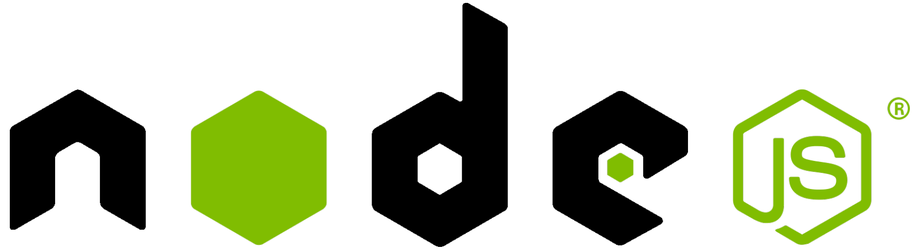
JAVASCRIPT

BACK-END(NODEJS, EXPRESS, API,CRUD-POSTMAN)

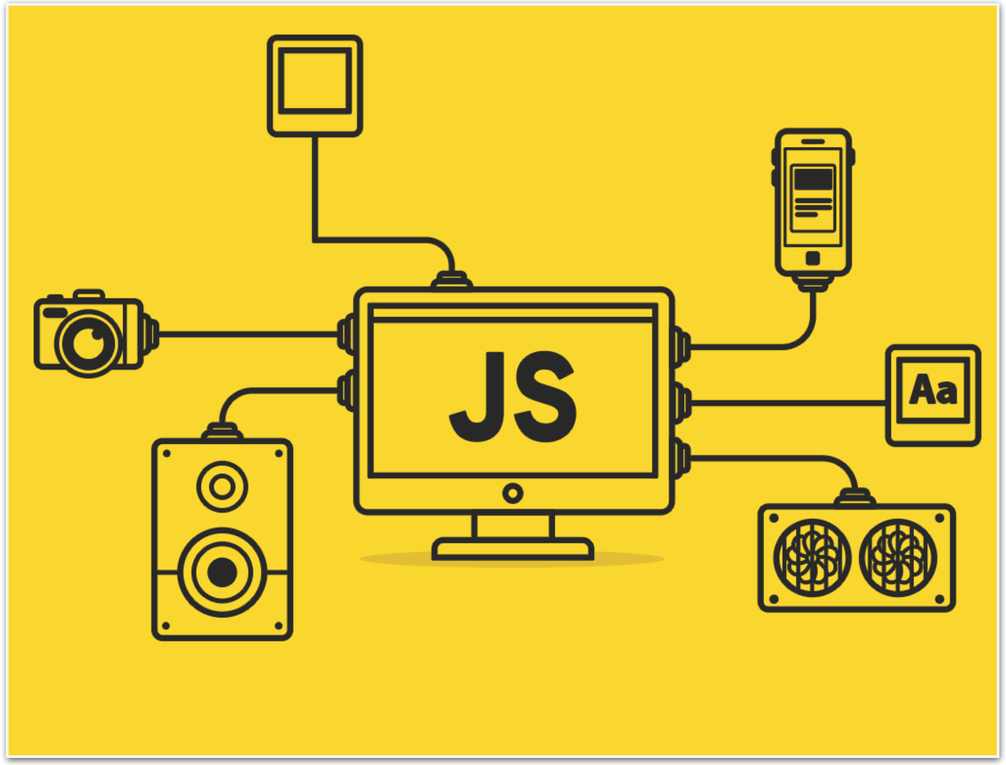
Anirach Mingkhwan

Anirach.m@fitm.kmutnb.ac.th



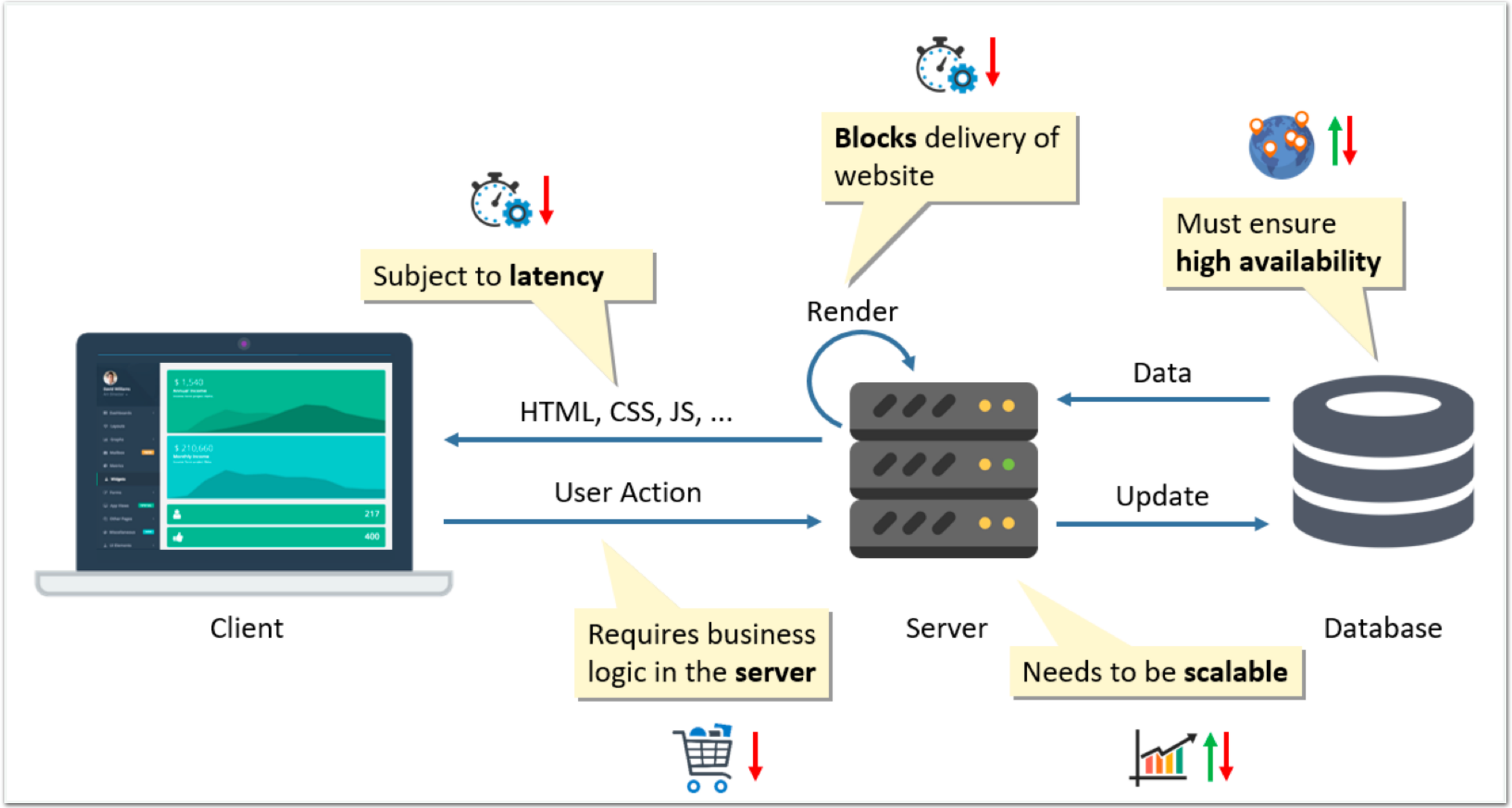
OUTLINE

- 3-TIERS Concepts
- API Architect
- Postman
- CRUD with Postman



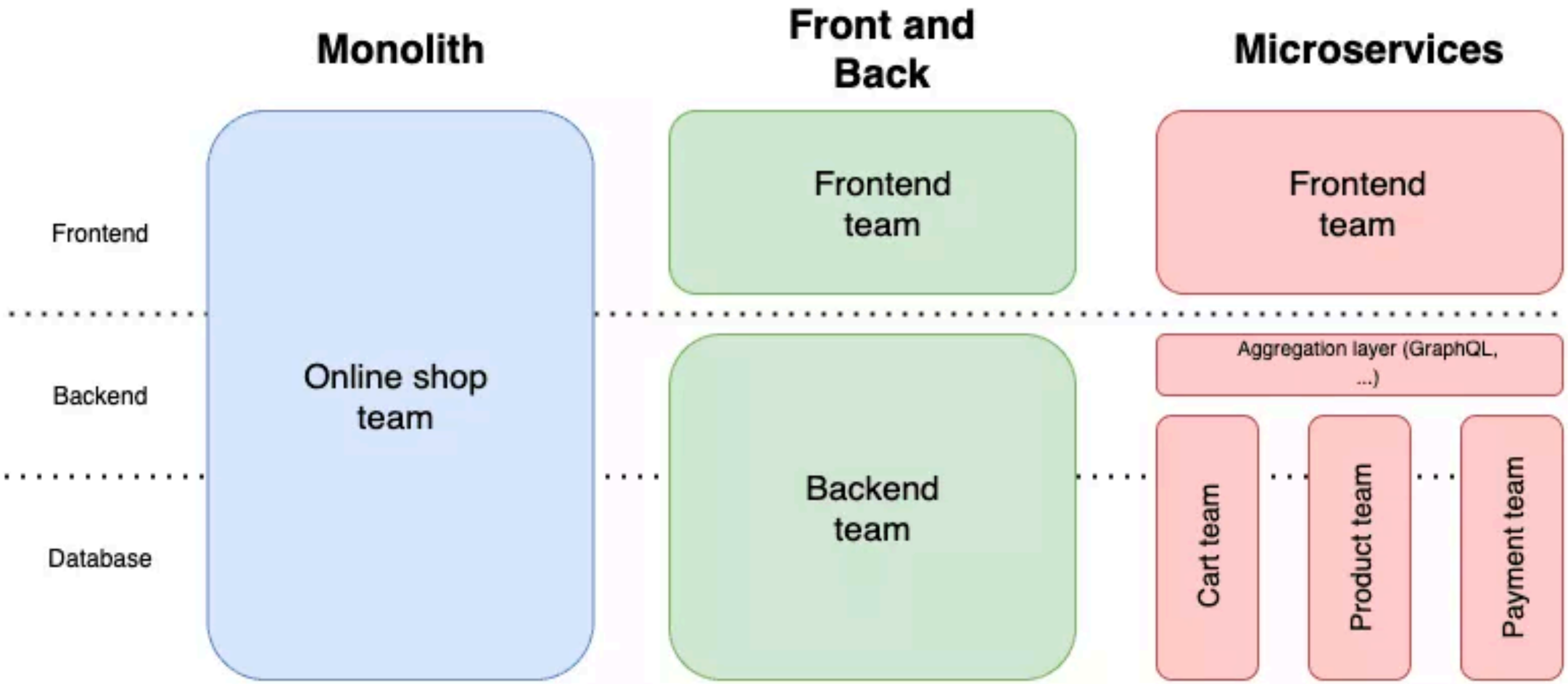
3-TIERS CONCEPT

The Classic 3-Tier Architecture

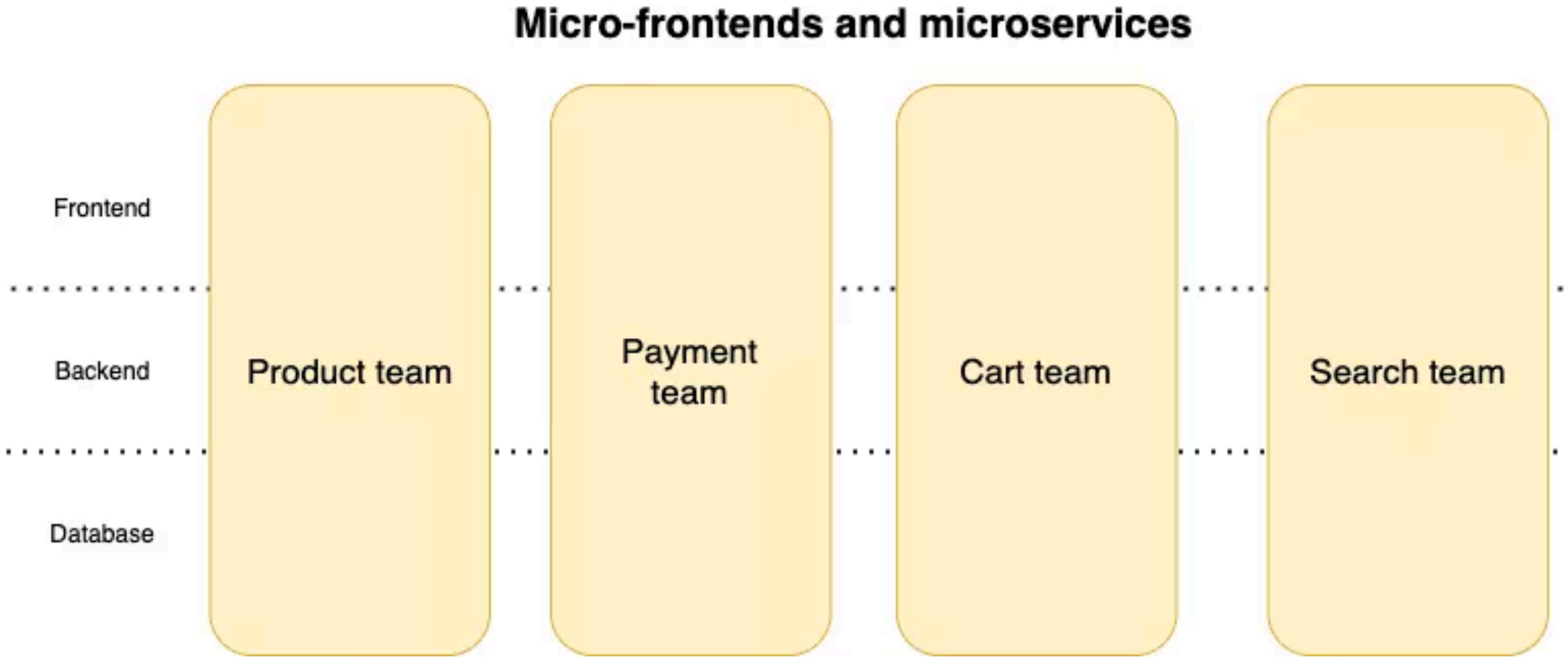


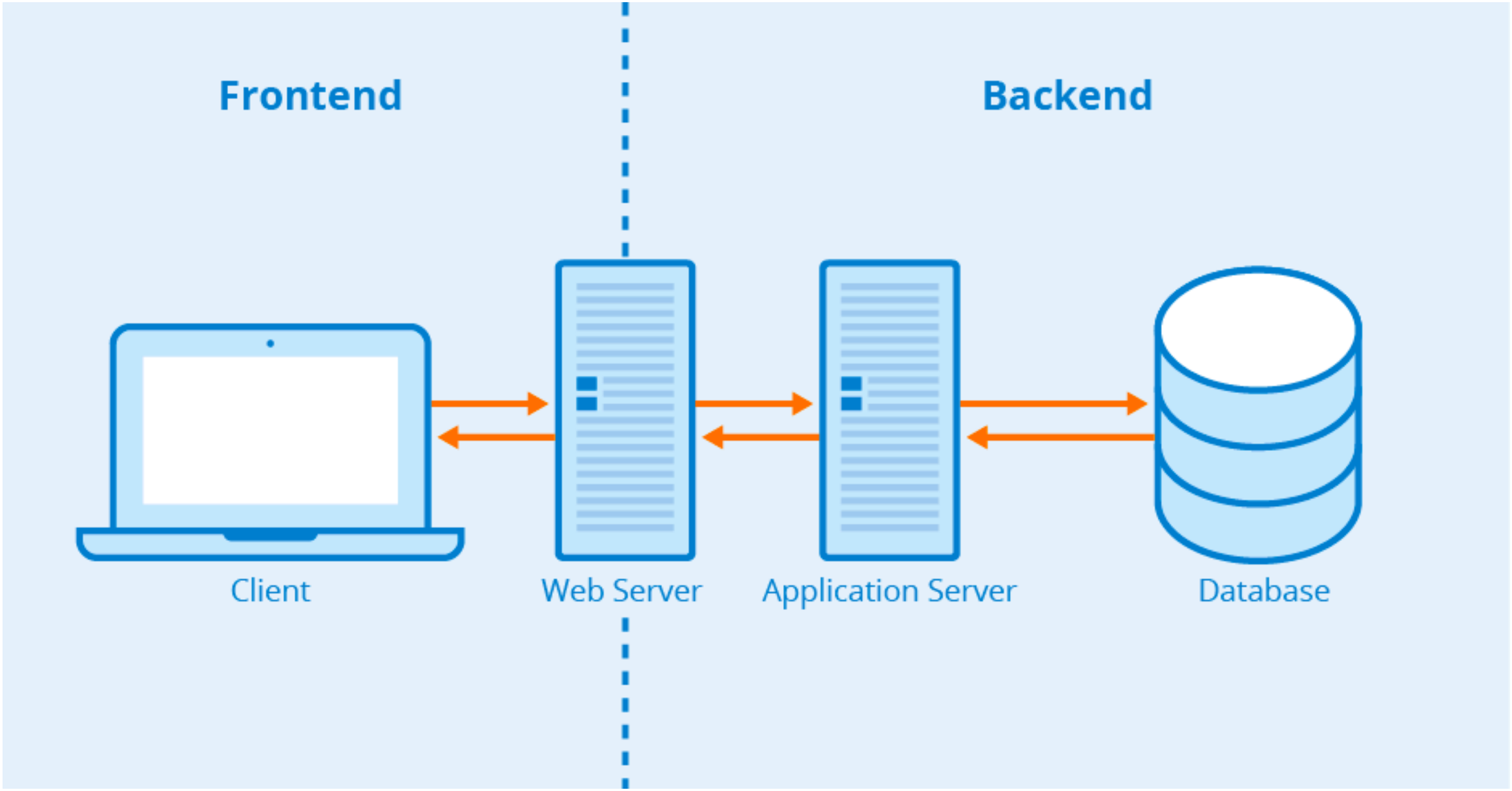
API ARCHITECTURE

API Platform Architecture

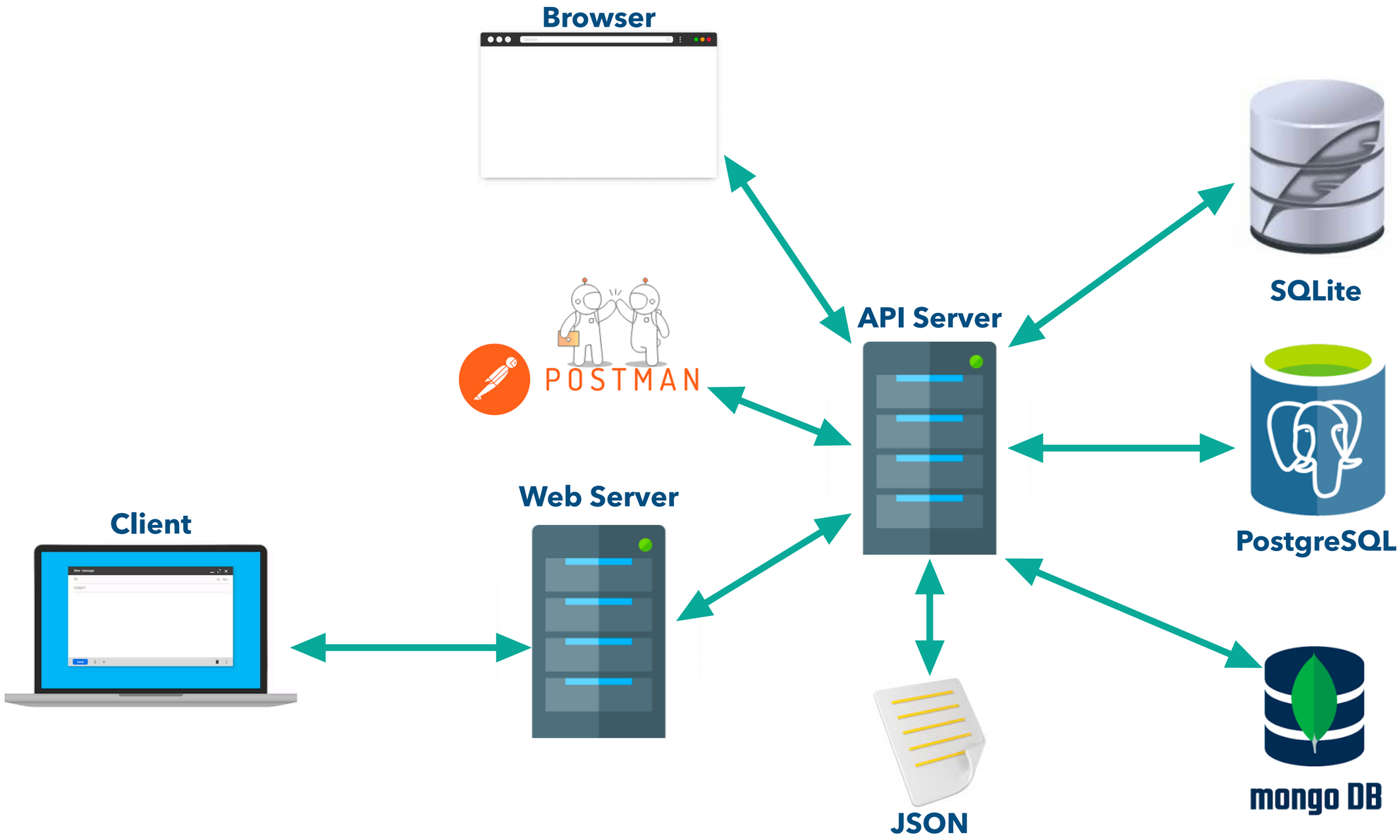


API Platform Architecture

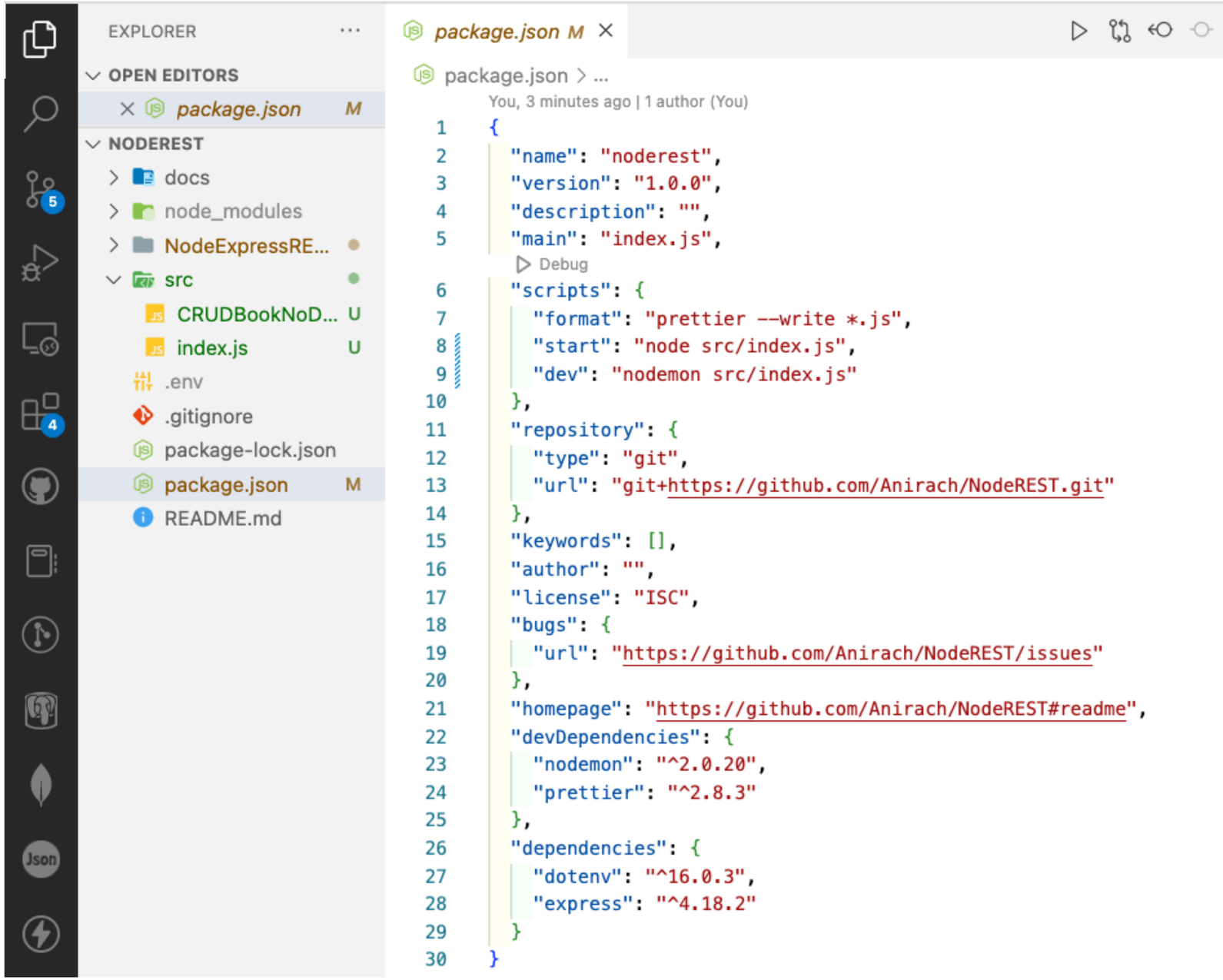


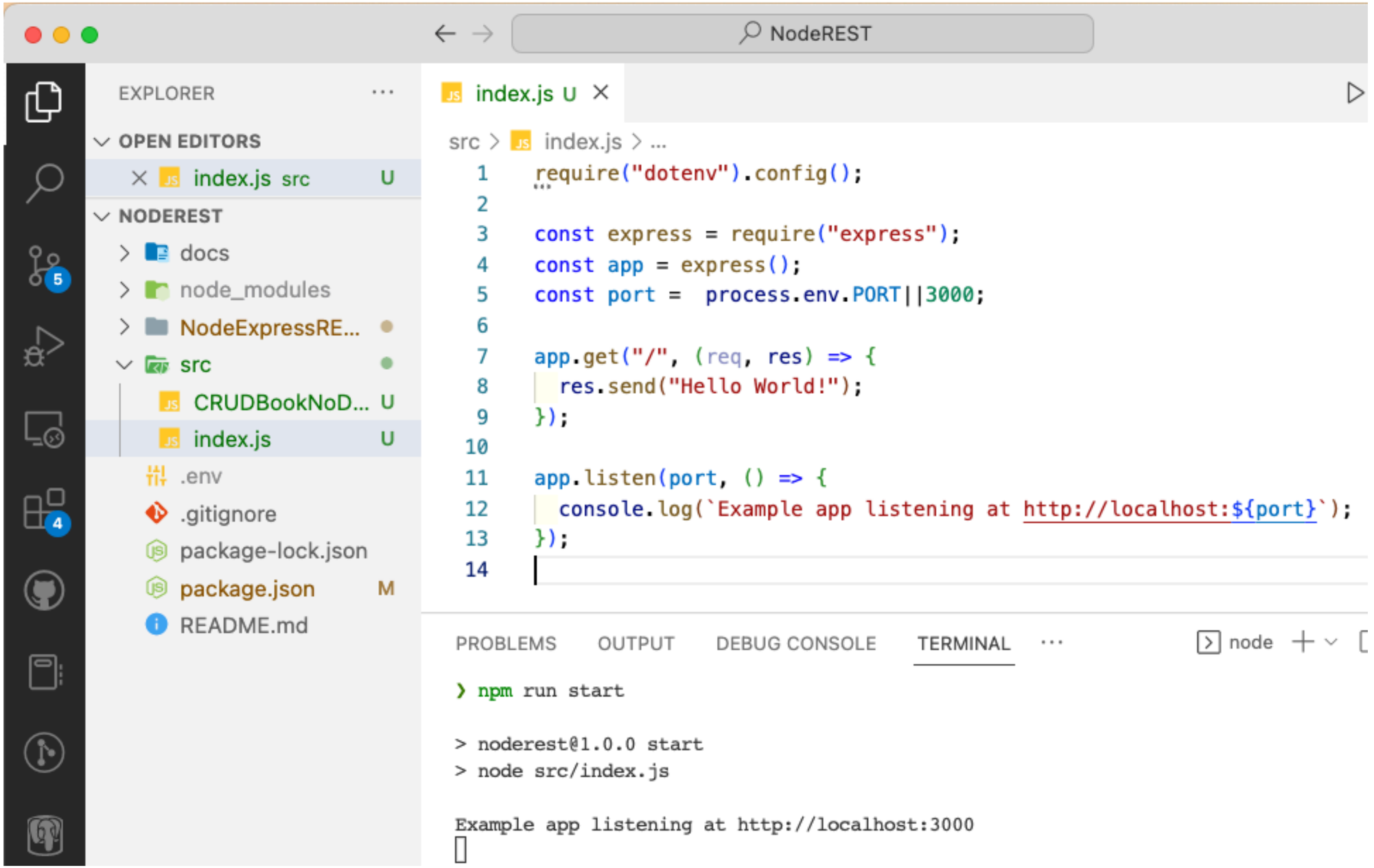

Frontend
Backend
Client
Web Server
Application Server
Database

What we will learn?

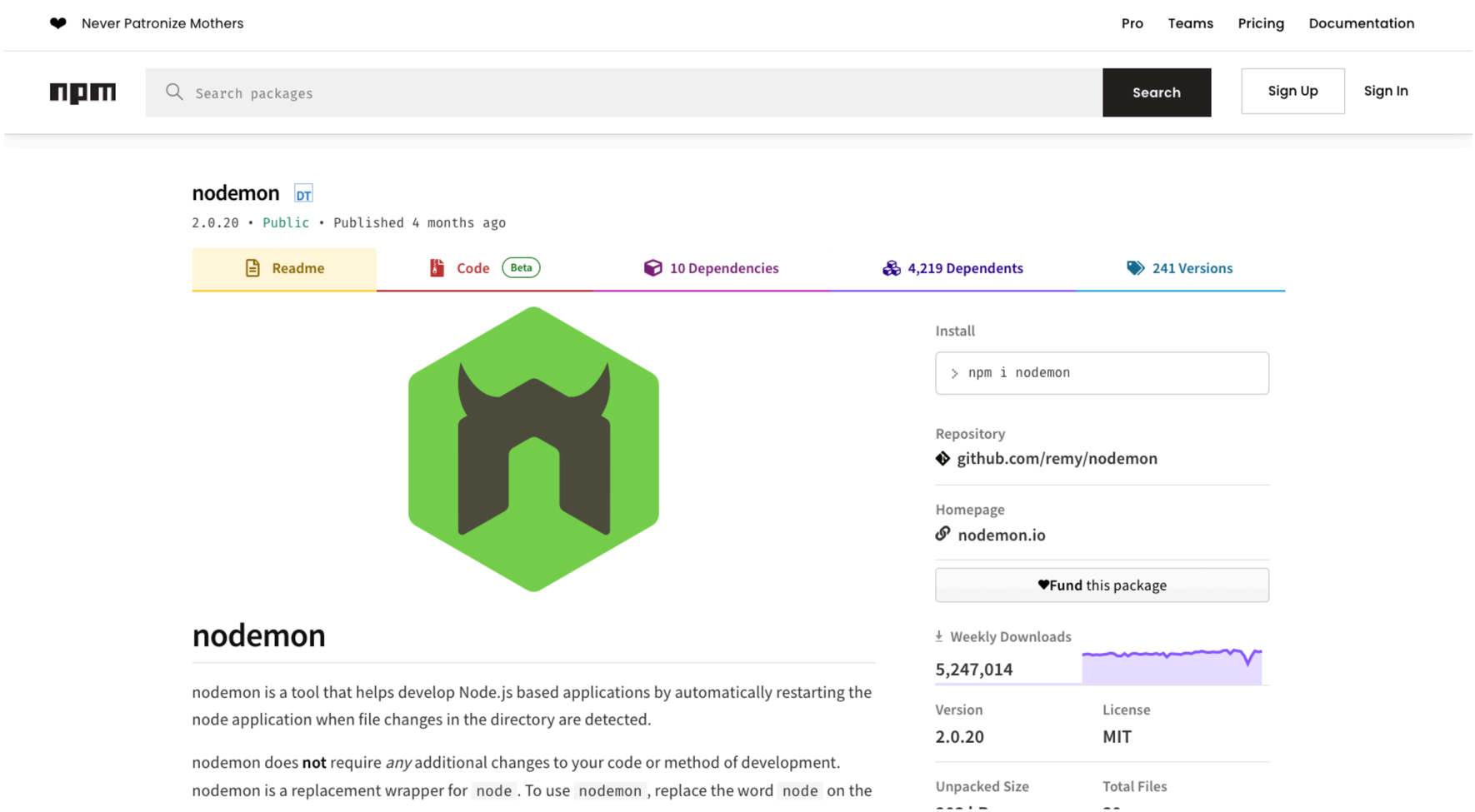


RECAP

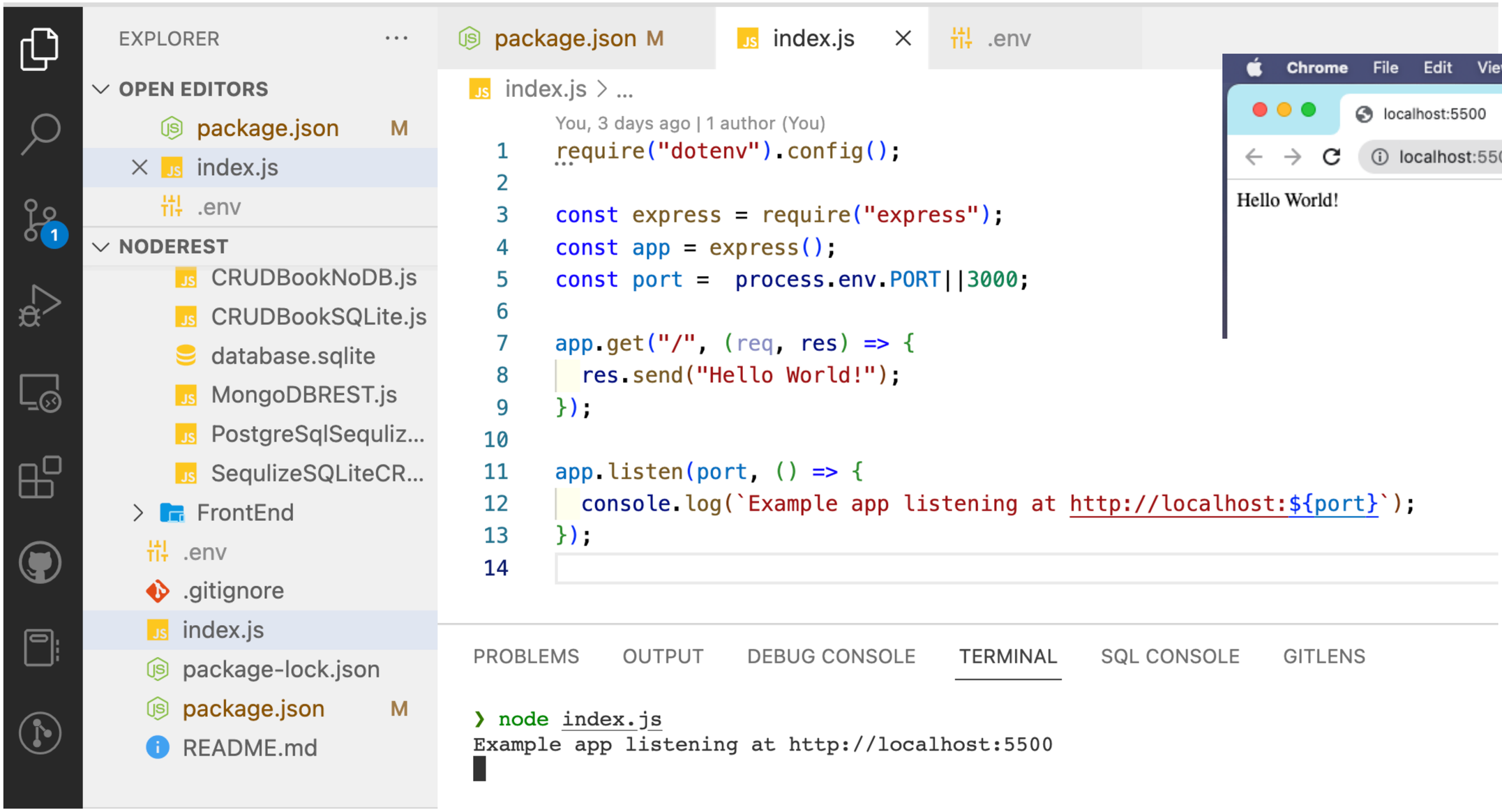

EXPLORER
OPEN EDITORS
package.json M
NODEREST
docs
node_modules
NodeExpressRE...
src
CRUDBookNoD... U
index.js U
.env
.gitignore
package-lock.json
package.json M
README.md
package.json M
package.json > ...
You, 3 minutes ago | 1 author (You)
{
"name": "noderest",
"version": "1.0.0",
"description": "",
"main": "index.js",
Debug
"scripts": {
"format": "prettier --write *.js",
"start": "node src/index.js",
"dev": "nodemon src/index.js"
},
"repository": {
"type": "git",
"url": "git+https://github.com/Anirach/NodeREST.git"
},
"keywords": [],
"author": "",
"license": "ISC",
"bugs": {
"url": "https://github.com/Anirach/NodeREST/issues"
},
"homepage": "https://github.com/Anirach/NodeREST#readme",
"devDependencies": {
"nodemon": "^2.0.20",
"prettier": "^2.8.3"
},
"dependencies": {
"dotenv": "^16.0.3",
"express": "^4.18.2"
}
}


NodeREST
EXPLORER
OPEN EDITORS
index.js src U
NODEREST
docs
node_modules
NodeExpressRE...
src
CRUDBookNoD... U
index.js U
.env
.gitignore
package-lock.json
package.json M
README.md
index.js U
src > index.js > ...
1 require("dotenv").config();
2
3 const express = require("express");
4 const app = express();
5 const port = process.env.PORT||3000;
6
7 app.get("/", (req, res) => {
8 res.send("Hello World!");
9 });
10
11 app.listen(port, () => {
12 console.log(`Example app listening at http://localhost:${port}`);
13 });
14
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
node
> npm run start
> noderest@1.0.0 start
> node src/index.js
Example app listening at http://localhost:3000

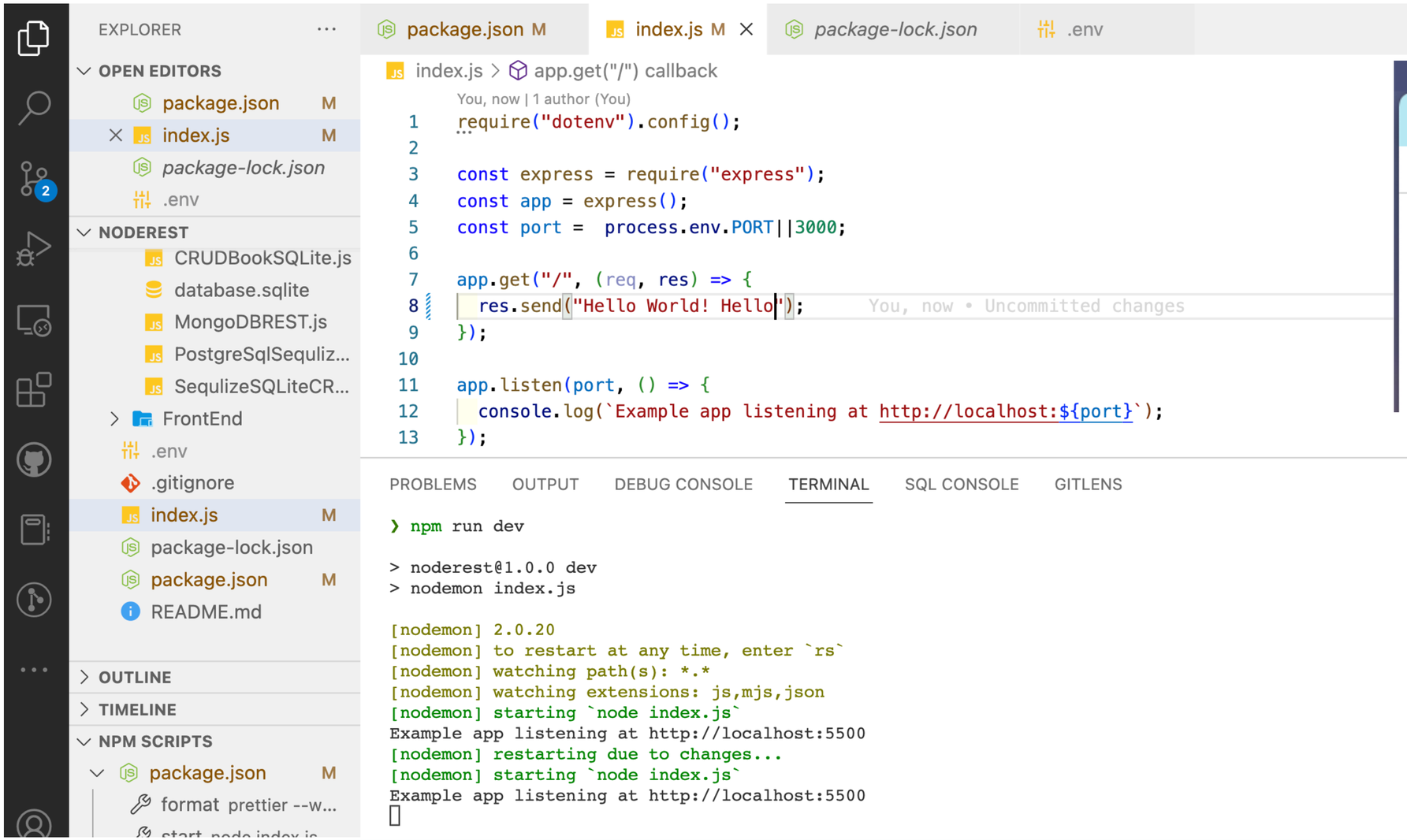
NODEMON



npm i -D nodemon

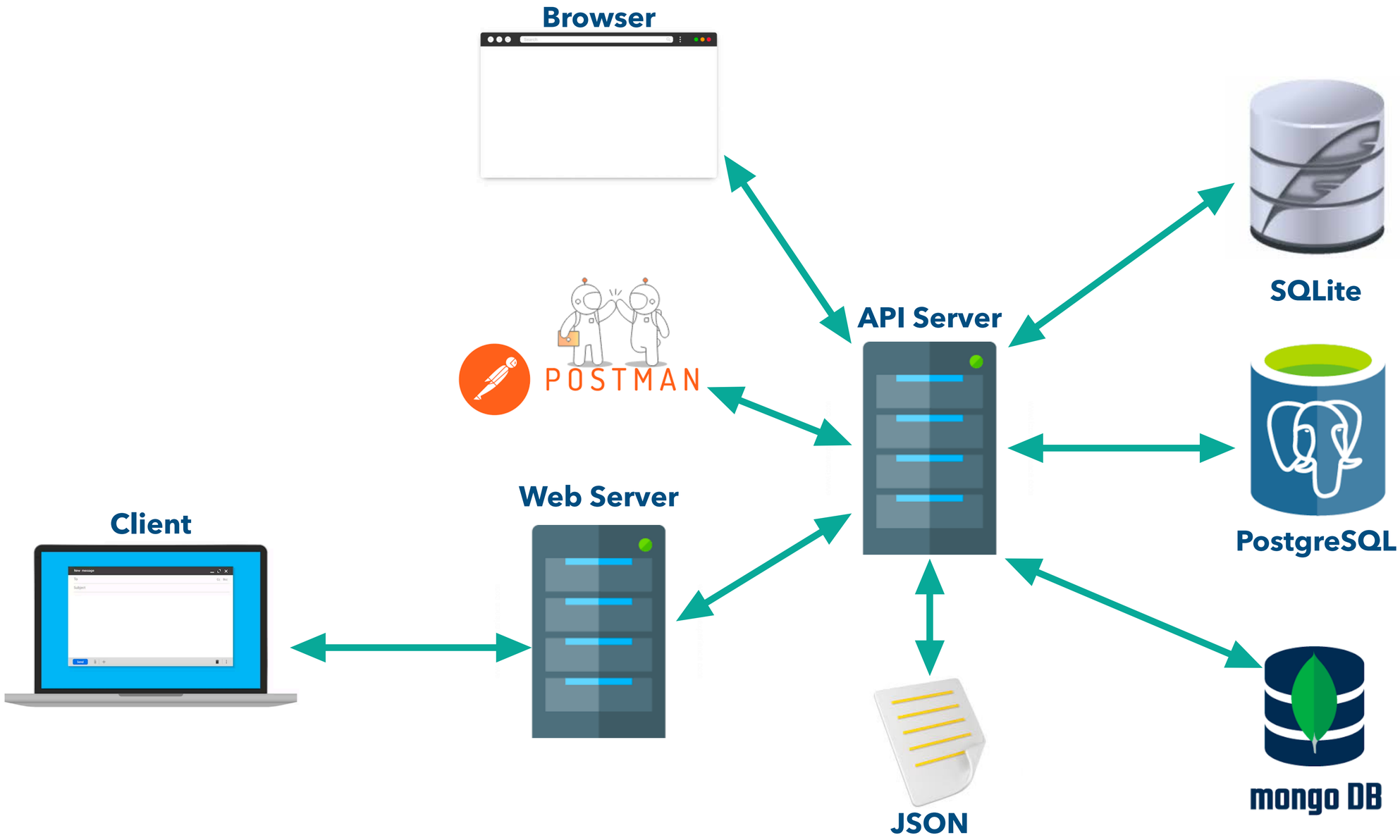

EXPLORER
OPEN EDITORS
package.json M
index.js
.env
NODEREST
CRUDBookNoDB.js
CRUDBookSQLite.js
database.sqlite
MongoDBREST.js
PostgreSqlSequliz...
SequlizeSQLiteCR...
FrontEnd
.env
.gitignore
index.js
package-lock.json
package.json M
README.md
package.json M
index.js
.env
index.js > ...
You, 3 days ago | 1 author (You)
1 require("dotenv").config();
2
3 const express = require("express");
4 const app = express();
5 const port = process.env.PORT||3000;
6
7 app.get("/", (req, res) => {
8 res.send("Hello World!");
9 });
10
11 app.listen(port, () => {
12 console.log(`Example app listening at http://localhost:${port}`);
13 });
14
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
GITLENS
❯ node index.js
Example app listening at http://localhost:5500
Chrome File Edit View History Bookmarks
localhost:5500
localhost:5500
Hello World!

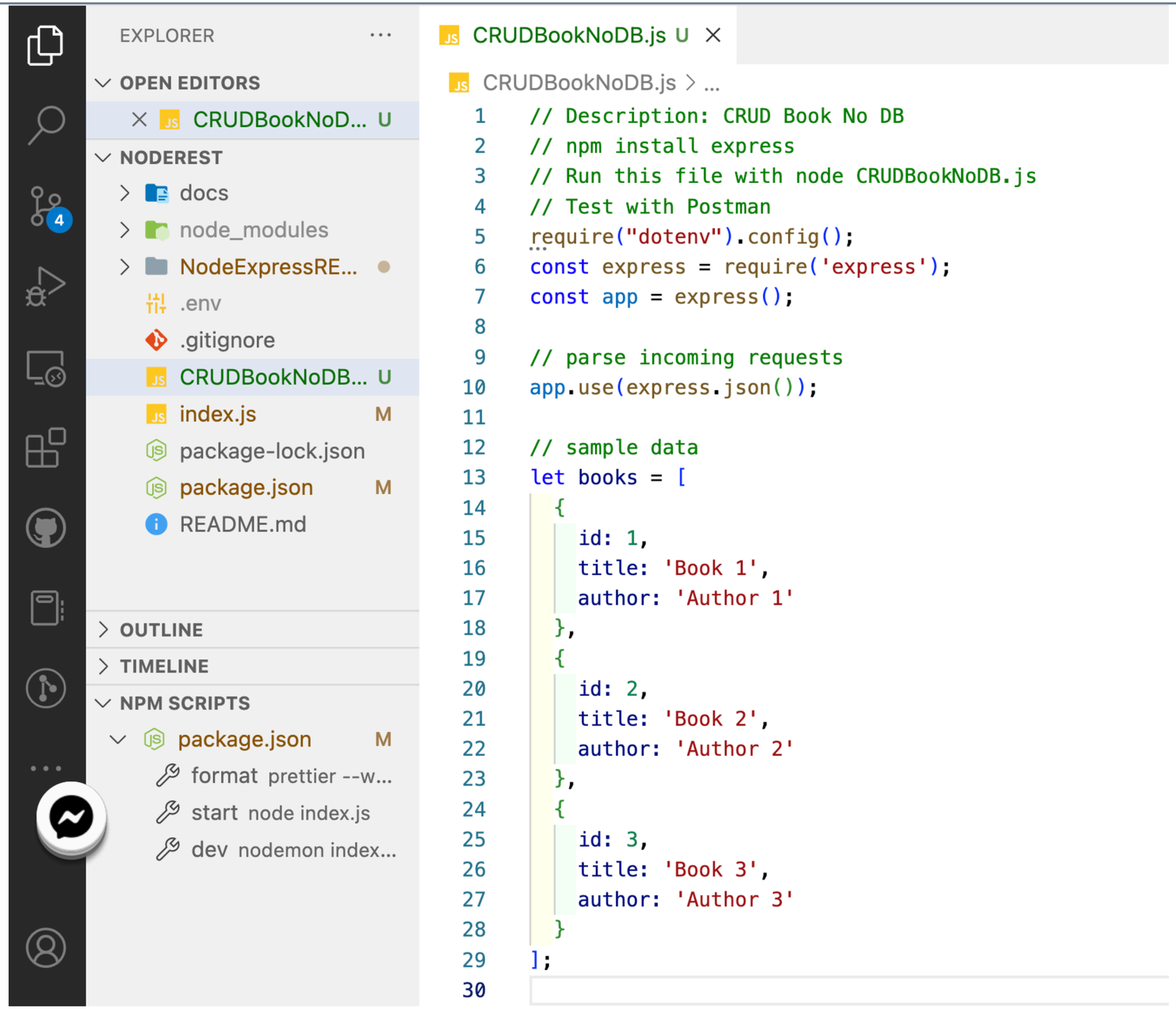
NODEMON EXAMPLE

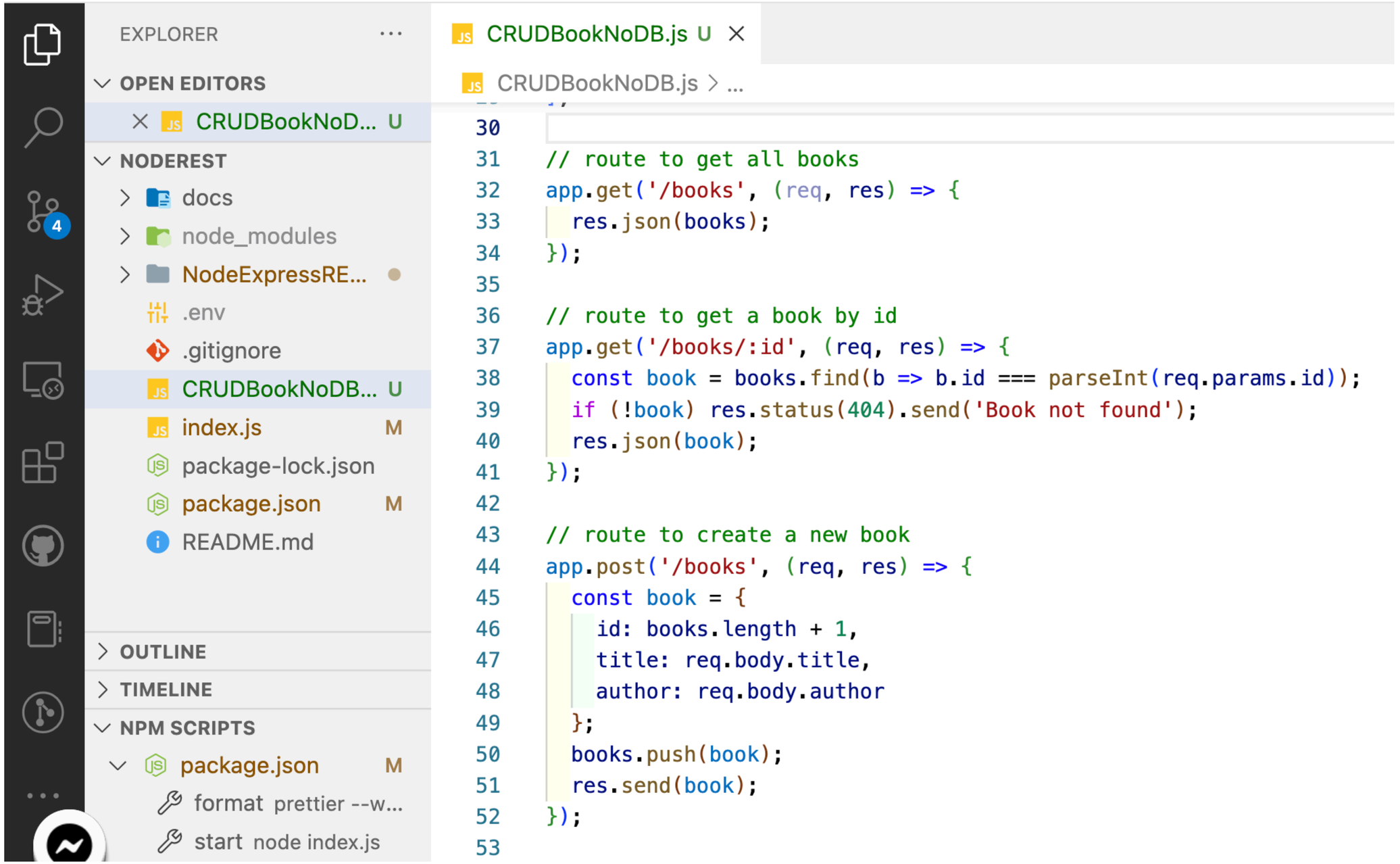


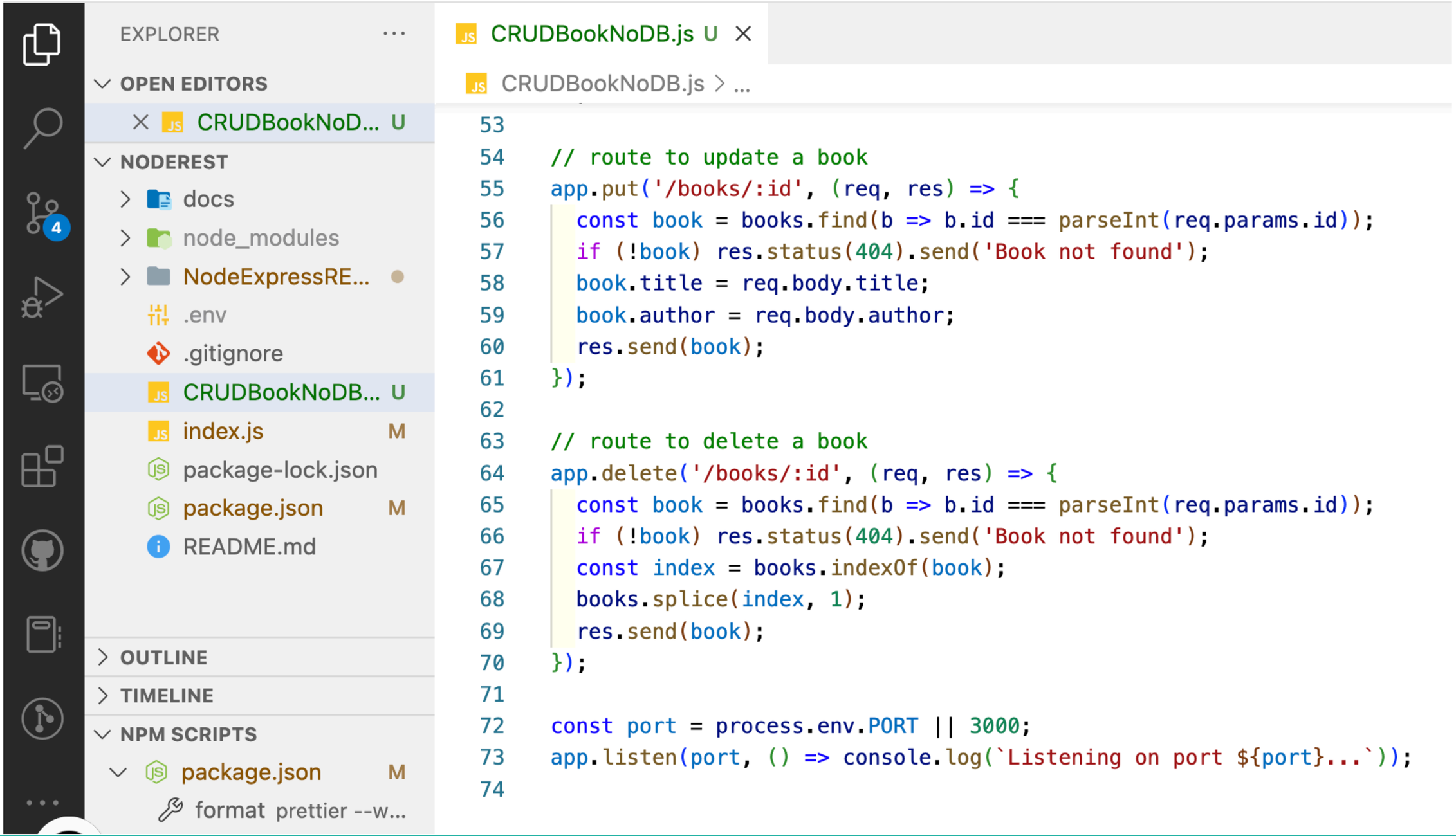
BACK-END

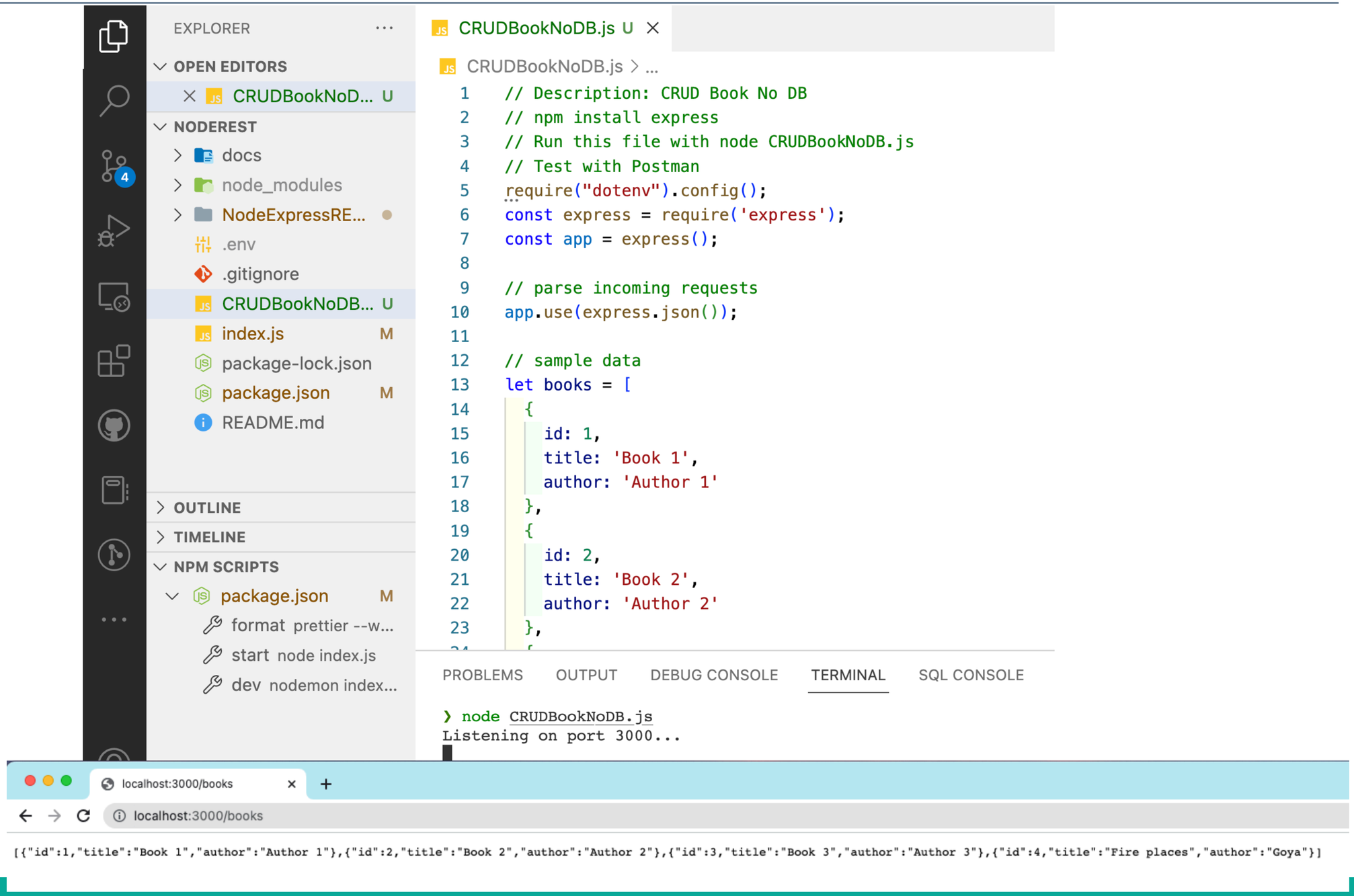
What we will learn?



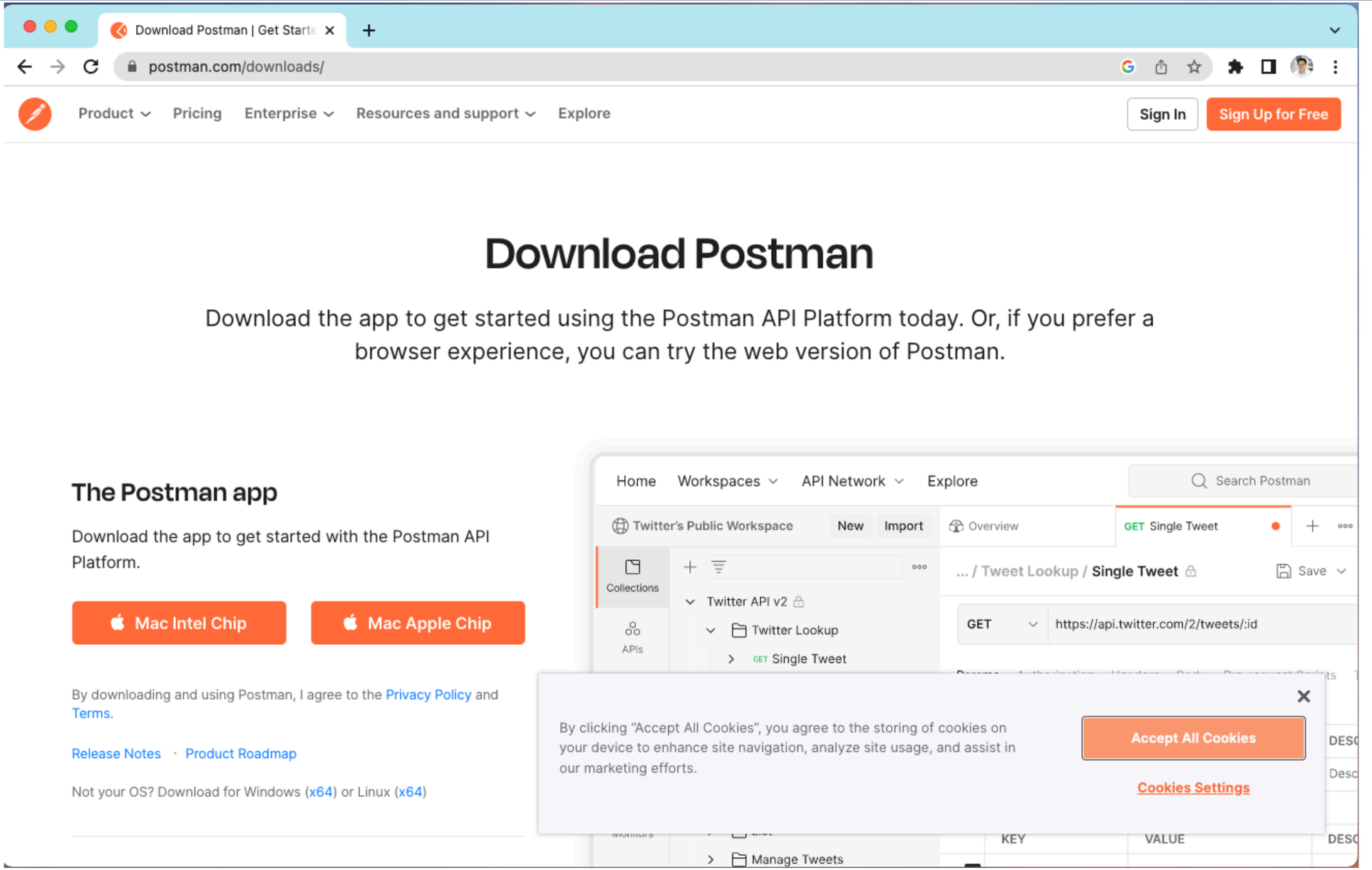

EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
dev nodemon index...
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
// Description: CRUD Book No DB
// npm install express
// Run this file with node CRUDBookNoDB.js
// Test with Postman
require("dotenv").config();
const express = require('express');
const app = express();
// parse incoming requests
app.use(express.json());
// sample data
let books = [
{
id: 1,
title: 'Book 1',
author: 'Author 1'
},
{
id: 2,
title: 'Book 2',
author: 'Author 2'
},
{
id: 3,
title: 'Book 3',
author: 'Author 3'
}
];


EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
30
31 // route to get all books
32 app.get('/books', (req, res) => {
33 res.json(books);
34 });
35
36 // route to get a book by id
37 app.get('/books/:id', (req, res) => {
38 const book = books.find(b => b.id === parseInt(req.params.id));
39 if (!book) res.status(404).send('Book not found');
40 res.json(book);
41 });
42
43 // route to create a new book
44 app.post('/books', (req, res) => {
45 const book = {
46 id: books.length + 1,
47 title: req.body.title,
48 author: req.body.author
49 };
50 books.push(book);
51 res.send(book);
52 });
53

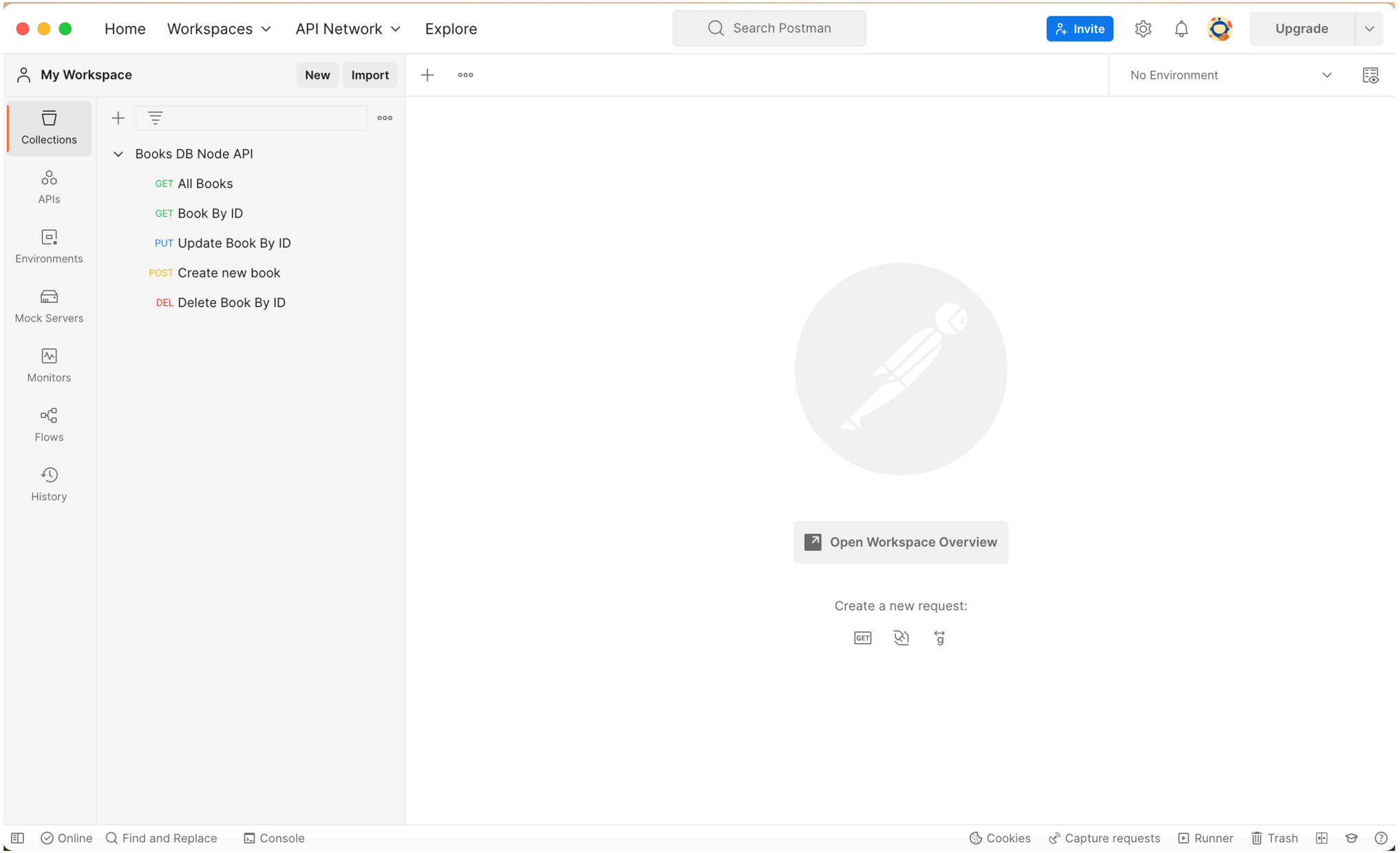

EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
53
54 // route to update a book
55 app.put('/books/:id', (req, res) => {
56 const book = books.find(b => b.id === parseInt(req.params.id));
57 if (!book) res.status(404).send('Book not found');
58 book.title = req.body.title;
59 book.author = req.body.author;
60 res.send(book);
61 });
62
63 // route to delete a book
64 app.delete('/books/:id', (req, res) => {
65 const book = books.find(b => b.id === parseInt(req.params.id));
66 if (!book) res.status(404).send('Book not found');
67 const index = books.indexOf(book);
68 books.splice(index, 1);
69 res.send(book);
70 });
71
72 const port = process.env.PORT || 3000;
73 app.listen(port, () => console.log(`Listening on port ${port}...`));
74


EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
dev nodemon index...
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
1 // Description: CRUD Book No DB
2 // npm install express
3 // Run this file with node CRUDBookNoDB.js
4 // Test with Postman
5 require("dotenv").config();
6 const express = require('express');
7 const app = express();
8
9 // parse incoming requests
10 app.use(express.json());
11
12 // sample data
13 let books = [
14 {
15 id: 1,
16 title: 'Book 1',
17 author: 'Author 1'
18 },
19 {
20 id: 2,
21 title: 'Book 2',
22 author: 'Author 2'
23 },
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
❯ node CRUDBookNoDB.js
Listening on port 3000...
localhost:3000/books
localhost:3000/books
[{"id":1,"title":"Book 1","author":"Author 1"},{"id":2,"title":"Book 2","author":"Author 2"},{"id":3,"title":"Book 3","author":"Author 3"},{"id":4,"title":"Fire places","author":"Goya"}]

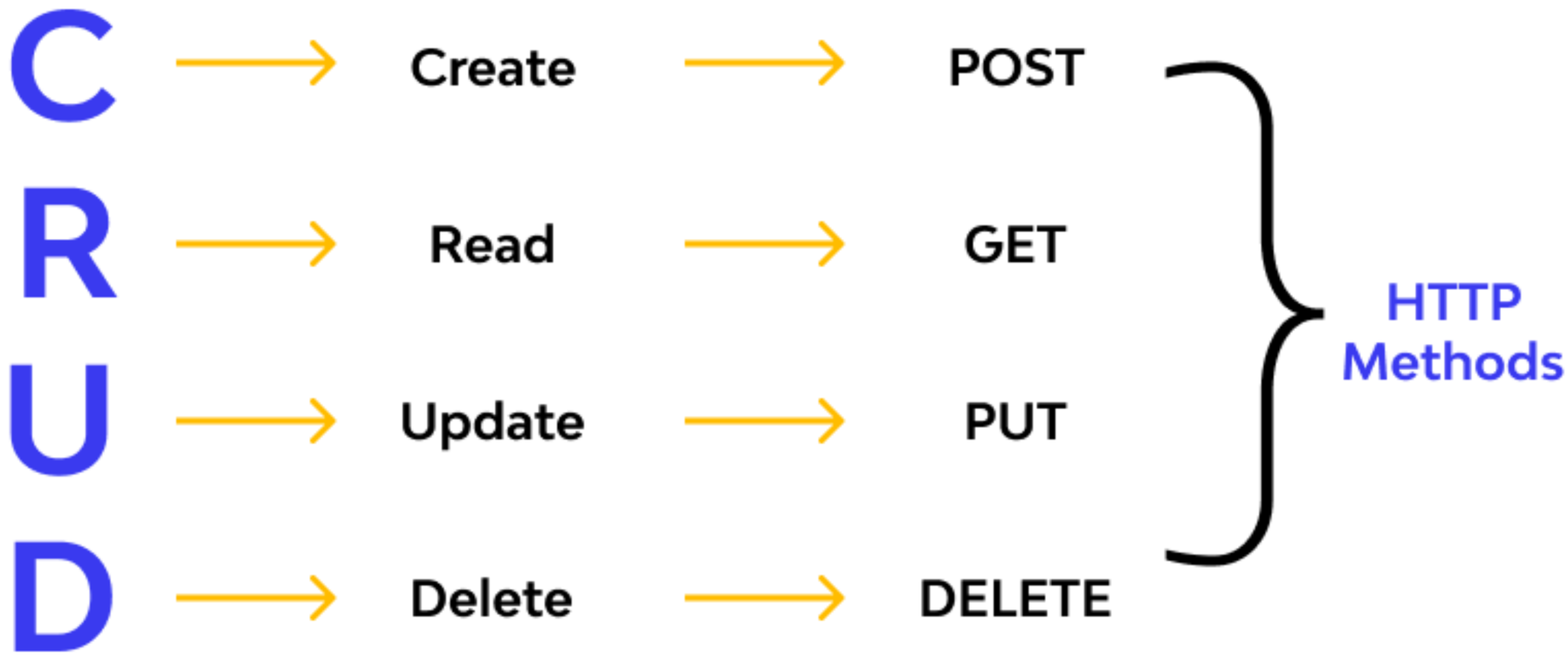
POSTMAN

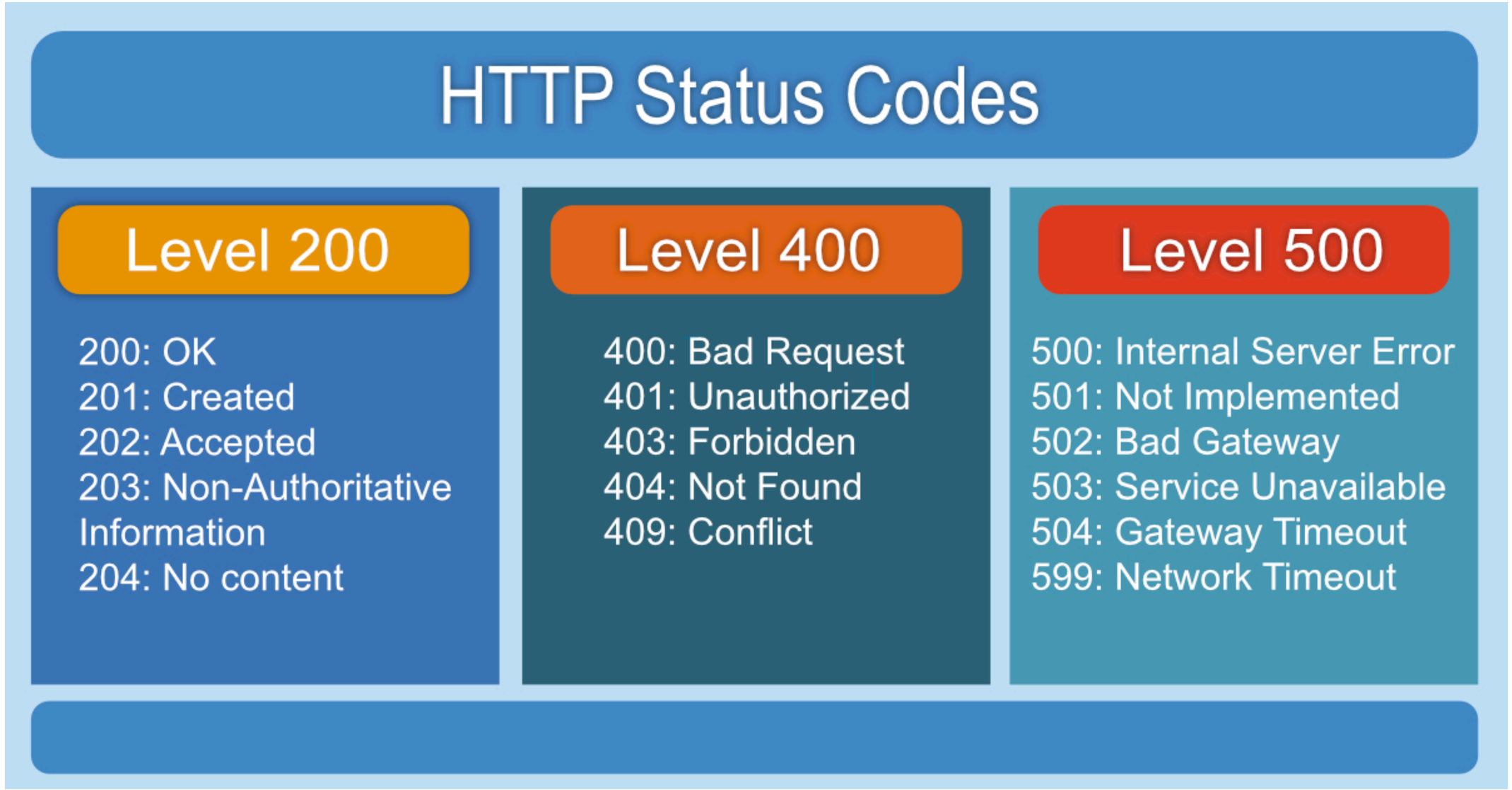


https://www.postman.com/downloads/


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Open Workspace Overview
Create a new request:
Online
Find and Replace
Console
Cookies
Capture requests
Runner
Trash

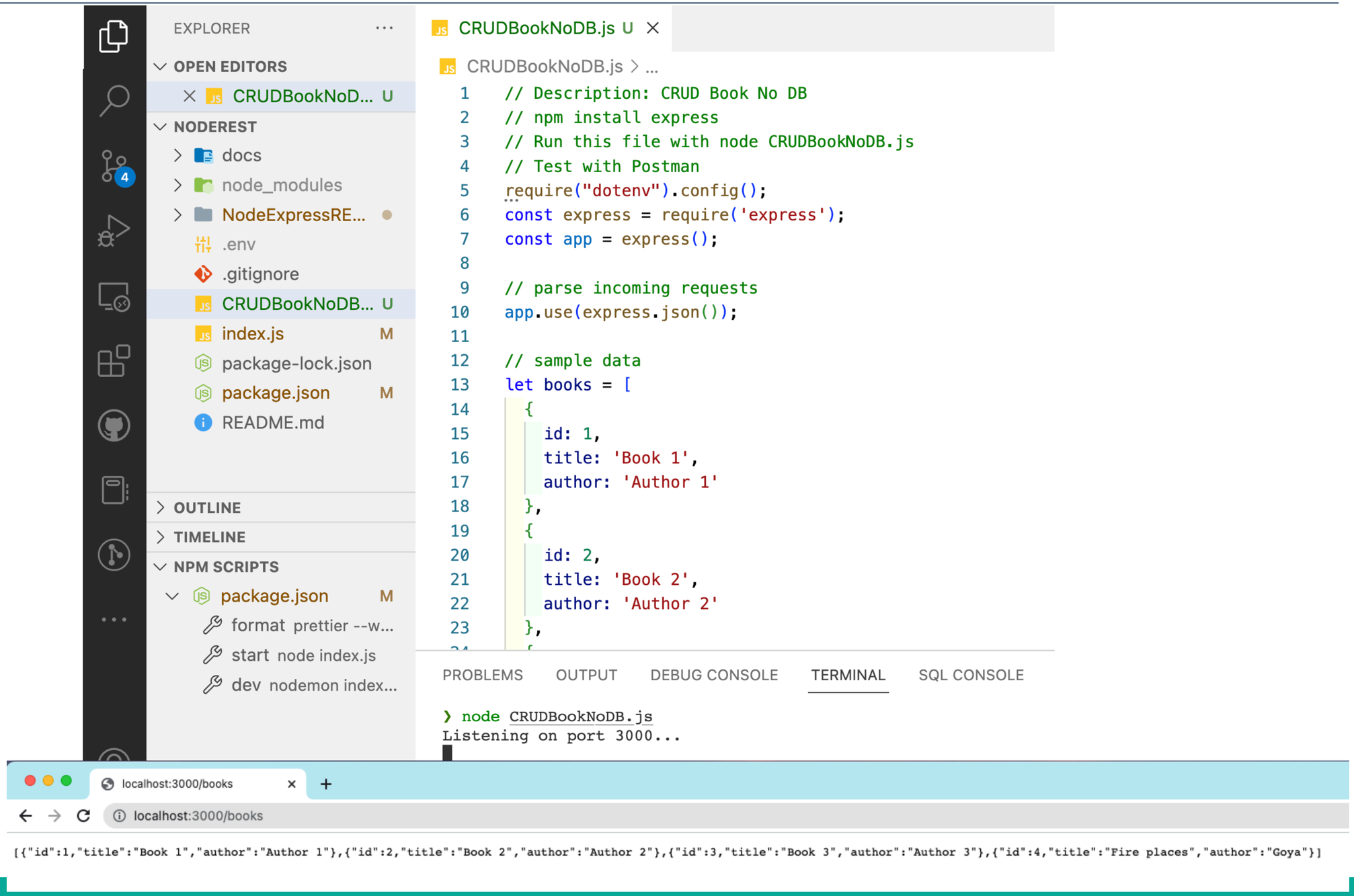
CRUD

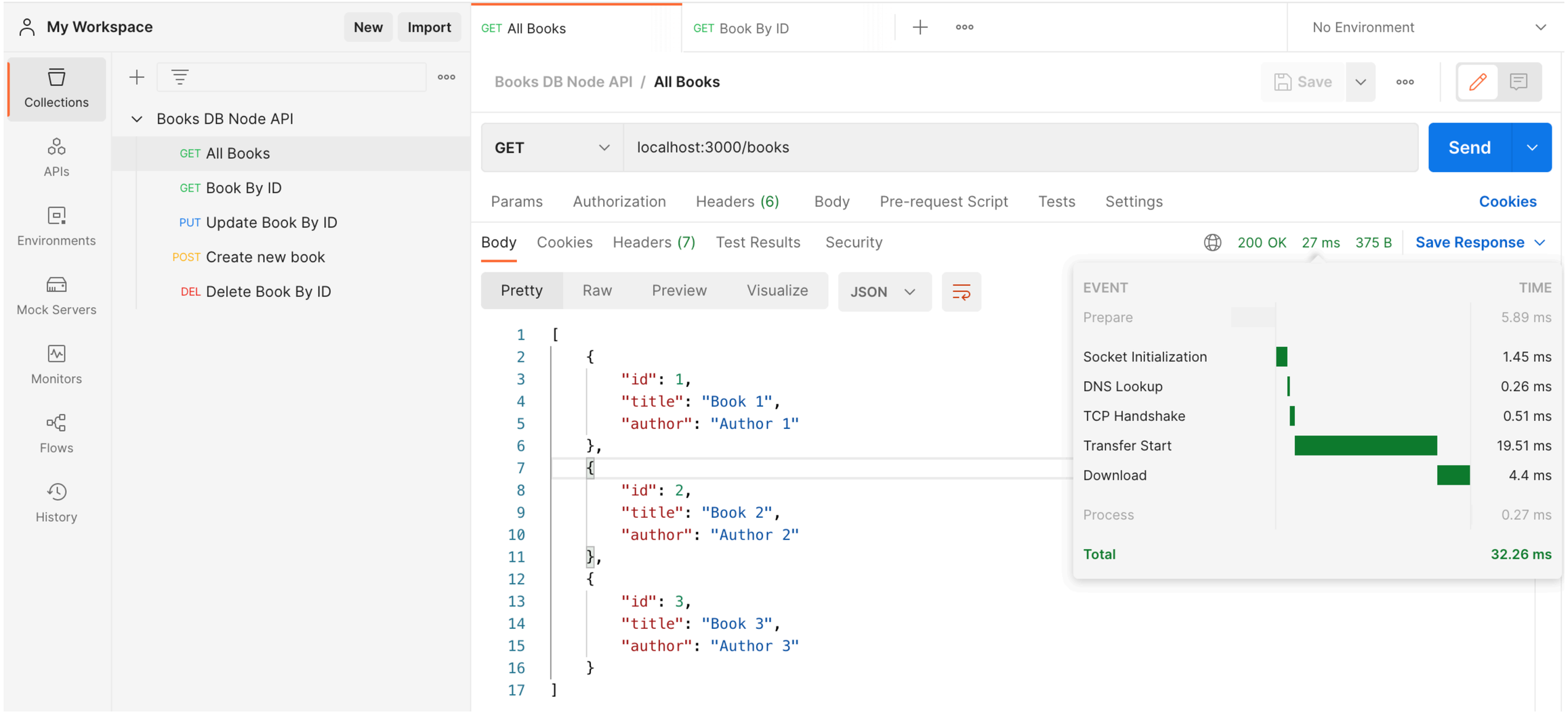

C
Create
POST
R
Read
GET
U
Update
PUT
D
Delete
DELETE
HTTP Methods

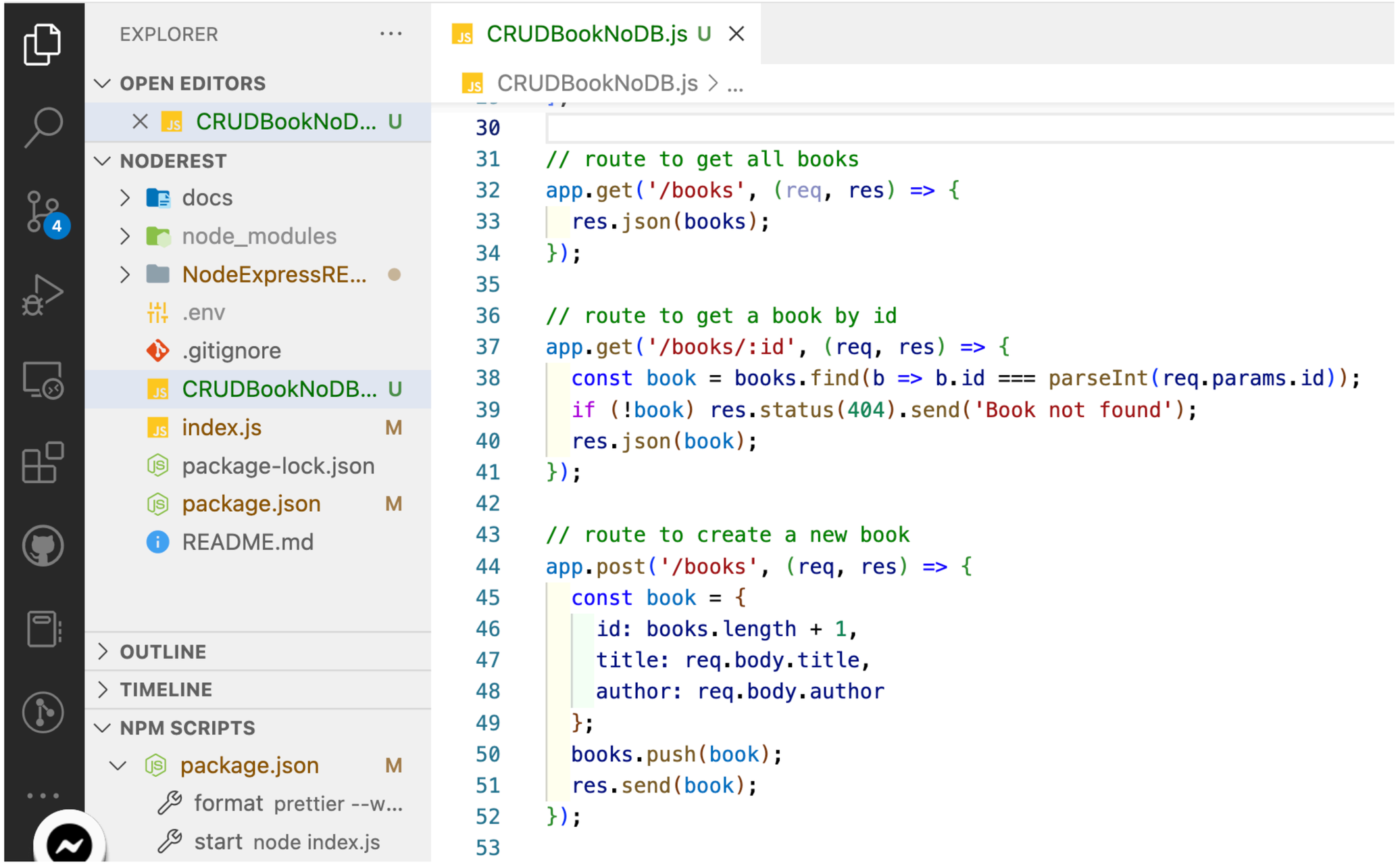


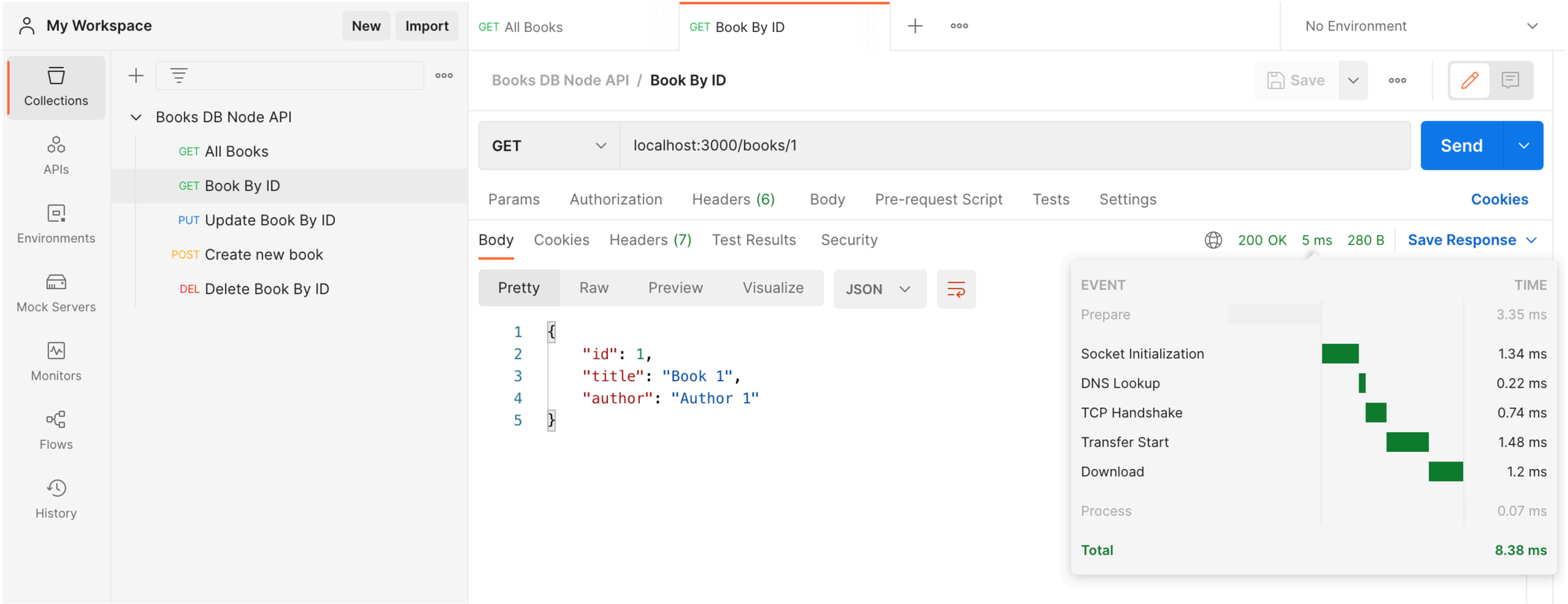
https://tips.thaiware.com/1077.html

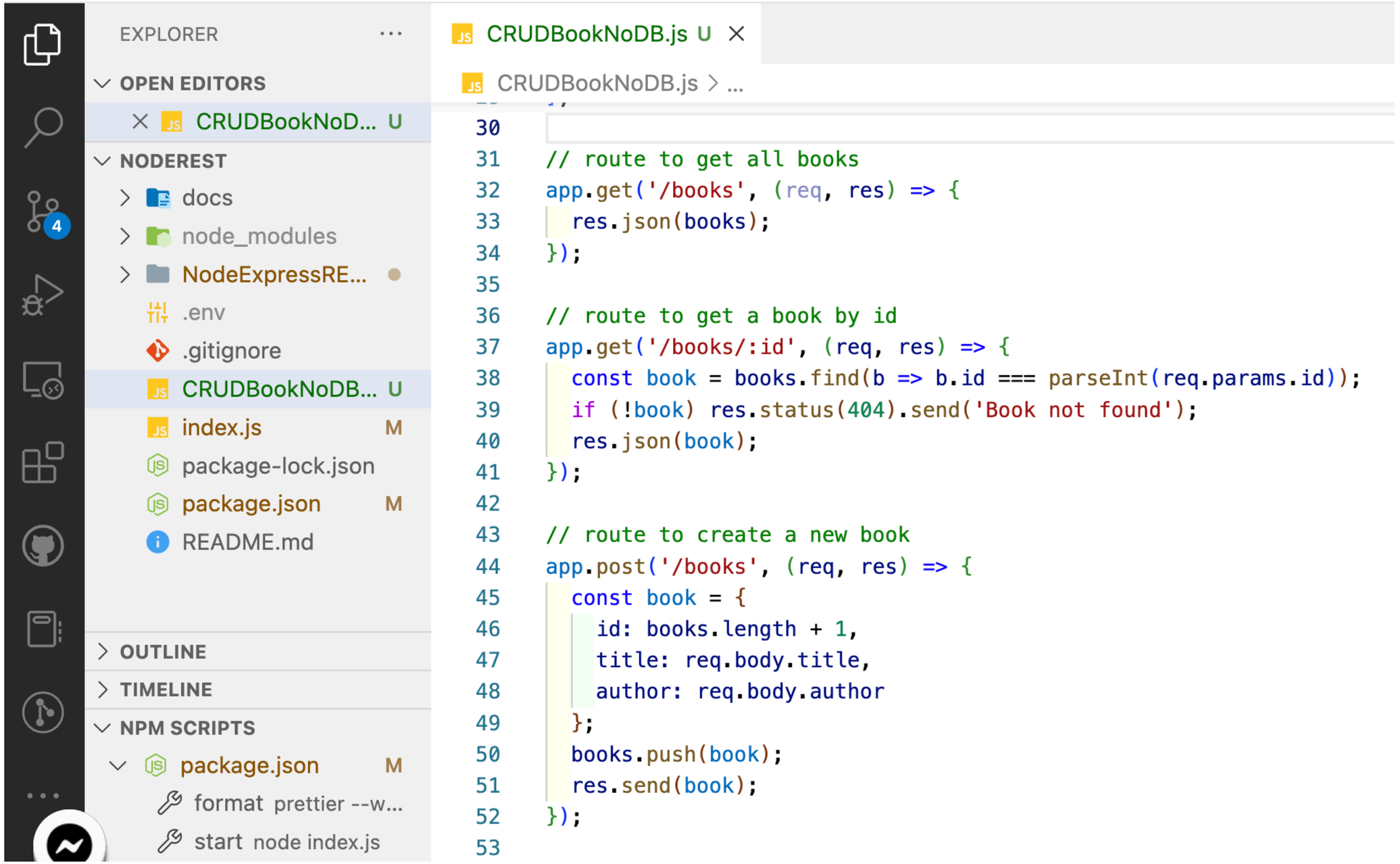
READ


EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
dev nodemon index...
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
1 // Description: CRUD Book No DB
2 // npm install express
3 // Run this file with node CRUDBookNoDB.js
4 // Test with Postman
5 require("dotenv").config();
6 const express = require('express');
7 const app = express();
8
9 // parse incoming requests
10 app.use(express.json());
11
12 // sample data
13 let books = [
14 {
15 id: 1,
16 title: 'Book 1',
17 author: 'Author 1'
18 },
19 {
20 id: 2,
21 title: 'Book 2',
22 author: 'Author 2'
23 },
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
SQL CONSOLE
❯ node CRUDBookNoDB.js
Listening on port 3000...
localhost:3000/books
localhost:3000/books
[{"id":1,"title":"Book 1","author":"Author 1"},{"id":2,"title":"Book 2","author":"Author 2"},{"id":3,"title":"Book 3","author":"Author 3"},{"id":4,"title":"Fire places","author":"Goya"}]


My Workspace
New
Import
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
GET All Books
GET Book By ID
No Environment
Books DB Node API / All Books
Save
GET
localhost:3000/books
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Body
Cookies
Headers (7)
Test Results
Security
200 OK
27 ms
375 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 1,
"title": "Book 1",
"author": "Author 1"
},
{
"id": 2,
"title": "Book 2",
"author": "Author 2"
},
{
"id": 3,
"title": "Book 3",
"author": "Author 3"
}
]
EVENT
TIME
Prepare
5.89 ms
Socket Initialization
1.45 ms
DNS Lookup
0.26 ms
TCP Handshake
0.51 ms
Transfer Start
19.51 ms
Download
4.4 ms
Process
0.27 ms
Total
32.26 ms

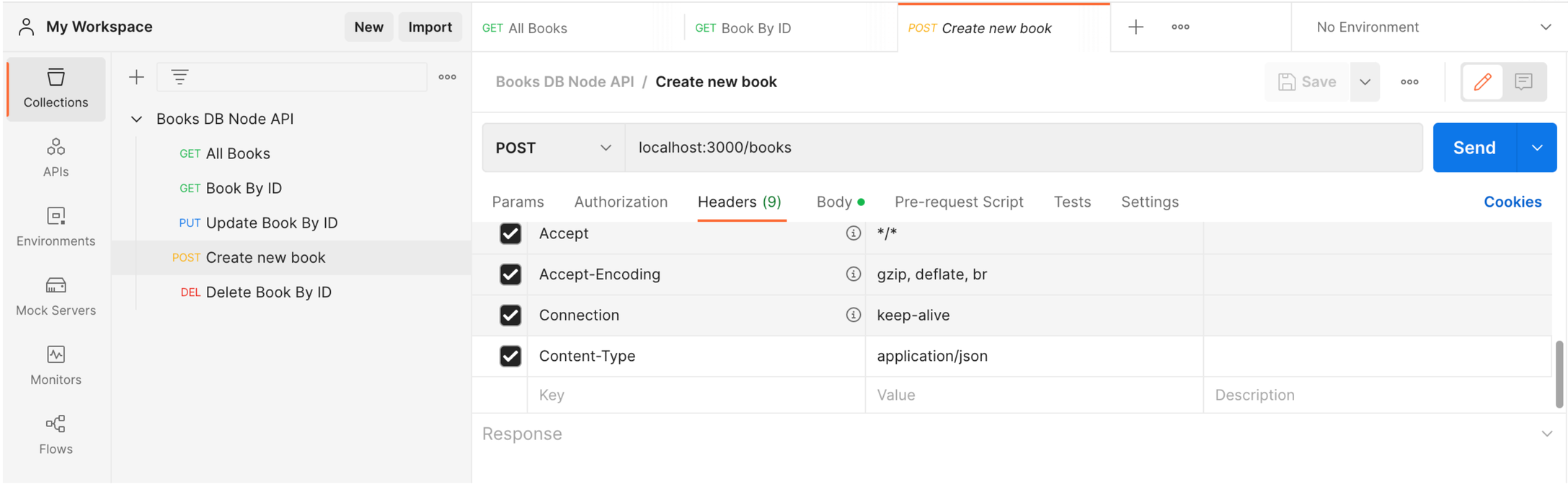

EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
30
31 // route to get all books
32 app.get('/books', (req, res) => {
33 res.json(books);
34 });
35
36 // route to get a book by id
37 app.get('/books/:id', (req, res) => {
38 const book = books.find(b => b.id === parseInt(req.params.id));
39 if (!book) res.status(404).send('Book not found');
40 res.json(book);
41 });
42
43 // route to create a new book
44 app.post('/books', (req, res) => {
45 const book = {
46 id: books.length + 1,
47 title: req.body.title,
48 author: req.body.author
49 };
50 books.push(book);
51 res.send(book);
52 });
53


My Workspace
New
Import
GET All Books
GET Book By ID
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Book By ID
Save
GET
localhost:3000/books/1
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Body
Cookies
Headers (7)
Test Results
Security
200 OK
5 ms
280 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"id": 1,
"title": "Book 1",
"author": "Author 1"
}
EVENT
TIME
Prepare
3.35 ms
Socket Initialization
1.34 ms
DNS Lookup
0.22 ms
TCP Handshake
0.74 ms
Transfer Start
1.48 ms
Download
1.2 ms
Process
0.07 ms
Total
8.38 ms

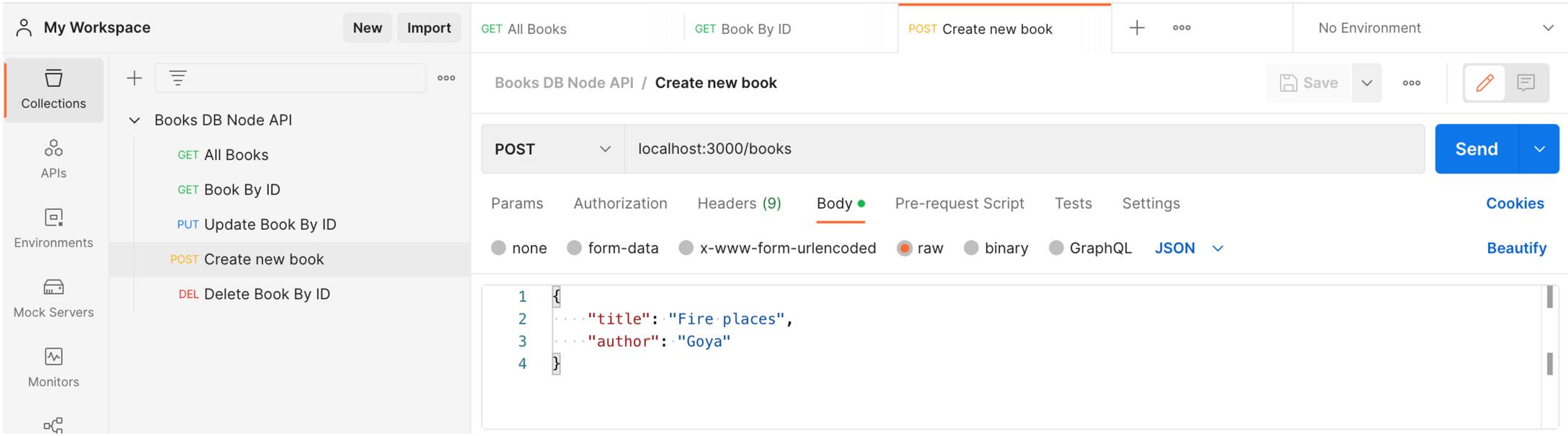

EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
30
31 // route to get all books
32 app.get('/books', (req, res) => {
33 res.json(books);
34 });
35
36 // route to get a book by id
37 app.get('/books/:id', (req, res) => {
38 const book = books.find(b => b.id === parseInt(req.params.id));
39 if (!book) res.status(404).send('Book not found');
40 res.json(book);
41 });
42
43 // route to create a new book
44 app.post('/books', (req, res) => {
45 const book = {
46 id: books.length + 1,
47 title: req.body.title,
48 author: req.body.author
49 };
50 books.push(book);
51 res.send(book);
52 });
53

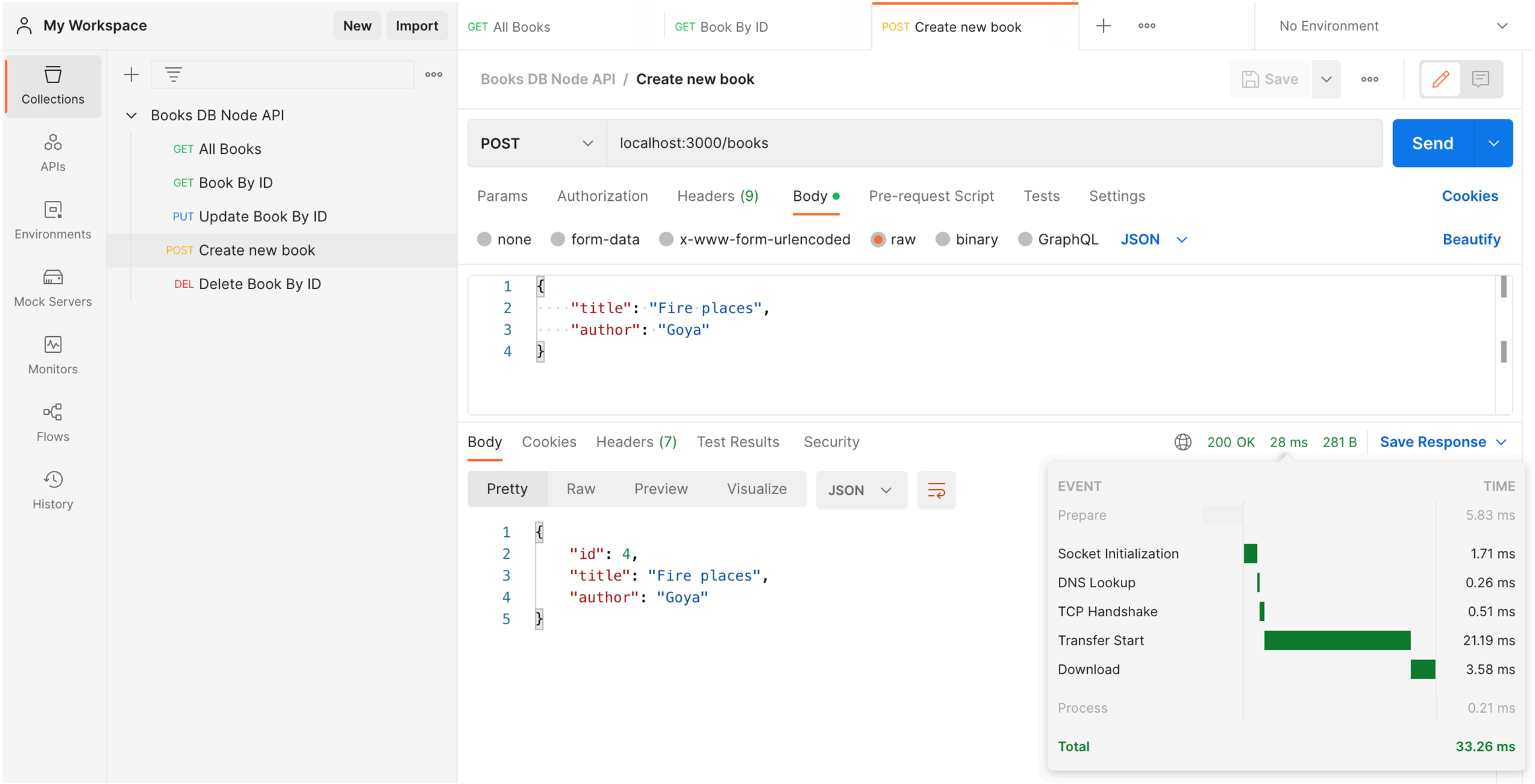
CREAT

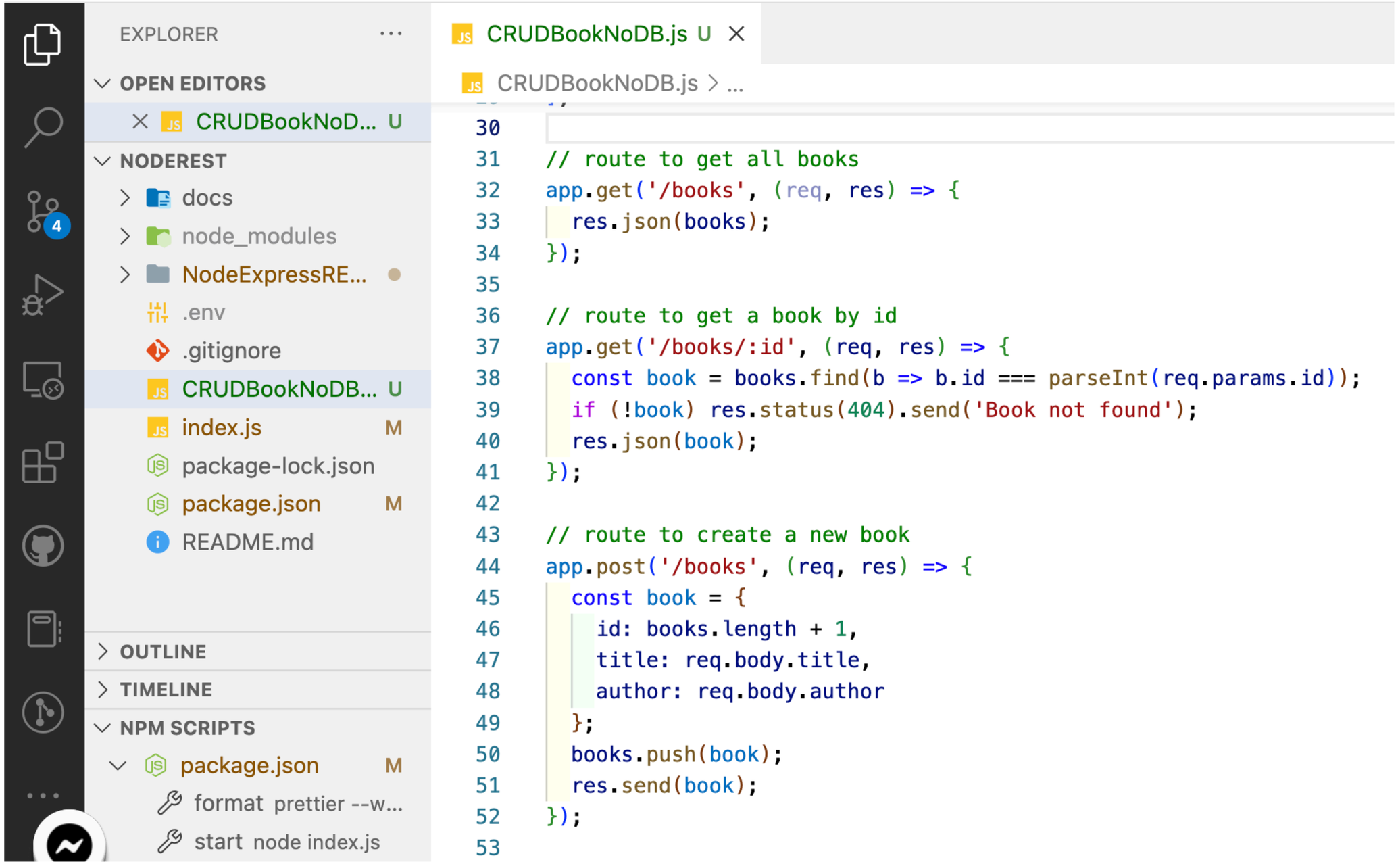
Step 1


My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new book
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Create new book
Save
POST
localhost:3000/books
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
Accept
/
Accept-Encoding
gzip, deflate, br
Connection
keep-alive
Content-Type
application/json
Key
Value
Description
Response

Step 2

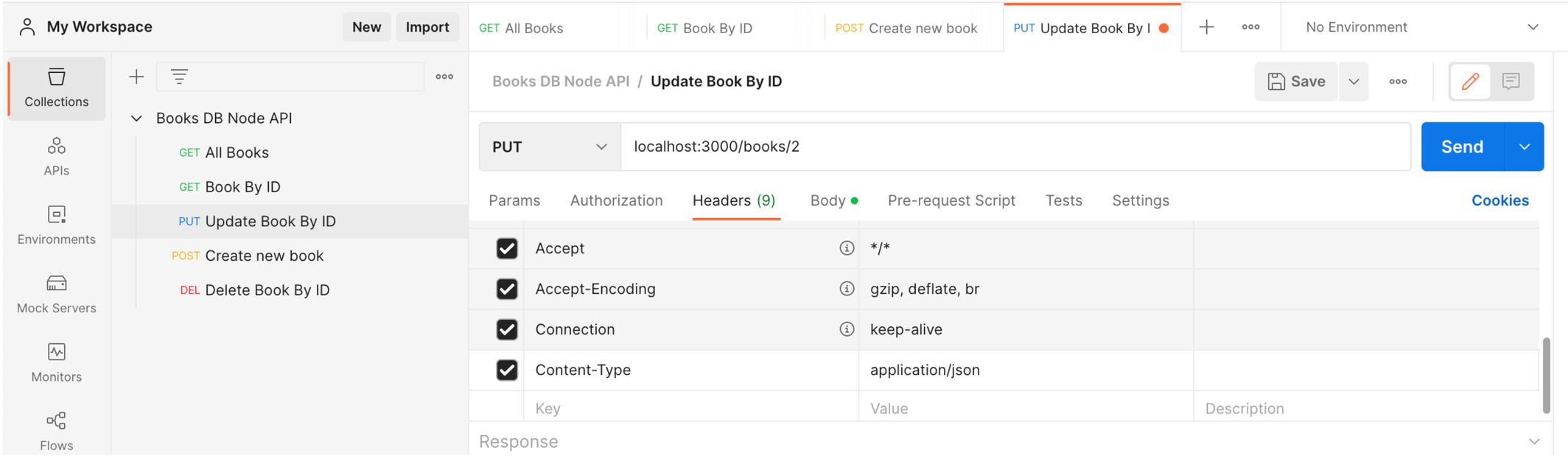

My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new book
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Create new book
Save
POST
localhost:3000/books
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"title": "Fire places",
"author": "Goya"
}


My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new book
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Create new book
Save
POST
localhost:3000/books
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"title": "Fire places",
"author": "Goya"
}
Body
Cookies
Headers (7)
Test Results
Security
200 OK
28 ms
281 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"id": 4,
"title": "Fire places",
"author": "Goya"
}
EVENT
TIME
Prepare
5.83 ms
Socket Initialization
1.71 ms
DNS Lookup
0.26 ms
TCP Handshake
0.51 ms
Transfer Start
21.19 ms
Download
3.58 ms
Process
0.21 ms
Total
33.26 ms

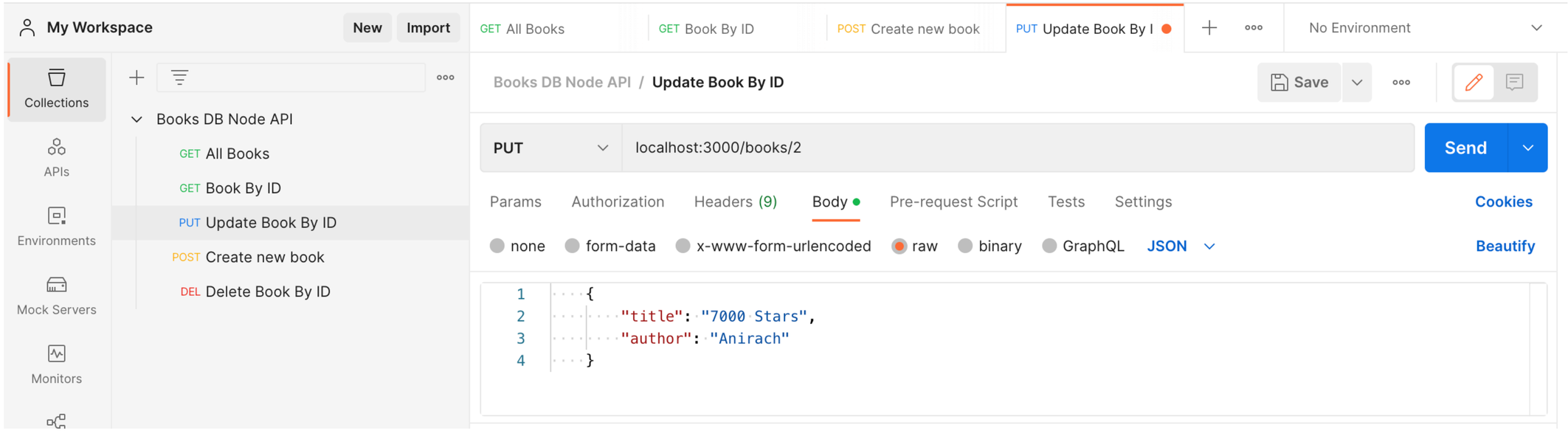

EXPLORER
OPEN EDITORS
CRUDBookNoD... U
NODEREST
docs
node_modules
NodeExpressRE...
.env
.gitignore
CRUDBookNoDB... U
index.js M
package-lock.json
package.json M
README.md
OUTLINE
TIMELINE
NPM SCRIPTS
package.json M
format prettier --w...
start node index.js
CRUDBookNoDB.js U
CRUDBookNoDB.js > ...
30
31 // route to get all books
32 app.get('/books', (req, res) => {
33 res.json(books);
34 });
35
36 // route to get a book by id
37 app.get('/books/:id', (req, res) => {
38 const book = books.find(b => b.id === parseInt(req.params.id));
39 if (!book) res.status(404).send('Book not found');
40 res.json(book);
41 });
42
43 // route to create a new book
44 app.post('/books', (req, res) => {
45 const book = {
46 id: books.length + 1,
47 title: req.body.title,
48 author: req.body.author
49 };
50 books.push(book);
51 res.send(book);
52 });
53

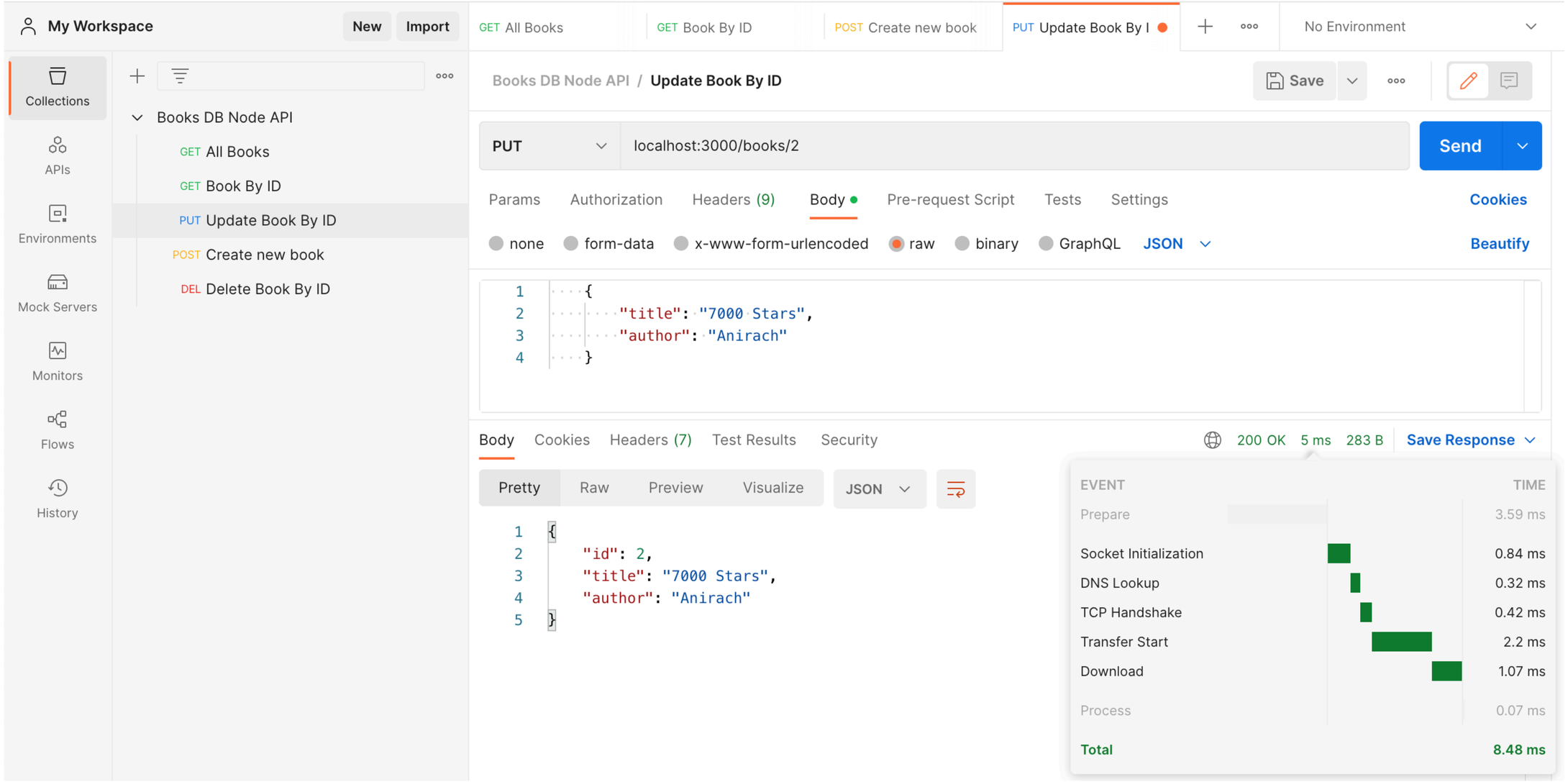
UPDATE

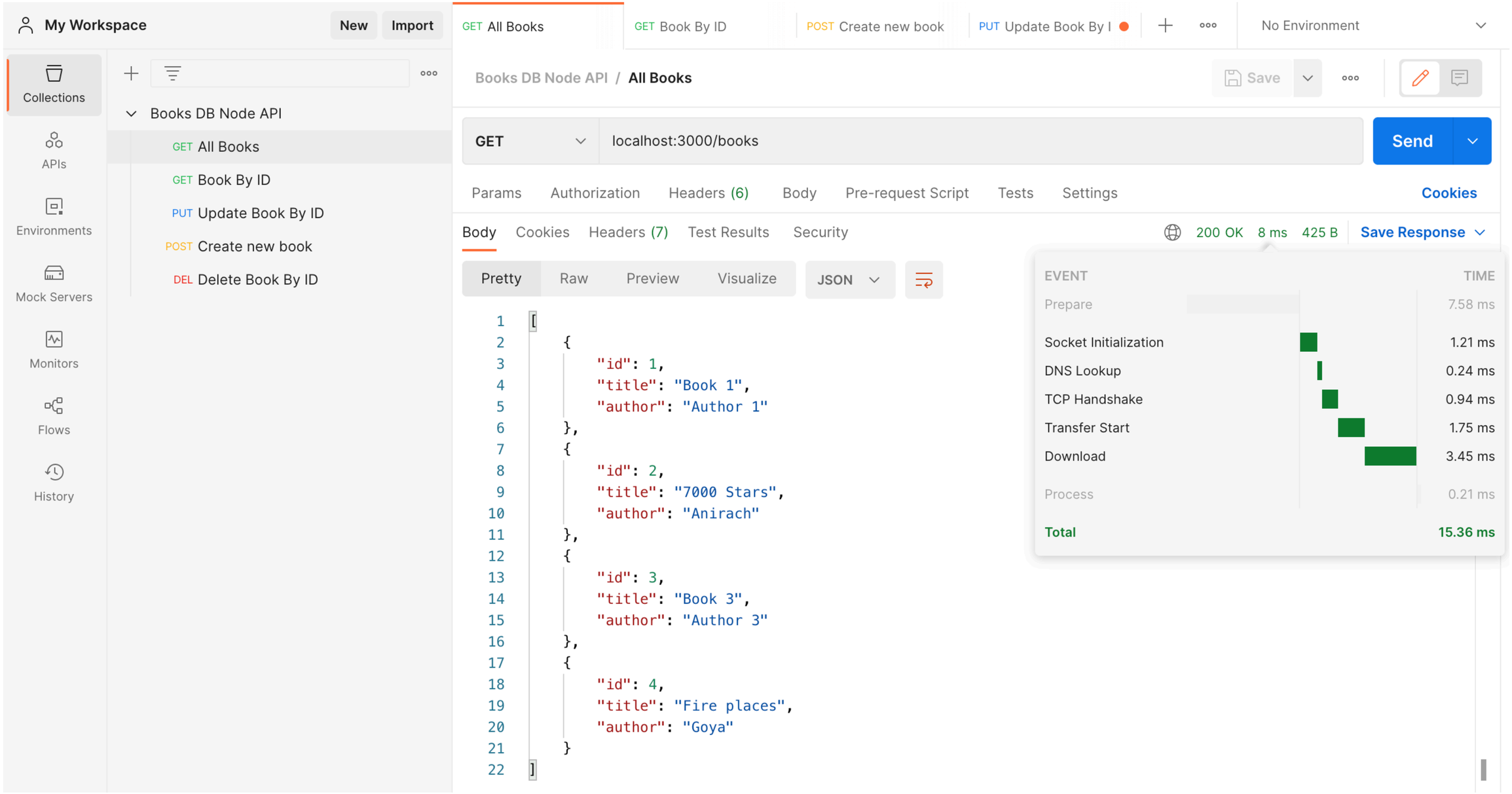
Step 1


My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new book
PUT Update Book By I
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Update Book By ID
Save
PUT
localhost:3000/books/2
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
Accept
/
Accept-Encoding
gzip, deflate, br
Connection
keep-alive
Content-Type
application/json
Key
Value
Description
Response

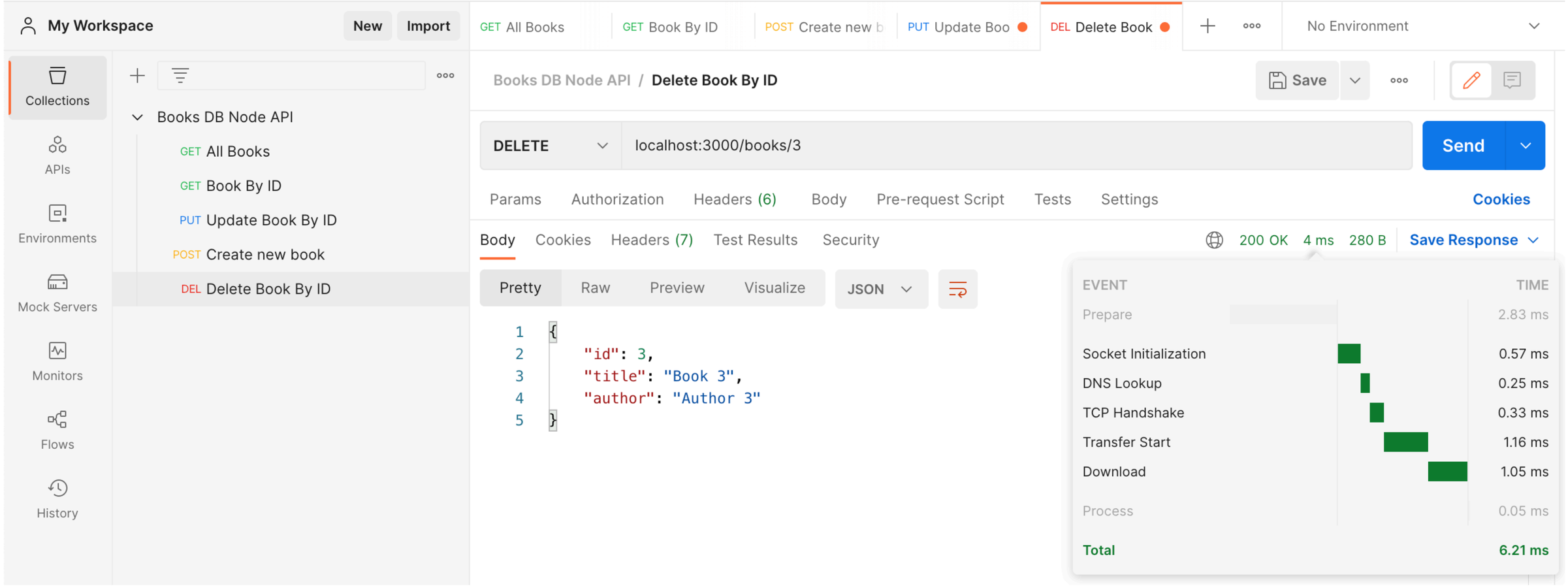
Step 2

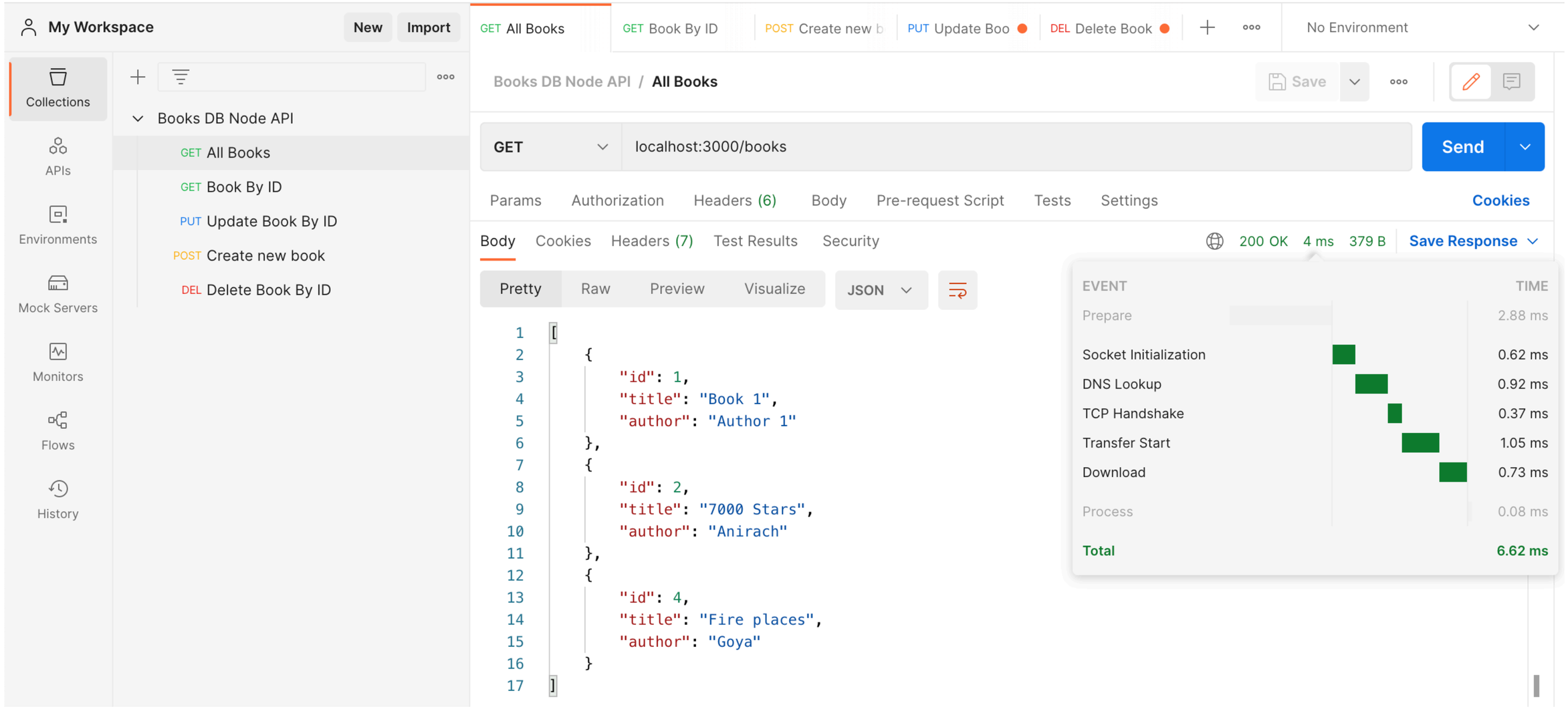

My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new book
PUT Update Book By I
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Update Book By ID
Save
PUT
localhost:3000/books/2
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
1 {
2 "title": "7000 Stars",
3 "author": "Anirach"
4 }


My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new book
PUT Update Book By I
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Update Book By ID
Save
PUT
localhost:3000/books/2
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"title": "7000 Stars",
"author": "Anirach"
}
Body
Cookies
Headers (7)
Test Results
Security
200 OK
5 ms
283 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"id": 2,
"title": "7000 Stars",
"author": "Anirach"
}
EVENT
TIME
Prepare
3.59 ms
Socket Initialization
0.84 ms
DNS Lookup
0.32 ms
TCP Handshake
0.42 ms
Transfer Start
2.2 ms
Download
1.07 ms
Process
0.07 ms
Total
8.48 ms


My Workspace
New
Import
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
GET All Books
GET Book By ID
POST Create new book
PUT Update Book By I
No Environment
Books DB Node API / All Books
Save
GET
localhost:3000/books
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Body
Cookies
Headers (7)
Test Results
Security
200 OK
8 ms
425 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 1,
"title": "Book 1",
"author": "Author 1"
},
{
"id": 2,
"title": "7000 Stars",
"author": "Anirach"
},
{
"id": 3,
"title": "Book 3",
"author": "Author 3"
},
{
"id": 4,
"title": "Fire places",
"author": "Goya"
}
]
EVENT
TIME
Prepare
7.58 ms
Socket Initialization
1.21 ms
DNS Lookup
0.24 ms
TCP Handshake
0.94 ms
Transfer Start
1.75 ms
Download
3.45 ms
Process
0.21 ms
Total
15.36 ms

DELETE


My Workspace
New
Import
GET All Books
GET Book By ID
POST Create new b
PUT Update Boo
DEL Delete Book
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
Books DB Node API / Delete Book By ID
Save
DELETE
localhost:3000/books/3
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Body
Cookies
Headers (7)
Test Results
Security
200 OK
4 ms
280 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1 {
2 "id": 3,
3 "title": "Book 3",
4 "author": "Author 3"
5 }
EVENT
TIME
Prepare
2.83 ms
Socket Initialization
0.57 ms
DNS Lookup
0.25 ms
TCP Handshake
0.33 ms
Transfer Start
1.16 ms
Download
1.05 ms
Process
0.05 ms
Total
6.21 ms


My Workspace
New
Import
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Books DB Node API
GET All Books
GET Book By ID
PUT Update Book By ID
POST Create new book
DEL Delete Book By ID
GET All Books
GET Book By ID
POST Create new b
PUT Update Boo
DEL Delete Book
No Environment
Books DB Node API / All Books
Save
GET
localhost:3000/books
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Body
Cookies
Headers (7)
Test Results
Security
200 OK
4 ms
379 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1 [
2 {
3 "id": 1,
4 "title": "Book 1",
5 "author": "Author 1"
6 },
7 {
8 "id": 2,
9 "title": "7000 Stars",
10 "author": "Anirach"
11 },
12 {
13 "id": 4,
14 "title": "Fire places",
15 "author": "Goya"
16 }
17]
EVENT
TIME
Prepare
2.88 ms
Socket Initialization
0.62 ms
DNS Lookup
0.92 ms
TCP Handshake
0.37 ms
Transfer Start
1.05 ms
Download
0.73 ms
Process
0.08 ms
Total
6.62 ms

THANK YOU

- 3-TIERS Concepts
- API Architect
- Postman
- CRUD with Postman