Bettina Kron 2023-04-28

Testrapport för Bookfinder-API

Denna rapport beskriver testningen av API:et BookFinder och innehåller manuell testning och automatisk testning med vardera 13 testfall. För varje testfall beskrivs kortfattat uppgiften, motivering till varför testet görs, förfrågan som skickas, det förväntade resultatet, det faktiska resultatet samt eventuella problem/fel eller kommentarer. Vissa tester ligger tillsammans i en undermapp till BOOKFINDER-collection. Detta därför att jag läste att det är ”best practice” att lägga test som hör ihop i separata mappar. Jag har även gjort mappar för de tester där jag gör en ”Runner” för att slippa riskera att köra flera test i runner-körningen.

**Manuell testning**

**Uppgift 1**

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

**Motivering**

Kontrollera om API:t returnerar en bok respektive alla böcker med tillhörande information.

**Request**

[{{baseURL}}/api/books](http://localhost:3480/api/books)

[{{baseURL}}/api/books/6445109b0d36e24c9a94c0e0](http://localhost:3480/api/books/6445109b0d36e24c9a94c0e0)

**Förväntat resultat**

Statuskod 200 samt titel på boken, författarens namn samt annan detaljerad information om boken.

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel**

Inga

**Uppgift 2**

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

**Motivering**

Kontrollera att API:t returnerar data i JSON-format för en bok.

**Request**{{baseURL}}/api/books/6445109b0d36e24c9a94c0e0

**Förväntat resultat**

Att API:t returnerar information i dataformatet JSON samt meddelande om att formatet är JSON.

**Faktiska resultat**

I Body visas datan i JSON-format. I Headers/Content-Type visas value: /json

**Problem/Fel**

Inga

**Uppgift 3**

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

**Motivering**

Kontrollera att API:t returnerar status kod 400 Bad Request eller 404 Not Found när jag gör en felaktig request.

**Request**

{{baseURL}}/api/boo

**Förväntat resultat**

Inget statusmeddelande ”400 Bad Request”

Statusmeddelande ”404 Not Found”

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel**

Jag har i min kod inte lyckats få try/catch att fungera för ”400 Bad Request”. Jag fick det att fungera men samtidigt var det annat som slutade fungera, t ex limit.

**Uppgift 4**

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

**Motivering**

Testa att filtrera sökningar för att få specifik information i retur, t ex på genre och specifik författare.

**Request**

{{baseURL}}/api/books?genre=Drama

{{baseURL}}/api/books?author=Irvin Langosh

**Förväntat resultat**

Att API:t returnerar ”status 200 OK” samt alla objekt där ”genre” = ”Drama” respektive det objekt vars ”author” är Irvin Langosh.

**Faktiska resultat**

Se förväntat resultat.

**Problem/Fel**

Inga

**Uppgift 5**

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

**Motivering**

Testa för att få ut specifikt antal objekt per sida, i detta testfall fem böcker med tillhörande information.

**Request**

{{baseURL}}/api/books?limit=5&page=1

**Förväntat resultat**

Att API:t returnerar ”status 200 OK” samt de fem första objekten.

**Faktiska resultat**

Se förväntat resultat.

**Problem/Fel**

Inga

**Uppgift 6**

*Test if the API handles special characters and non-English text correctly in input data and returned responses using an automated testing tool.*

**Motivering**

Testa om API:t hanterar ickeengelska tecken samt specialtecken, i detta fall äÅÄÖ respektive !~#%

**Request**

1. {{baseURL}}/api/books?title=Kärlekens ÅÄÖ

2. {{baseURL}}/api/books?title=!~#%

**Förväntat resultat**

Status code 200 OK

**Faktiska resultat**

Status code 200 OK samt en tom array i request 2.

**Problem/Fel/Kommentarer**

Eftersom jag först lade upp böcker med ovan titlar så trodde jag att jag skulle få tillbaka dem i responsen, men jag fick en tom array i fallet med specialtecken. API:t klarar bokstäver som t ex åäö men inte specialtecken som t ex !#%

**Uppgift 7**

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

**Motivering**

Kolla om API:t klarar konkurrerande/simultana requests.

**Request**

{{baseURL}}/api/books

**Förväntat resultat**

Vet ej, någon form av krasch, error code 503 eller liknande

**Faktiska resultat**

Status code 200 OK.

**Problem/Fel**

Svårt att testa detta manuellt. Jag klickade flera gånger på musen. Jag testade också med Run Collection/run manually/iteration:20 respektive 50. Testade kanske för få gånger.

**Uppgift 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.*

**Motivering**

Testa att API:t klarar CRUD, i detta fall: skapa (create) en ny bok, hämta (read) den nya boken genom författarnamnet, uppdatera (put) någon genre och antal sidor samt ta bort (delete) boken.

**Request**

POST {{baseURL}}/api/books

GET {{baseURL}}/api/books?author=Kazuo Ishiguro

PUT {{baseURL}}/api/books/64464c7bfb76d9c72260835c

DELETE {{baseURL}}/api/books/64464c7bfb76d9c72260835c

**Förväntat resultat**

Status code 200 OK samt 201 Created samt information om nya boken. Någon sorts information om att boken är borttagen.

**Faktiska resultat**Se förväntat resultat.  
**Problem/Fel**

Inga

**Uppgift 9**

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

**Motivering**

Kolla om API:t klarar att hantera uppdateringar av en bok.

**Request**

{{baseURL}}/api/books/6445109b0d36e24c9a94c0e4

**Förväntat resultat**

Status code 200 OK samt (uppdaterad) information om nya boken.

**Faktiska resultat**

Se förväntat resultat.

**Problem/Fel**

Inga

**Uppgift 10**

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

**Motivering**

Kolla om/att API:ts prestanda under hög belastning. Detta test gjordes med 100 anrop med hjälp av ”Run collection (Runner)”.

**Request**

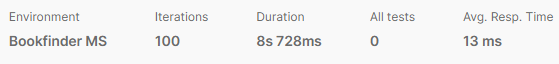
{{baseURL}}/api/books

**Förväntat resultat**

Statuskod 200 OK men med fördröjning av något slag.

**Faktiska resultat**

Statuskod 200 OK



**Problem/Fel/Kommentar**

Inget fel. Viss fördröjning pga att 100 anrop gjordes. *Ett* anrop med samma get-request tog 22 msek.

**Uppgift 11**

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

**Motivering**

Kolla om API:t klarar ett avbrott från servern och databasen. Detta gjorde jag genom att:

1. Skicka en vanlig get-request, 2. Stoppa servern och anslutningen till databasen MongoDB 3. Skicka samma get-request igen 4. Starta servern och återansluta till databasen igen 5. Skicka samma get-request igen.

**Request**

{{baseURL}}/api/books

**Förväntat resultat**

Status code 200 OK respektive status code 500 (Server error)

**Faktiska resultat**

Status code 200 OK respektive ”Error: connect ECONNREFUSED” Could not send request

**Problem/Fel/Kommentar**

Inga fel egentligen. Allt startade upp bra efter avbrottet, och fungerade igen.

**Uppgift 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.*

**Motivering**

Kolla om API:t klarar att hantera när användaren skickar in fel parametrar, i detta fall ”publisher” (som inte finns).

**Request**

{{baseURL}}/api/books?publisher=Bloomsbury

**Förväntat resultat**

Status code 400 Bad request.

**Faktiska resultat**

Se förväntat resultat.

**Problem/Fel**

Inga

**Uppgift 13**

*Verify that the API correctly implements rate limiting or throttling mechanisms to prevent abuse or excessive use of resources.*

**Motivering**

Kontrollera om/att API:et implementerar hastighetsbegränsande mekanismer korrekt om man skickar ett stort antal förfrågningar inom en kort tidsram. Kontrolleras med ”Runner” och 500 itereringar.

**Request**

{{baseURL}}/api/books/6445109b0d36e24c9a94c0e0

**Förväntat resultat**

Code 200 OK men med fördröjning.

**Faktiska resultat**

Code 200 OK. Gjorde 500 itereringar och under den tiden överskreds ingen gräns.



**Problem/Fel/Kommentarer**

Det finns ingen tidsgräns i min kod och i detta fall satte jag en ”mental” gräns på 15 sekunder, så inget konstigt med resultatet. Om fördröjningen är rimlig har jag svårare att bedöma.

**Automatiserad testning**

**Uppgift 1**

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

**Motivering**

Testa att responsen är korrekt, dvs ger statuskod 200 OK

**Request**

{{baseURL}}/api/books?rating=5

**Förväntat resultat**

statuskod 200 OK

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentarer**

Inga.

Testet kontrollerade att ”Status code is 200”.

**Uppgift 2**

*Verify that the API returns the expected data format (e.g., JSON, XML) in the response.*

**Motivering**

Testa att responsen är JSON-format/innehåller JSON-kod

**Request**

{{baseURL}}/api/books?rating=5

**Förväntat resultat**

PASS Status 200 OK

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentarer**

Inga.

Gjorde två test, ett som kollade att responsen innehåller JSON och ett som validerar att ”Content-Type value is application/json”

**Uppgift 3**

*Ensure that the API returns the correct HTTP status code (e.g., 400 Bad Request) for an invalid requests.*

**Motivering**

Testa att responsen visar statuskod 400 Bad Request om felaktig parameter skickas med requesten

**Request**

{{baseURL}}/api/books?hello

**Förväntat resultat**

Statuskod 400 Bad Requst

**Faktiska resultat**

FAIL testet ger tillbaka status 200 OK och hela listan på böcker

**Problem/Fel/Kommentarer**

Testet fallerade eftersom min js-kod inte fångar upp detta fel. Jag har försökt lägga in try/catch med 400 i booksRouter.get() men jag har inte fått det att fungera. Ändrar jag koden får jag det att funka men då kraschar t ex limit och/eller sort.

**Uppgift 4**

*Create an automated test that sends a request with specific filters or search criteria and checks if the API returns the correct data.*

**Motivering**

Testa att ett specifikt filter returnerar korrekt information

**Request**

{{baseURL}}/api/books?genre=Drama

**Förväntat resultat**

PASS Status 200 OK

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentarer**

Inga.

Testet gick ut på att filtreringen på genren ’Drama” ska matcha genren ”Drama” i JSON-responsens body. Testet får ”PASS” om minst 1 matchning sker med det kriteriet.

**Uppgift 5**

*Write an automated test to verify that the API returns paginated results when a large number of records are requested.*

**Motivering**

Testa att API:t returnerar korrekt antal poster per sida

**Request**

{{baseURL}}/api/books?limit=5&page=1

**Förväntat resultat**

PASS Status 200 OK

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentarer**

Inga.

Tre test gjordes och kontrollerade att:

1. ”Status code is 200”
2. Svaret från databasen returnerar exakt 5 objekt/böcker
3. Titeln i det 1-5 (0-4) objekten ska innehålla en viss sträng. Om de gör det, är testet godkänt

**Uppgift 6**

*Test if the API handles special characters and non-English text correctly in input data and returned responses using an automated testing tool.*

**Motivering**

Syftet är att testa om API:et returnerar specialtecken och icke-engelska tecken korrekt

**Request**

{{baseURL}}/api/books?title=Kärlekens ÅÄÖ

{{baseURL}}/api/books?title=!~#%

**Förväntat resultat**

PASS Status 200 OK

PASS Status 200 OK

**Faktiska resultat**

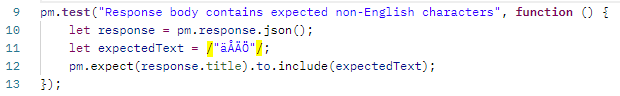
PASS Status 200 OK och att objektet/boken returnerades

PASS Status 200 OK och att endast en tom array returnerades

**Problem/Fel/Kommentarer**

Jag lade först upp en ny bok innehållande icke-engelska bokstäver samt en med specialtecken.

Jag gjorde sedan flera test men fick FAIL, tills jag angav ”/” (se gulmarkerat nedan). Testen fick då status PASS. I fallet med specialtecknen kunde API:et inte returnera något annat än en tom array.



**Uppgift 7**

*Develop an automated test that sends concurrent requests to the API to ensure that it can handle multiple users and maintain data consistency.*

**Motivering**

Syftet med testet är att säkerställa att API:et kan hantera flera förfrågningar samtidigt och upprätthålla ”datakonsistens” (vilket jag tolkar som att säkerställa att resultatet/responsen förblir korrekt och intakt för alla som skickar samma ’request’).

**Request**

{{baseURL}}/api/books?genre=Drama

**Förväntat resultat**

PASS Status 200 OK

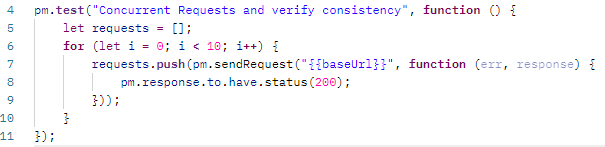
**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentarer**

Inga.

Jag loopade ”requesten” 10 gånger och villkoret var att svaren skulle ha status 200, vilket det blev. Testet gick alltså ut på att skicka 10 samtidiga förfrågningar (concurrent requests) och sedan verifiera att alla förfrågningar får ett svar med HTTP-statuskoden 200.



**Uppgift 8**

*Create an automated test and 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.*

**Motivering**

Syftet är att testa om API:et hanterar de olika HTTP-metoderna GET, POST, PUT och DELETE) korrekt.

**Request**

{{baseURL}}/api/books

{{baseURL}}/api/books/64491580082297dfa83d67fd

**Förväntat resultat**

POST: ny post med status 201 Created PASS

GET/:id: hämta nya posten och status 200 OK PASS

PUT/:id: se den uppdaterade posten samt få status 200 OK PASS

DELETE: se den borttagna posten samt få status 200 OK PASS

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentarer**

Inga.

Jag började med att skapa en ny bok och sedan uppdatera den och ta bort den. Jag testade alla HTTP-metoderna genom att kontrollera status 200 respektive 201. Jag gjorde också tester som validerade att JSON-data fanns i responsen, dvs att den inte innehöll ”null”. Resultaten sparades i tillhörande exempel.

**Uppgift 9**

*Write an automated test to check if the API correctly handles updates to existing records, ensuring that changes are saved and reflected in subsequent requests.*

**Motivering**

Syftet med testet är att säkerställa att ändringar sparas och återspeglas i efterföljande förfrågningar.

**Request**

PUT: {{baseURL}}/api/books/’:id’

GET: {{baseURL}}/api/books/’:id’

**Förväntat resultat**

PASS 200

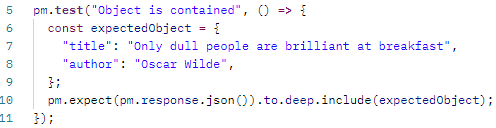
**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentar**

Inga.

Förutom ett test för att säkerställa att responsen har statuskod 200 så gjorde jag ett som kollar att returnerat objekt innehåller uppdateringarna som gjordes. Jag gjorde exakt samma test både i PUT och GET.



**Uppgift 10**

*Design an automated performance test that simulates a large number of users making requests simultaneously to check the API’s performance under heavy load.*

**Motivering**

Testa API:ts prestanda genom att simulera flera förfrågningar/requests samtidigt.

**Request**

{{baseURL}}/api/books?genre=Drama

**Förväntat resultat**PASS 200 OK men med fördröjning

**Faktiska resultat**

Som förväntat resultat





**Problem/Fel/Kommentarer**

Jag testade statuskod 200, vilket jag fick PASS på. Jag gjorde också ett test för att svarstiden skulle vara under 500 msek. Det klarade API:et också. I dessa Runner-test kördes 50, 100 respektive 500 itereringar i ”Runner”.

Så vitt jag kan läsa mig till kan inte Postman Runner hantera *heavy load* eftersom ”Runner” skickar den ena ”requesten” efter den andra och inte samtidigt. Sålunda simulerar man inte *load*. Vad jag förstår kan detta heavy load-test göras genom att ladda hem filen ”parallel-collection-runs.js” från Postman’s Github-repo, exportera Postman-collection in i js-filen och köra testet i Node.js. Jag vet inte om det är meningen att man förväntas göra denna procedur eller kan lösa problemet på något annat sätt. Jag har i alla fall inte gjort det, och förmodligen då inte heller löst uppgiften som man ska.

**Uppgift 11**

*Create an automated test that verifies the API can recover gracefully from failures, such as database connection issues or third-party service outages, without compromising data integrity.*

**Motivering**

Testa om/hur API:t kan återhämta sig från fel, t ex databasanslutningar.

**Request**

{{baseURL}}/api/books

**Förväntat resultat**

Förväntade mig inget resultat eftersom jag inte lyckades lösa denna uppgift.

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentar**

Försökte googla och ta hjälp av chatGPT men jag fick inga tips som gjorde att jag kunde lösa uppgiften, så denna uppgift får jag lämna olöst.

**Uppgift 12**

*Develop an automated test to handle edge cases, such as requests with missing or invalid parameters, and ensure that appropriate error messages are returned.*

**Motivering**

Testa hur API:t hanterar förfrågningar som saknar parametrar eller är ogiltiga.

**Request**

{{baseURL}}/api/books/{{invalidInput}}

**Förväntat resultat**

PASS Status code 500 Indicates server error

FAIL AssertionError: expected response to have status code 404 but got 500

**Faktiska resultat**

Se förväntat resultat

**Problem/Fel/Kommentar**

Gjorde två tester för att fånga upp error code 400 Invalid Request och error code 500 Server Error. Att det förstnämnda inte fungerar visste jag efter som jag inte har fått min try/catch att fånga det felet i min booksRouter.get() i books.js. Oavsett går koden in i else-satsen med

res.status(500).json({ message: err.message });

**Uppgift 13**

*Write an automated test to verify that the API correctly implements any rate limiting or throttling mechanisms to prevent abuse or excessive use of resources.*

**Motivering**

Att skriva ett automatiserat test för att verifiera att API:et implementerar alla hastighetsbegränsande eller ”strypningsmekanismer” korrekt för att förhindra missbruk eller överdriven användning av resurser.

**Request**

{{baseURL}}/api/books

**Förväntat resultat**

PASS Status code 200 or code 429

**Faktiska resultat**

Lyckades inte lösa uppgiften

**Problem/Fel/Kommentarer**

Jag antar att jag i koden måste lägga in en limit i min js-fil på en användares sökningar per minut (som exempel). Överskrids denna limit ska statuskod 429 (”indicates the client application has surpassed its rate limit”) aktiveras. Testet ska förstås fånga upp detta. Tyvärr en för svår uppgift för mig att lösa.