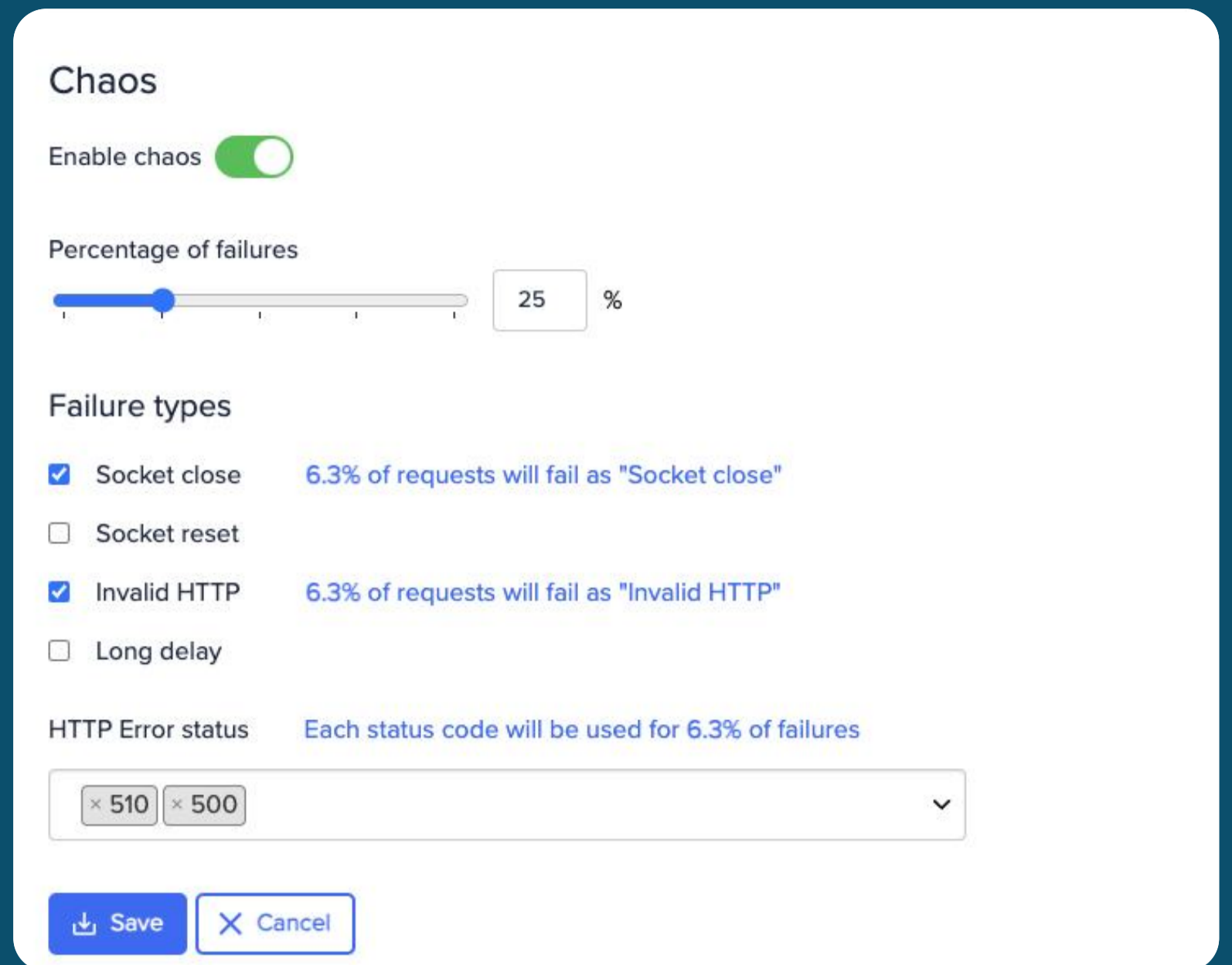**Chaos Engineering**

Add Chaos Engineering to your API Testing and Development

Test product reliability in unexpected fault scenarios by incorporating common (and uncommon) errors in your mock APIs

Chaos

Enable chaos

Percentage of failures

25 %

Failure types

☑ Socket close   6.3% of requests will fail as "Socket close"

☐ Socket reset

☑ Invalid HTTP   6.3% of requests will fail as "Invalid HTTP"

☐ Long delay

HTTP Error status   Each status code will be used for 6.3% of failures
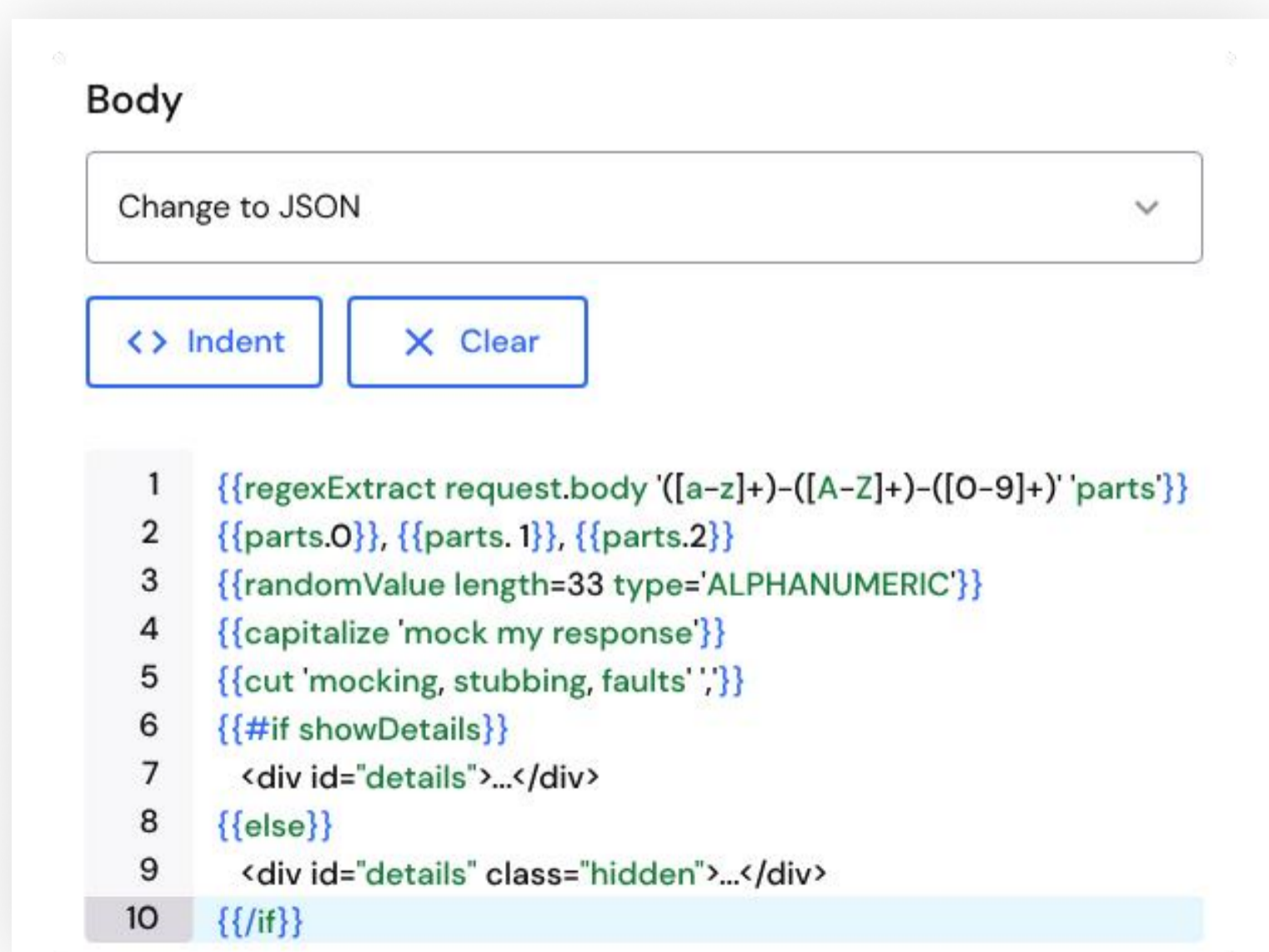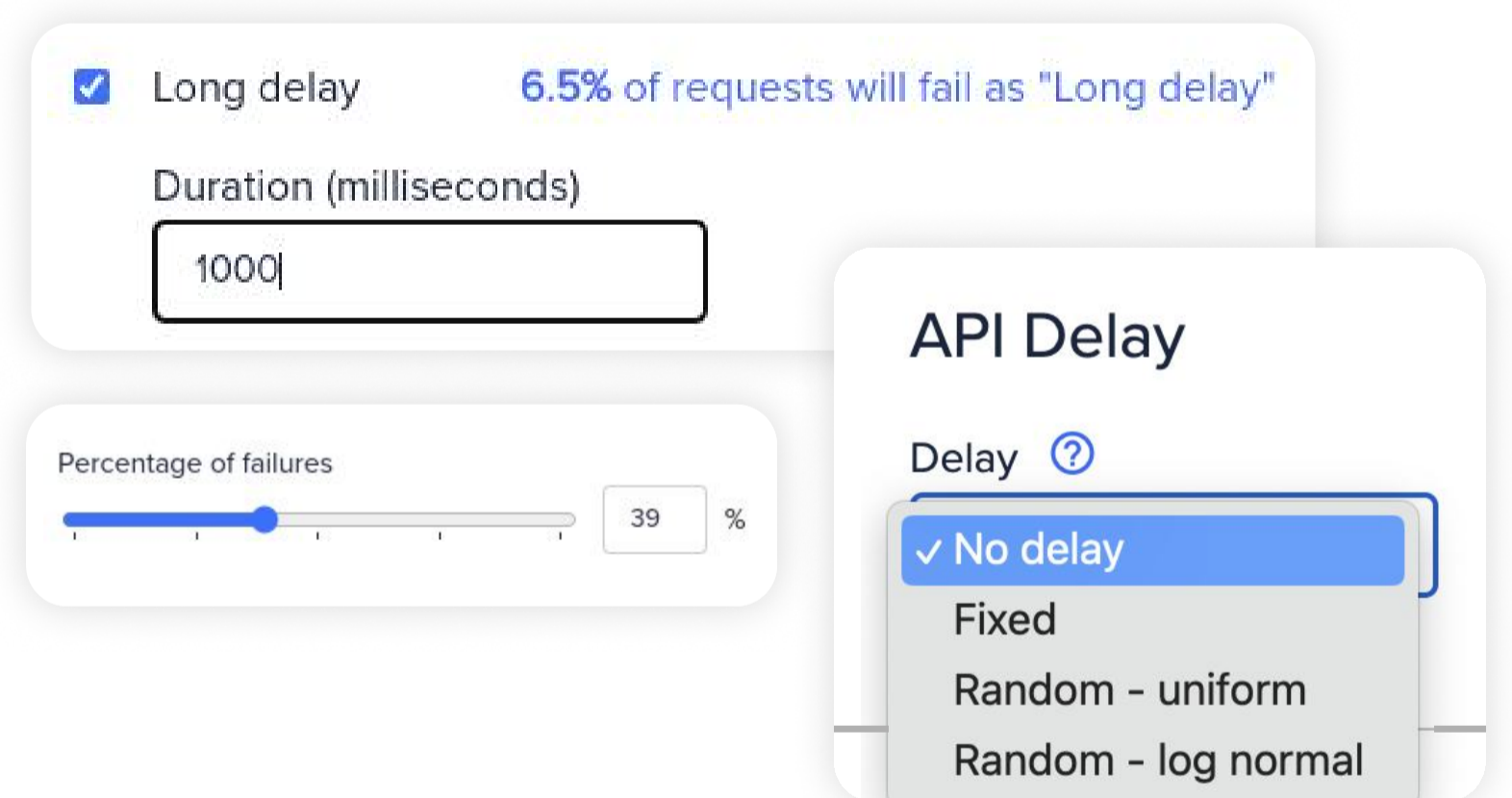
× 510   × 500

Save   Cancel

The Challenge

In the real world, APIs can and will fail. Timeouts, network failures, or unexpected error statuses are an everyday occurrence – and your testing and development should reflect this.

The Solution

WireMock Cloud's Chaos features allow your API mocks to simulate a wide range of unexpected failure scenarios, so you can learn how to build more resilience into your app.

Inject complex failure states into your mocked APIs

- Introduce a wide range of random chaos elements and error statuses
- Control the percentage and distribution of failures to mimic real-world conditions
- See how your app handles fail states that happen in the network layer: invalid HTTP, socket reset, socket close

☑ Long delay   **6.5%** of requests will fail as "Long delay"

Duration (milliseconds)

1000

Percentage of failures

39 %

API Delay

Delay ⓘ

✓ No delay

Fixed

Random - uniform

Random - log normal

Body

Change to JSON

< > Indent   × Clear

```
1   {{regexExtract request.body '([a-z]+)-([A-Z]+)-([0-9]+)' 'parts'}}
2   {{parts.0}}, {{parts. 1}}, {{parts.2}}
3   {{randomValue length=33 type='ALPHANUMERIC'}}
4   {{capitalize 'mock my response'}}
5   {{cut 'mocking, stubbing, faults' ','}}
6   {{#if showDetails}}
7      <div id="details">...</div>
8   {{else}}
9      <div id="details" class="hidden">...</div>
10  {{/if}}
```

Run more realistic integration tests

- See how your entire environment is impacted by 3rd party API errors or a microservice outage
- Avoid disasters in production by identifying show-stopping fault scenarios testing
- Identify and mitigate upstream outages in situations that are likely to occur