

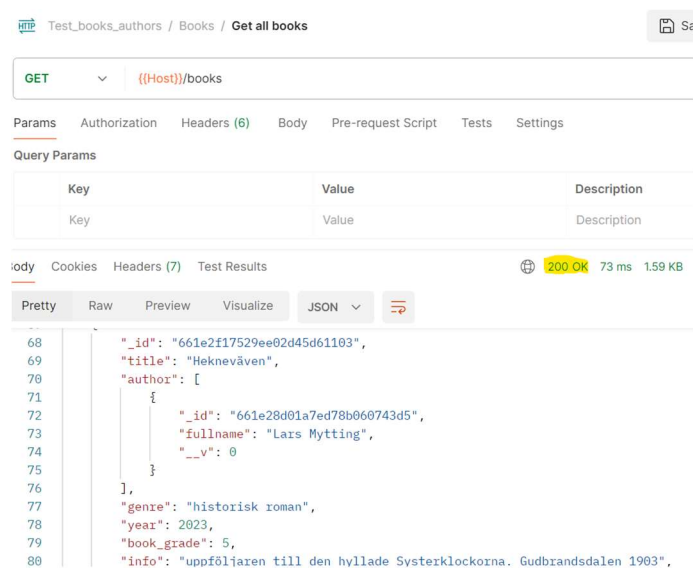
Rapport testprocess Bookfinder API

Test 1. Verify that the API returns the correct HTTP status code (e.g., 200 OK) for a successful GET request.

- Manuellt test

Hämtar alla böcker med en GET-request i Postman och förväntas att statuskod 200 visas om request lyckas.

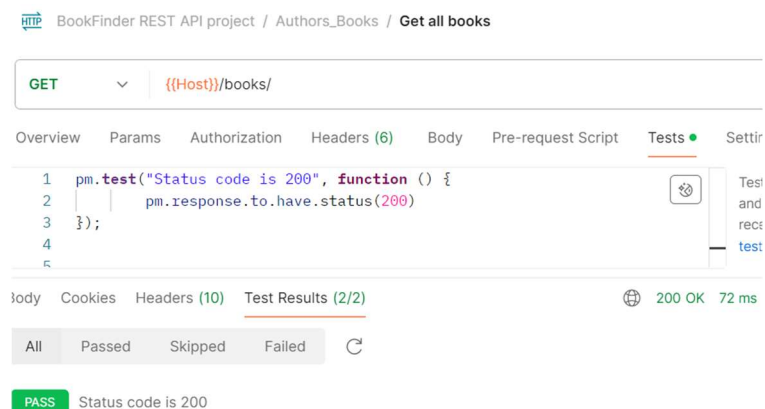
Resultat: Statuskod 200 OK.



- Automatiserat test

Testscript för att kontrollera att statuskod 200 returneras för lyckad GET request som förväntat.

Resultat: PASS för "Status code is 200".



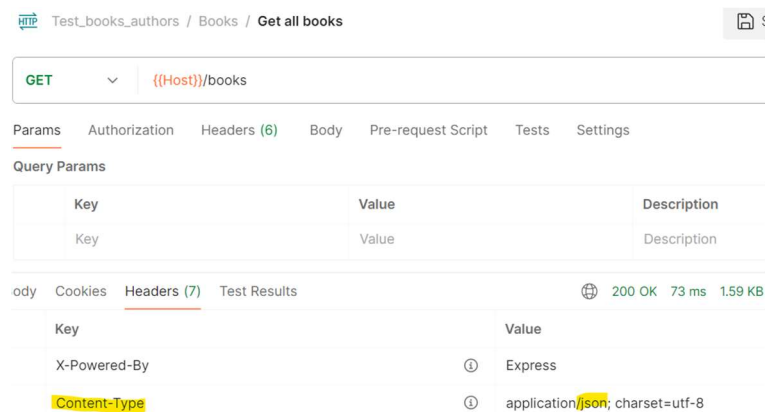
Test 2. Check if the API returns the expected data format (e.g., JSON, XML in the response).

- Manuellt test

Hämtar alla böcker med en GET-request och kontrollera om det i Headers och Content-Type visas det förväntade dataformatet.

Resultat: application/json visas i Headers.

Statuskod 200 OK

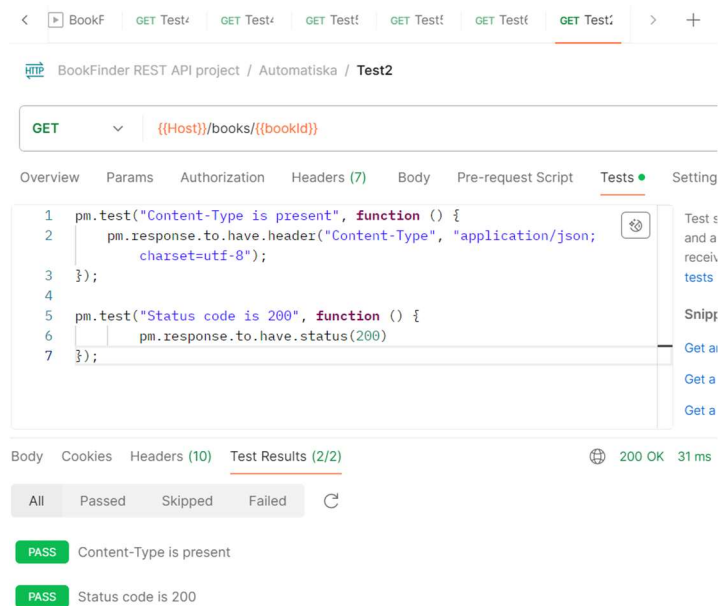


- Automatiserat test

Testscript i GET-request för att kontrollera att Headers "Content type key" med value application/json finns tillgängligt som förväntat.

Resultat: PASS för Content-Type is present.

PASS Status Code is 200



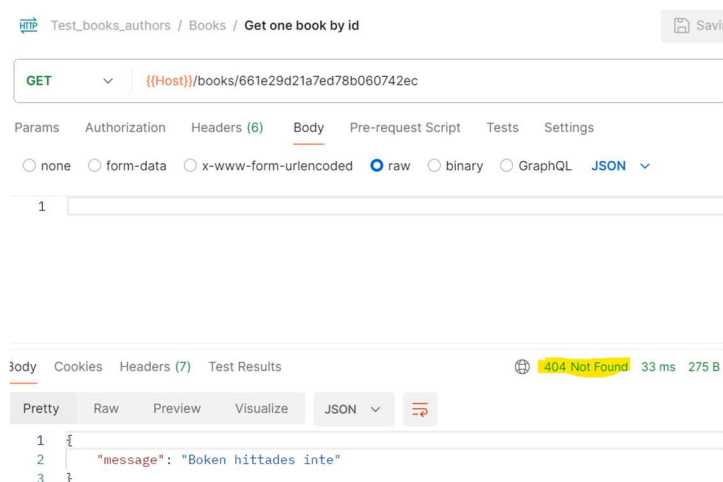
Test 3. Ensure that the API returns the correct http status code (e.g., 400 Bad Request) for an invalid request.

- Manuellt test

Hämtar en bok med en GET-request och id som inte finns tillgängligt och ska då få den förväntade felkoden och meddelande om att boken inte finns.

Resultat: 404 "Not Found".
message: "Boken hittades inte"

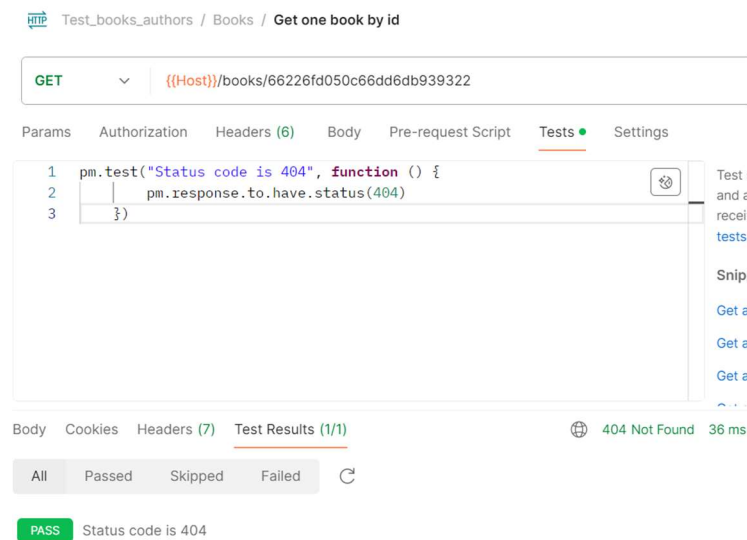
(Jag valde statuskod 404 i detta testfall eftersom statuskod 400 testas av i test nr.12)



- Automatiserat test

GET-request på id som inte finns tillgängligt. Statuskod 404 ska visas och då förväntas testet vara passerat.

Resultat: PASS "404 Not Found"
message: "Boken hittades inte"



Test 4. Test if the API returns the correct data when querying with specific filters or search criteria.

- Manuellt test

GET-request och id alternativt `{{bookId}}` för att hämta senast inlagda bok. Sätter parametern `fields` för att kunna söka på exempelvis `title`, `year` eller `info`. Test förväntas passera om titel visas och statuskod är 200.

Resultat: Titel presenteras i body
Statuskod 200 OK

Test_books_authors / Books / Filtering

GET `{{Host}}/books/661e7b56b614e0593a73b2d1?fields=title`

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

<input checked="" type="checkbox"/> Key	Value	Description
<input checked="" type="checkbox"/> fields	title	
Key	Value	Description

Body Cookies Headers (7) Test Results 200 OK 31 ms 298 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "_id": "661e7b56b614e0593a73b2d1",
3   "title": "Pippi Långstrump2"
4 }
```

- Automatiserat test

GET request mot bok id (alternativt `{{bookId}}`) för att göra filtrering på senast inlagda bok). Sätter parametern `fields` för att kunna söka på exempelvis titel i detta fall. Förväntas att titel visas i body och att test passerar och visar statuskod 200.

Resultat: test passerar och titel för den valda boken presenteras i body.
"Object only contains allowed keys" visas samt status code is 200.

BookFinder REST API project / Automatiska / Test4

GET `{{Host}}/books/{{bookId}}?fields=title`

Overview Params Authorization Headers (6) Body Pre-request Script Tests Settings

```
1 pm.test("Object only contains allowed keys", function () {
2   const notesJsonStructure = ['title'];
3   const responseObj = pm.response.json();
4   console.log(responseObj);
5   _.each(responseObj.response, (item) => {
6     pm.expect(item).to.be.oneOf(notesJsonStructure);
7   });
8 });
9
10 pm.test("Status code is 200", function () {
11   pm.response.to.have.status(200);
12 });
13
```

Test s and ar receiv tests : Snipp Get ar Get a Get a Get a

Body Cookies Headers (10) Test Results (2/2) 200 OK 30 ms

All Passed Skipped Failed

PASS Object only contains allowed keys

PASS Status code is 200

Test 5. Verify that the API returns paginated results when a large number of records are requested.

- Manuellt test –

Hämtar alla böcker med en GET-request (api/books/all), använder query parametrarna page med value = 1 och limit med value = 5. Förväntas visa 5 böcker per sida i body och längts ner på sidan ska currentPage, totalPages och totalBooks siffrorna presenteras.

Resultat: 5 böcker visas på första sidan.
Statuskod 200 OK

- Automatiserat test –

Hämtar alla böcker med en GET-request (api/books/all), använder query parametrarna page med value = 1 och limit med value = 10.
Förväntat resultat är att test passerar och att samtliga böcker visas i body och längts ner på sidan ska currentPage, totalPages och totalBooks siffrorna presenteras.

Resultat: se nedan skärmdumpar där förväntat resultat visas
PASS "Correct number of results per page"
PASS Status code is 200

HTTP BookFinder REST API project / Automatiska / Test5

GET `{{Host}}/books/all?page=1&limit=10`

Overview Params Authorization Headers (6) Body Pre-request Script Tests • Settings

```
1
2 pm.test("Correct number of results per page", function () {
3     const resultsPerPage = 10;
4     const jsonData = pm.response.json();
5     const results = jsonData.books.length;
6     pm.expect(results).to.equal(resultsPerPage);
7 })
8
9 pm.test("Status code is 200", function () {
10     pm.response.to.have.status(200)
11 })
```

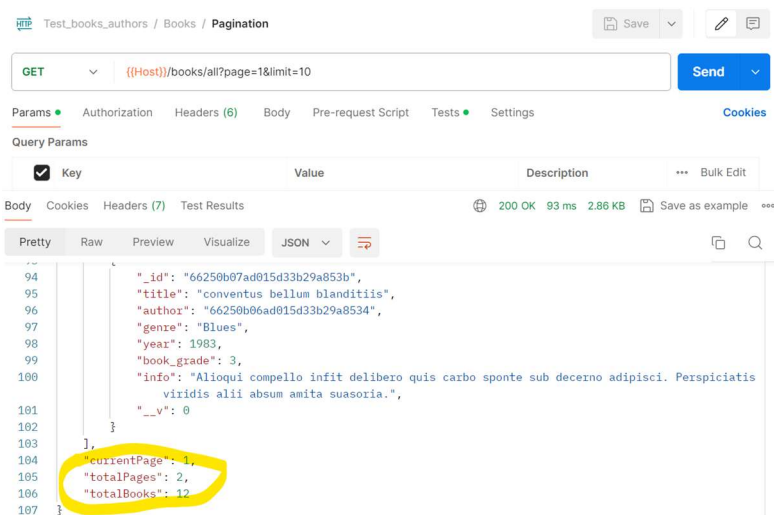
Test and receive tests Snippets Get a test Get a test Get a test Get a test

body Cookies Headers (10) Test Results (2/2) 200 OK 62 ms

All Passed Skipped Failed ↻

PASS Correct number of results per page

PASS Status code is 200



```
server.get("/api/books/all", async (req, res) => {
  try {
    const page = parseInt(req.query.page) || 1 // aktuell sida
    const limit = parseInt(req.query.limit) || 10 // antal elementer på en sida

    const totalBooks = await Book.countDocuments();
    const totalPages = Math.ceil(totalBooks / limit);

    const books = await Book.find()
      .limit(limit);

    res.status(200).json({
      books,
      currentPage: page,
      totalPages,
      totalBooks
    });
  }
});
```

Test 6. Check if the API handles special characters and non-English text correctly in input data and returned responses.

- Manuellt test

Utför en PUT request för att se om det funkar att uppdatera titel, som i detta fall innehåller bokstaven ä. Förväntas att specialtecken och åäö fungerar fint att både mata in och få ut presenterat.

Resultat: Test passerar och i body visas boktitel med bokstaven ä.
Statuskod 200 OK.

BookFinder REST API project / Manuella / **Test6**

GET `{{Host}}/books/{{bookId}}?fields=title`

Overview Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

<input checked="" type="checkbox"/> Key	Value	Description
<input checked="" type="checkbox"/> fields	title	
Key	Value	Description

body Cookies Headers (10) Test Results (1/2) 200 OK 26 ms 37

Pretty Raw Preview Visualize JSON

```
1 {
2   "_id": "664f96a5fa6737378bd23515",
3   "title": "Hekneväven"
4 }
```

- Automatiserat test

GET request för att hämta senast inlagda bok id, med parametern field satt till title. Test för att kontrollera om response field (i detta fall title) innehåller andra tecken än a-z, 0-9, A-Z. Skulle så vara fallet så skrivs det förväntade meddelandet "response field contains other characters" ut.

Resultat: Test passerar och i body visas boktitel med bokstaven ä.
PASS "response field contains other characters"
PASS Status code is 200

BookFinder REST API project / Automatiska / **Test6**

GET `{{Host}}/books/{{bookId}}?fields=title`

Overview Params Authorization Headers (6) Body Pre-request Script Tests

```
1 let response = pm.response.json(),
2   title = _.get(response, 'LineItems.0.title');
3 pm.test("response field contains other characters", () => {
4   pm.expect(/^([0-9a-zA-Z-]+)$/.test(title)).to.be.true;
5 });
6
7 pm.test("Status code is 200", function () {
8   pm.response.to.have.status(200)
9 })
10
11
```

body Cookies Headers (10) Test Results (2/2) 200 OK

All Passed Skipped Failed

PASS response field contains other characters

PASS Status code is 200

Test 7. Test the API's response when sending concurrent requests to ensure that it can handle multiple users and maintain data consistency.

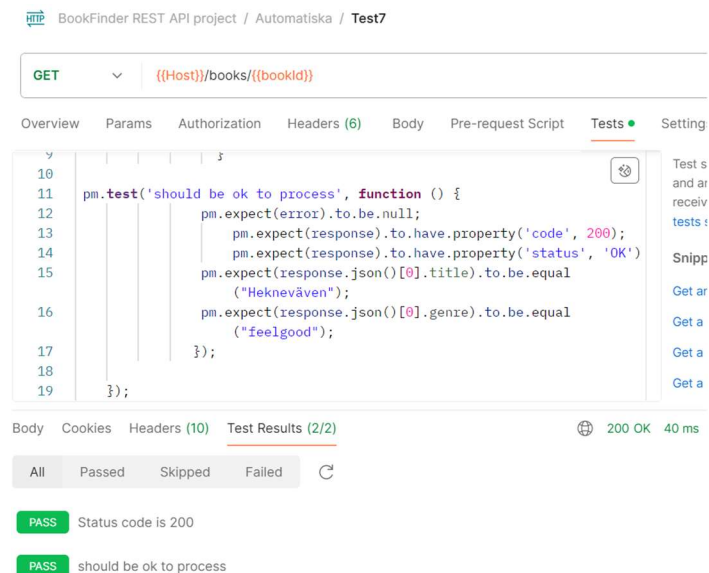
- Manuellt test

Kan inte se hur detta kan testas av realistiskt på ett manuellt sätt, utan det kräver automatiska processer.

- Automatiserat test

GET request för att hämta senast inlagda bok id och i denna request görs en kedjerequest, dvs. en request i en request. Förväntas att {{bookId}} titel och genre matchar den hämtade bokens för att testerna ska passera.

Resultat: PASS Status Code is 200
PASS should be ok to process



Test 8. Test if the API correctly handles different http methods (GET, POST, PUT, DELETE) for each endpoint and returns appropriate status codes and responses for each method.

- Manuellt test
- Hämta alla böcker med en GET-request och få förväntad info samt statuskod 200.

Resultat: Alla inlagda böcker presenteras.
Statuskod 200 OK

Test_books_authors / Books / Get all books

GET `{{Host}}/books`

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

Key	Value	Description
Key	Value	Description

body Cookies Headers (7) Test Results 200 OK 73 ms 1.59 KB

Pretty Raw Preview Visualize JSON

```

68   "_id": "661e2f17529ee02d45d61103",
69   "title": "Hekneväven",
70   "author": [
71     {
72       "_id": "661e28d01a7ed78b060743d5",
73       "fullname": "Lars Mytting",
74       "uv": 0
75     }
76   ],
77   "genre": "historisk roman",
78   "year": 2023,
79   "book_grade": 5,
80   "info": "uppföljaren till den hyllade Systerklockorna. Gudbrandsdalen 1903",

```

- Hämta en bok med en GET-request och id (alternativt `{{bookId}}` för att ta fram senast inlagda bok). Förväntas att bok visas i body och statuskod 200.

Resultat: Bok presenteras.

Statuskod 200 OK

BookFinder REST API project / Manuella / Test8 / One book

GET `{{Host}}/books/{{bookId}}`

Overview Params Authorization Headers (7) Body Pre-request Script Tests Settings

1 2

Test scripts are run and are run as received. Learn more about test scripts

Body Cookies Headers (10) Test Results 200 OK 35 ms 539 B

Pretty Raw Preview Visualize JSON

```

1  {
2    "_id": "6654dfa1d500c8da4ea6375f",
3    "title": "Hekneväven",
4    "author": "664f963efa6737378bd23502",
5    "genre": "feelgood",
6    "year": 2023,
7    "book_grade": 5,
8    "info": "Uppföljaren till den hyllade Systerklockorna. Gudbrandsdalen 1903",
9    "uv": 0
10 }

```

- Skapar en bok med POST-request och förväntar att boken skapas korrekt samt statuskod 201 för Created visas.

Resultat: Bok skapas och presenteras i body

Statuskod 201 Created

BookFinder REST API project / Manuella / Test8 / Create a book

POST {{Host}}/books

Overview Params Authorization Headers (8) Body Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

```
1 {
2   "title": "Hekneväven",
3   "author": "{{authorId}}",
4   "genre": "feelgood",
5   "year": 2023,
6   "book_grade": 5,
7   "info": "Uppföljaren till den hyllade Systerklockorna.Gudbrandsdalen 1903"
8 }
```

body Cookies Headers (10) Test Results 201 Created 55 ms 791 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "newBook": {
3     "title": "Hekneväven",
4     "author": "664f963efa6737378bd23502",
5     "genre": "feelgood",
6     "year": 2023,
7     "book_grade": 5,
8     "info": "Uppföljaren till den hyllade Systerklockorna.Gudbrandsdalen 1903",
9   }
10 }
```

- Uppdaterar en bok med PUT request och bokens id. Förväntas att uppdateringen genomförs och resultatet visas i body och med statuskod 200.

Resultat: Uppdatering utfördes
Statuskod 200 OK

BookFinder REST API project / Manuella / Test8 / Update a book

PUT {{Host}}/books/{{bookId}}

Overview Params Authorization Headers (8) Body Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

```
1 {
2   "title": "Hekneväven",
3   "author": "664f963efa6737378bd23502",
4   "genre": "feelgood",
5   "year": 2023,
6   "book_grade": 5,
7   "info": "Uppföljaren till den hyllade Systerklockorna.Gudbrandsdalen 1903"
8 }
```

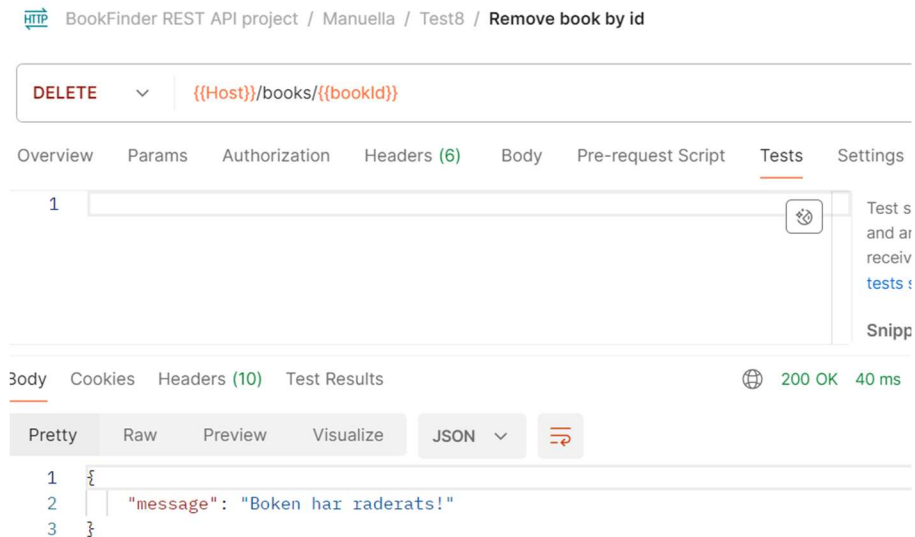
body Cookies Headers (10) Test Results 200 OK 48 ms 539 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "_id": "6654dfa1d500c8da4ea6375f",
3   "title": "Hekneväven",
4   "author": "664f963efa6737378bd23502",
5   "genre": "feelgood",
6   "year": 2023,
7   "book_grade": 5,
8   "info": "Uppföljaren till den hyllade Systerklockorna.Gudbrandsdalen 1903",
9   "__v": 0
10 }
```

- Raderar en bok med DELETE request och id (alternativt {{bookId}} för att radera senast inlagda bok) förväntas att boken raderas och statuskod 200.

Resultat: message: "Boken har raderats!"
Statuskod 200 OK



- Automatiserat test

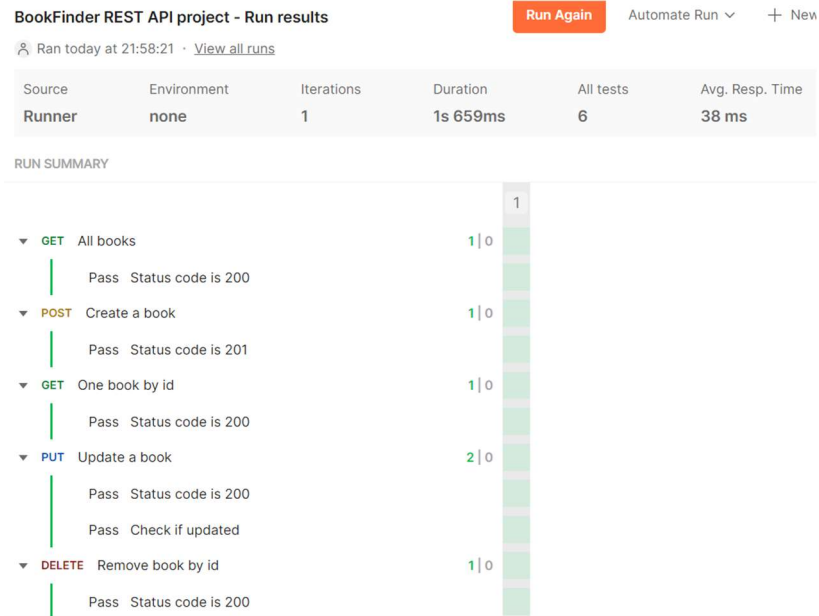
Kör Run Folder på de 5st olika request som finns i folder Automatiska/Test8 i Postman. Förväntas att alla tester passerar och att statuskod 200 och 201(created) visas.

Misslyckas uppdateringen av boktitel beror det på att titeln som uppdateras i body skiljer sig mot bokens titel som finns i Tests fliken. För att få detta test att passera ska titlarna vara lika.

Resultat – Samtliga tester passerar

Pass Status code 200

Pass Status code 201



Test 9. Check if the API correctly handles updates to existing records, ensuring that changes are saved and reflected in subsequent requests.

- Manuellt test

Skapar en bok med POST, uppdaterar sedan bokens titel med PUT och kontrollerar att titeln uppdaterats genom att därefter köra en GET request på den skapade boken. Förväntas då att titeln är uppdaterad och statuskod är 200.

Resultat: Titeln uppdateras.
Statuskod 200 OK

- Automatiserat test

Väljer att uppdatera titeln för en bok med en PUT request på bokens id. Om titeln som uppdateras i body är likvärdig med titeln som finns i testet, förväntas testet passera.

Resultat: PASS Check if updated.
PASS Status code is 200

HTTP BookFinder REST API project / Automatiska / Test9

PUT `{{Host}}/books/{{bookId}}?fields=title`

Overview Params Authorization Headers (8) Body Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

```
1 {
2   "title": "Hekneväven"
3 }
4
```

Body Cookies Headers (10) Test Results (2/2) 200 OK 29 ms 539 B

All Passed Skipped Failed

PASS Status code is 200

PASS Check if updated

HTTP BookFinder REST API project / Automatiska / Test9

PUT `{{Host}}/books/{{bookId}}?fields=title`

Overview Params Authorization Headers (8) Body Pre-request Script Tests Settings

```
1 pm.test("Status code is 200", function () {
2   pm.response.to.have.status(200)
3 })
4
5 pm.test('Check if updated', function () {
6   pm.expect(pm.response.json().title).to.be.equal("Hekneväven");
7
8   });
```

Test scripts are run after the response is received. Learn more about test scripts

Snippets

- Get an environment variable
- Get a global variable
- Get a variable from the response
- Get a collection item

Body Cookies Headers (10) Test Results (2/2) 200 OK 29 ms 539 B

All Passed Skipped Failed

PASS Status code is 200

PASS Check if updated

Test 10. Test the API's performance under heavy load, simulating a large number of users making requests simultaneously.

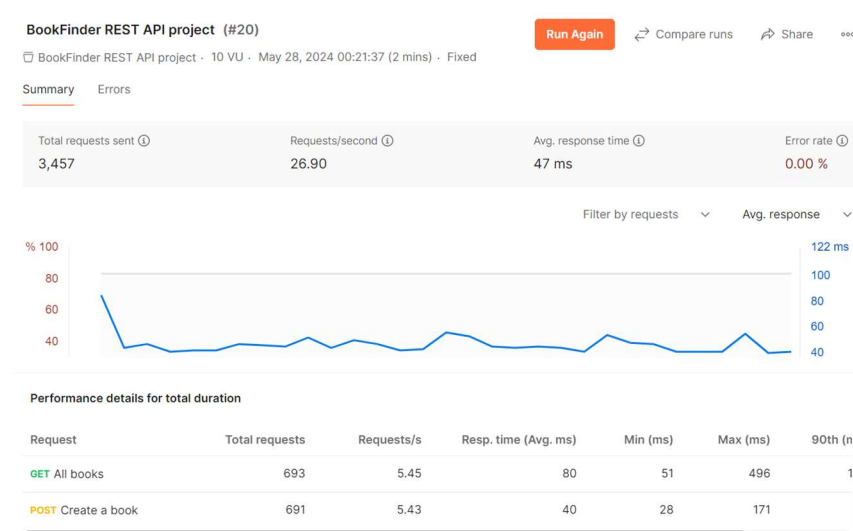
- Manuellt test

Kan inte se hur detta testas av manuellt. Att simulera flera användare är en automatisk process.

- Automatiserat test

Test av API performance testing i Postman genom att köra Run collection av ett antal request (test8). Satte antal användare till 10st och valde att köra testet i 2 minuter. Förväntade att testerna skulle köras med en låg Error rate.

Resultat: Error rate 0.00%

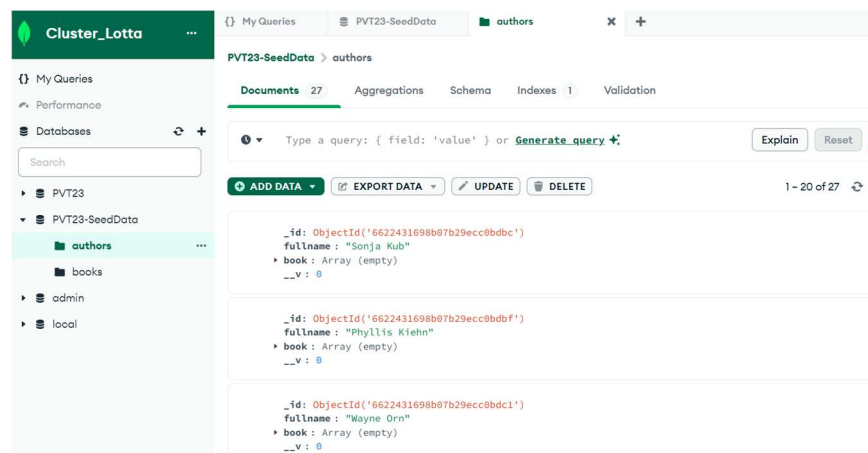


Test 11. Verify that the API can recover gracefully from failures, such as database connection issues without compromising data integrity.

- Manuellt test

Disconnectar Mongo DB och connectar igen och kan då se att all data i authors och books finns kvar under min PVT23-SeedData.

Resultat: Ingen förändring, all data finns kvar i min DB efter avbrottet.

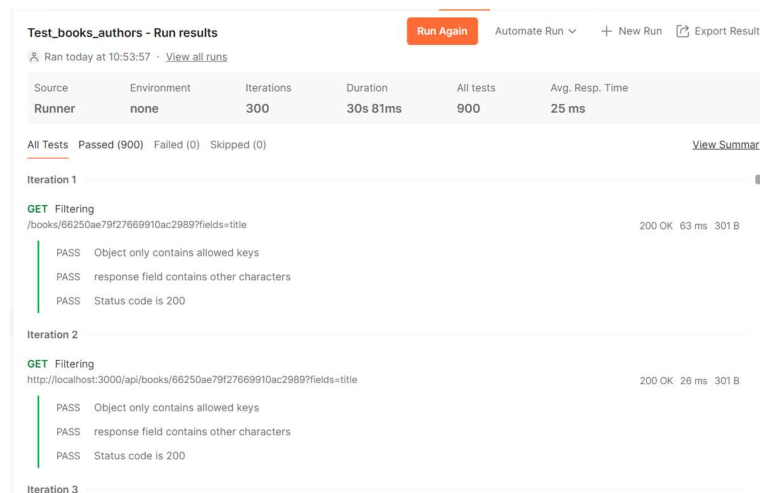


- Automatiserat test

Väljer att köra en GET request med 300 iterationer. Att kopplingen till Mongo DB bryts innan jag kör testet eller mitt i testet, verkar inte spela någon roll. Testet fortsätter att köra och alla 300 iterationer lyckas.

Antar att det beror på att Postman sparar in datan när testet körs vid första tillfället och därför inte är beroende av databaskopplingen i dessa fall.

Resultat: Testet går igenom trots tillfälligt avbrott till Mongo DB.



The screenshot shows the 'Run results' for a test named 'Test_books_authors'. At the top, there are buttons for 'Run Again', 'Automate Run', 'New Run', and 'Export Results'. Below this, a summary table provides an overview of the test run.

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	none	300	30s 81ms	900	25 ms

Below the summary table, it indicates 'All Tests Passed (900) Failed (0) Skipped (0)' with a 'View Summary' link. The results are then broken down by iteration. Iteration 1 and 2 are shown in detail, each featuring a 'GET Filtering' test with three sub-tests: 'Object only contains allowed keys', 'response field contains other characters', and 'Status code is 200'. All sub-tests passed, and the overall status for each iteration was '200 OK'.

Iteration 1

GET Filtering
/books/66250ae79f27669910ac2989?fields=title 200 OK 63 ms 301 B

- PASS Object only contains allowed keys
- PASS response field contains other characters
- PASS Status code is 200

Iteration 2

GET Filtering
http://localhost:3000/api/books/66250ae79f27669910ac2989?fields=title 200 OK 26 ms 301 B

- PASS Object only contains allowed keys
- PASS response field contains other characters
- PASS Status code is 200

Iteration 3

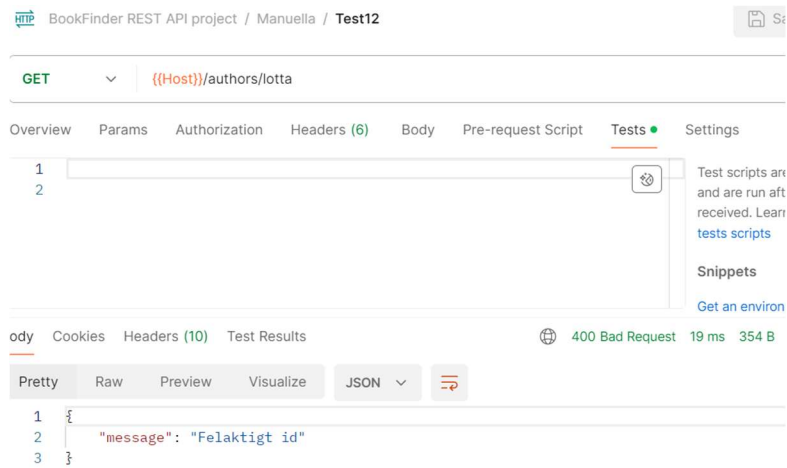
Test 12. Test the API's ability to handle edge cases, such as requests with missing or invalid parameters, and ensure that appropriate error messages are returned.

- Manuellt test

Försöker köra GET request på ett ID som har ett felaktigt format (tex. lotta). Förväntar att få ett felmeddelande om felaktigt id samt felkod 400.

Resultat: "400 Bad Request"

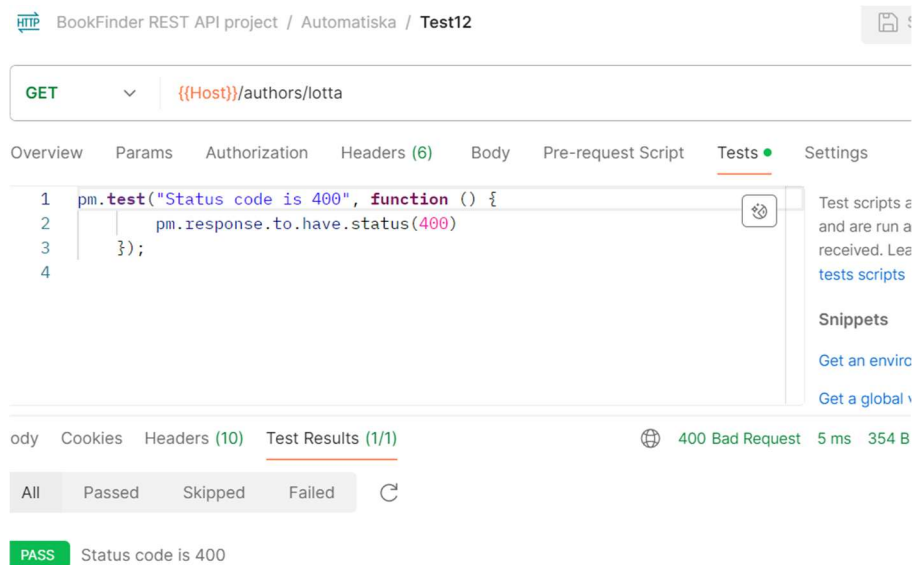
Message: "Felaktigt id"



- Automatiserat test

GET-requesten har ett id av felaktigt format och testet förväntas visa statuskod "400 Bad Request" för att passera.

Resultat: "400 Bad Request"
Message: "Felaktigt id"



authors.js

```
server.get('/api/authors/:id', async (req, res) => {
  if (!mongoose.Types.ObjectId.isValid(req.params.id)) {
    return res.status(400).json({ message: "Felaktigt id" });
  }
})
```

Test 13. Verify that the API correctly implements rate limiting to prevent abuse or excessive use of resources.

- Manuellt test (se nedan på automatiserat test ang. installation av rate-limiting paketet)

Hämtar alla böcker med en GET-request och kontrollerar att det i Postman under Headers visas den förväntade infon för ratelimit.

Resultat: Följande tre "X-RateLimit" finns tillgängliga i Headers för att identifiera att konsumtions limit.

Key		Value
X-Powered-By	①	Express
X-RateLimit-Limit	①	100
X-RateLimit-Remaining	①	50

- Automatiserat test

Installera ***npm i express-rate-limit*** paketet för rate limiting.

I server.js ***import rateLimit from "express-rate-limit"*** och skriver in nedan kod för rateLimit inställningar.

Sätter värdena på rate limit till 100 förfrågningar på en timme. Överskrids värdet får man upp ett meddelande att inga fler förfrågningar kan göras under denna period.

Limit sätts på samtliga request under detta API.

```
import rateLimit from "express-rate-limit"

const server = express()

const port = 3000

server.use(express.json())

const limiter = rateLimit({
  max: 100,
  windowMs: 60 * 60 * 1000,
  message: "To many requests from this IP, please try again in an hour",
});

server.use("/api", limiter);
```

I Postman under Headers finns nu följande tre "X-RateLimit" Headers för att identifiera att konsumtions limit är implementerat.

Jag har valt att testa att nå dessa tre RateLimits från min GET all authors request i Postman och förväntar att test visar att samtliga är korrekt implementerade.

Resultat:

PASS X-RateLimit-Limit

PASS X-RateLimit-Remaining

PASS X-RateLimit-Reset

PASS Status code is 200

The screenshot displays the Postman interface for a GET request to the endpoint `{{Host}}/authors`. The 'Tests' tab is active, showing three assertions:

```
1 pm.test("X-RateLimit-Limit", function () {  
2   pm.response.to.have.header("X-RateLimit-Limit", "100");  
3 });  
4  
5 pm.test("X-RateLimit-Remaining", function () {  
6   pm.response.to.have.header("X-RateLimit-Remaining");  
7 });  
8  
9 pm.test("X-RateLimit-Reset", function () {  
10   pm.response.to.have.header("X-RateLimit-Reset");  
11 });
```

The 'Test Results' tab shows the following results:

Test Name	Status
X-RateLimit-Limit	PASS
X-RateLimit-Remaining	PASS
X-RateLimit-Reset	PASS

The overall status is 200 OK, and the response time is 58 ms.

Länk till min Postman, finns i Readme.md på GitHub och även här:

<https://www.postman.com/lottasv/workspace/my-workspace>