**Positive Test Cases:**

* Test with a valid loan amount, loan period, ensuring that the total repayable amount, monthly payment and APRC match the expected values based on manual calculations.

**Boundary Test Cases:**

* Test with the loan amount and loan period at their lower boundaries (minimum values allowed). Check if the calculator handles these edge cases without errors.
* Test with the loan amount and loan period at their upper boundaries (maximum values allowed). Verify if the calculator provides correct results without exceeding any limits.

**Negative Test Cases:**

* Test with invalid loan amounts (e.g., negative value or zero), and verify that the calculator returns appropriate error messages.
* Test with invalid loan period(e.g., negative value or zero), and verify that the calculator returns appropriate error messages.

**Load Testing:**

* Test the endpoint with a large number of concurrent requests to check its performance under load and verify if it responds within acceptable time limits.

**Integration Tests:**

* Test the integration between the frontend and the backend to ensure that the data is passed correctly to the "calculate" endpoint and that the results are displayed accurately on the frontend.

**Reasoning Behind the Chosen Scenarios:**

* Positive test cases ensure that the basic functionality works as expected.
* Boundary tests help identify potential issues near the edges of valid input ranges.
* Negative tests help ensure that the calculator handles invalid inputs.
* Load testing helps identify any performance bottlenecks and ensure the system can handle peak loads without crashing.
* Integration testing verifies that the entire system works seamlessly, passing data correctly between the frontend and backend.