPL", "id": 4794602, "ctx": "initandlisten", "msg": "Attempting to return replica mode"}
```

CONNECTION AVEC MONGODB :

MongoDB Compass - localhost:27017/booktracker.books

Connections Edit View Collection Help

Compass

My Queries Data Modeling

CONNECTIONS (1)

localhost:27017 admin booktracker books config local

localhost:27017 > booktracker > books

Documents 1 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query +](#)

ADD DATA EXPORT DATA UPDATE DELETE

25 1-1 of 1 Explain Reset Find Options

```
_id: ObjectId('691097570bf3f4c6717e0e1c')
title: "harry potter"
author: "jk rowling"
pages: 100
status: "Currently reading"
price: 200
pagesRead: 40
format: "PDF"
suggestedBy: "mom"
finished: false
__v: 0
```

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with 'Connections' (localhost:27017) and a 'books' collection under 'booktracker'. The main area shows the 'Documents' tab for the 'books' collection, which contains one document. The document details are displayed in a code block: an ObjectId for '_id', 'harry potter' for 'title', 'jk rowling' for 'author', 100 for 'pages', 'Currently reading' for 'status', 200 for 'price', 40 for 'pagesRead', 'PDF' for 'format', 'mom' for 'suggestedBy', and false for 'finished'. The version is also listed as __v: 0. Navigation buttons like 'Find', 'Reset', and 'Options' are visible at the top right.