

# API testing: strategy

**Test Suite:** Bigbank Loan Calculator

**QA:** Liis Nelis

**Date:** 13.06.2025

My testing focused on validating the logic and error handling of /calculate endpoint through realistic and edge-case scenarios.

I created a Postman collection covering:

- **Valid input:** standard loan amounts and period to verify expected outputs for monthly payment and APRC.
- **Missing or unexpected fields:** adding or removing fields from the request body to verify how the system responds and what kind of errors are shown.
- **Invalid data types:** changing expected data types and validating correct system behavior.
- **Boundary cases:** minimum and maximum supported loan amounts and durations to ensure the system behaves correctly at its limits.
- **Invalid inputs:** Negative values, missing fields, or malformed data to verify proper validation and error handling.

These scenarios were chosen to ensure the functional correctness of the /calculate endpoint. The test scripts also include automated assertions to validate both the HTTP status codes and the structure/content of the response.


If I had more time to test I would:

- check for data integrity if I knew more about the formula that calculates the APRC values and whether it is even reasonable to add such validations to the collection (depending on how often the “fixed” parameters change)
- test for a combination of negative/invalid inputs to see if the response is still valid
- include performance testing – e.g. measure response time with a large number of virtual users and requests (Postman has a handy feature for this).

## Test Report for API-tests

Test Case ID	Method	Description	Expected result	Actual result	Status
API01	POST	Input contains valid values	200	200	✓
API02	POST	Amount value is missing	400	400	✓
API03	POST	Maturity value is missing	400	400	✓
API04	POST	Unexpected field in body	400	400	✓
API05	POST	Invalid input type	400	400	✓
API06	POST	Amount value exceeds upper limit	400	200	✗
API07	POST	Maturity value is below lower limit	400	500	✗
API08	POST	Loan amount is at lower boundary	200	200	✓
API09	POST	Loan amount is at upper boundary	200	200	✓
API10	POST	Maturity value lower boundary	200	200	✓
API11	POST	Maturity value upper boundary	200	200	✓

Elaboration of failed test results

	API06	API accepts value above UI max; backend may lack boundary validation
		<div><p>While UI restricts the loan amount to the range of 500 to 30000 EUR (any attempt to enter out-of-range values is automatically corrected to the nearest valid limit), the backend /calculate endpoint does not follow these boundaries. Amount values below 500 or over 30000 still return a valid response with the complete calculation. This could lead to inconsistencies when someone is interacting with the API directly.</p></div> <div><div><div>POST</div><div>https://taotlus.bigbank.ee/api/v1/loan/calculate</div></div><div><div>Params</div><div>Authorization</div><div>Headers (10)</div><div>Body</div><div>Pre-request Script</div><div>Tests</div><div>Settings</div></div><div><div><div><div></div><div>none</div></div><div><div></div><div>form-data</div></div><div><div></div><div>x-www-form-urlencoded</div></div><div><div></div><div>raw</div></div><div><div></div><div>binary</div></div><div><div></div><div>GraphQL</div></div><div><div></div><div>JSON</div></div></div></div><div><pre>1 { 2   "administrationFee": 3.99, 3   "amount": 150000, 4   "conclusionFee": 100, 5   "currency": "EUR", 6   "interestRate": 15.1, 7   "maturity": 60, 8   "monthlyPaymentDay": 15, 9   "productType": "SMALL_LOAN_EE01" 10 }</pre></div><div><div>body</div><div>Cookies (3)</div><div>Headers (23)</div><div>Test Results (0/1)</div></div><div><div><div></div><div>Status: 200 OK</div></div></div><div><div><div>Pretty</div><div>Raw</div><div>Preview</div><div>Visualize</div><div>JSON</div></div><div><div></div></div></div><div><pre>1 { 2   "totalRepayableAmount": 216089.9, 3   "monthlyPayment": 3601.5, 4   "apr": 16.52 5 }</pre></div></div>



API07

Server responds with 500 error instead of rejecting invalid input with a 4xx error. Backend may have some unhandled errors.

When testing with an out-of-bounds maturity value, instead of the expected 400 (invalid request), the server returned 500 with the message "Well, we did not see this one coming". Without complete documentation and business requirements to suggest otherwise, this points to some unhandled errors.

POST

https://taotlus.bigbank.ee/api/v1/loan/calculate

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1 {

2 "administrationFee": 3.99,

3 "amount": 5000,

4 "conclusionFee": 100,

5 "currency": "EUR",

6 "interestRate": 15.1,

7 "maturity": 0,

8 "monthlyPaymentDay": 15,

9 "productType": "SMALL\_LOAN\_EE01"

10 }

Body

Cookies (3)

Headers (21)

Test Results (0/1)

Status: 500 Internal Server Error

Pretty

Raw

Preview

Visualize

JSON

1 {

2 "error": {

3 "code": 500,

4 "message": "Well, we did not see this one coming"

5 }

6 }