EFFICIENT END-TO-END TESTING WITH PUPPETEER AND GRAPHQL SNAPSHOTS...

\$ whoami



=> Name: Sebastian Scholl

Title: Product Manager @8base

Contact: ~/twitter/medium/linkedin/@sebscholl

SO MANY DEVELOPMENT BOTTLENECKS...

"an inefficiency where the flow of the project and/or work stops or slows at a particular point"

Code Review + QA + Teams + Fuzzy Spec + Regulation +
...conferences + Security + etc.

SPEEDY AND CHEEP E2E CLIENT TESTS...

PROBLEMS

- E2E tests slowest type of tests;
- E2E tests spend money (we use AWS Lambda);
- Sometimes E2E tests could be unstable by server-side reasons;

GOAL

- E2E tests for all basic user scenarios;
- E2E tests run on all merge requests, reducing bug discovery time;
- Reduce the cost of running E2E tests, if possible (API Calls);
- Reduce the occurrence of failed runs for **non**-client reasons.



JEST + PUPPETEER SOLUTION...

```
it('As a user, I can open page an do something.', async () => {
    // Puppeteer listens to all requests and overrides responses
    const page = await initPage(__BROWSER_CONTEXT__);
    // Use Jest plugin to add handlers to Puppeteer page object to use/record mocks
    await expect(page).toMatchPuppeteerRequestMocks();
    // Open the page in headless browser
    await page.open('http://localhost:3000');
    // ...do something that's a test/important!

    // Close page and plugin records collected
    await page.close();
});
```



WHAT'S HAPPENING...

- 1) Page event listeners added the request,
 response and close events
- 2) Requests event counts requests to be saved on close.
- 3) Response collects pairs of requests and corresponding responses.
- 4) **Close** signals all request/response are collected and can get saved

```
"CompanyName": [
   "request": {
     "operationName": "CompanyName",
     "variables": {},
     "query": "query CompanyName { companyName }"
   "response": {
     "data": {
        "companyName": "8base"
```



LETS RUN IT...

...about to make code happen /

SOME LIGHT BENCHMARKING...

8base runs 200 E2E tests for user scenarios in three modes:

- Use recorded requests mocks merge requests and local checking;
- Use recording requests mocks new tests and after server-side updates;
- On a real server run locally for tests in release branches;

Time for modes:

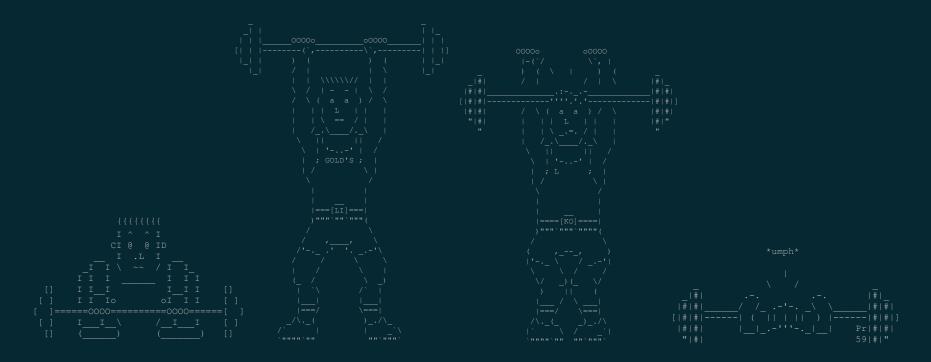
- Recorded requests mocks developer laptop: ~390s, CI: ~230s;
- Recording requests mocks developer laptop: ~954s, CI: ~600s;
- Real server developer laptop: ~954s, CI: ~600s.
- 2.5-times speed boost with no paralyzation/added checks, the ability to run tests without a server, and less money spent on AWS Lambda.



REVISITING OUR GOALS LIST...

- ✓ E2E tests run on all merge requests, reducing bug discovery time;
- \checkmark Reduce the cost of running E2E tests, if possible (API Calls);
- ✓ Reduce the occurrence of failed runs for non-client reasons.
- ...test for all basic user scenarios

TESTING FOR ALL BASIC SCENARIOS...





EXIT STATUS 0...

```
Repo: Jest/Puppeteer Plugin: https://github.com/8base/jest-mock-puppeteer-requests
...thank you /
```