From testing to observing

Synthetic monitoring unpacked

elastic

Jessica Garson

@JessicaGarson

@JessicaGarson@macaw.social

Senior Developer Advocate

elastic

https://github.com/JessicaGarson/Synthetic-Monitoring-Unpacked

elastic

Agenda

- Overview of testing and synthetic monitoring
- Introduce the framework for testing
- Synthetic monitoring in Elastic
- How to write and run tests
- Closing & Key Takeaways

elastic

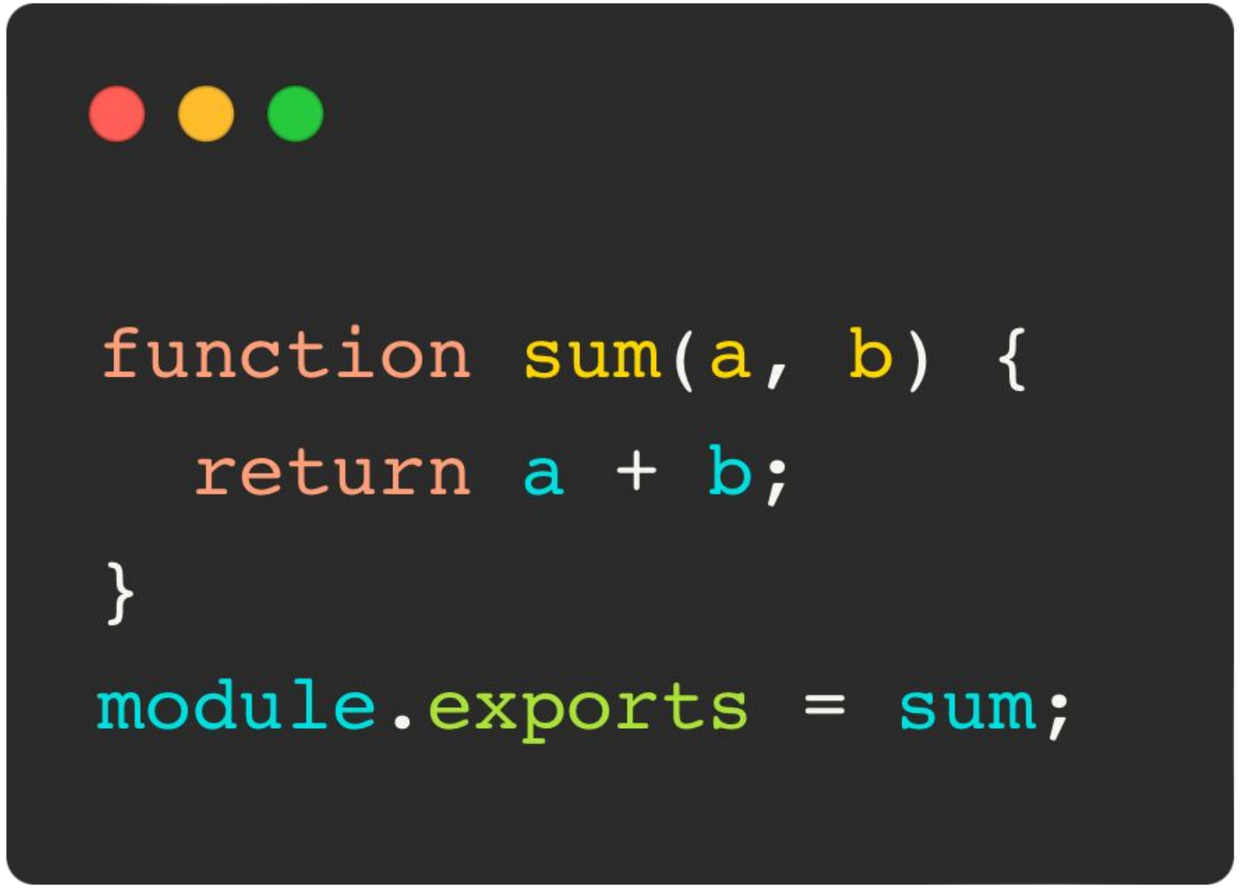
A simple example

elastic

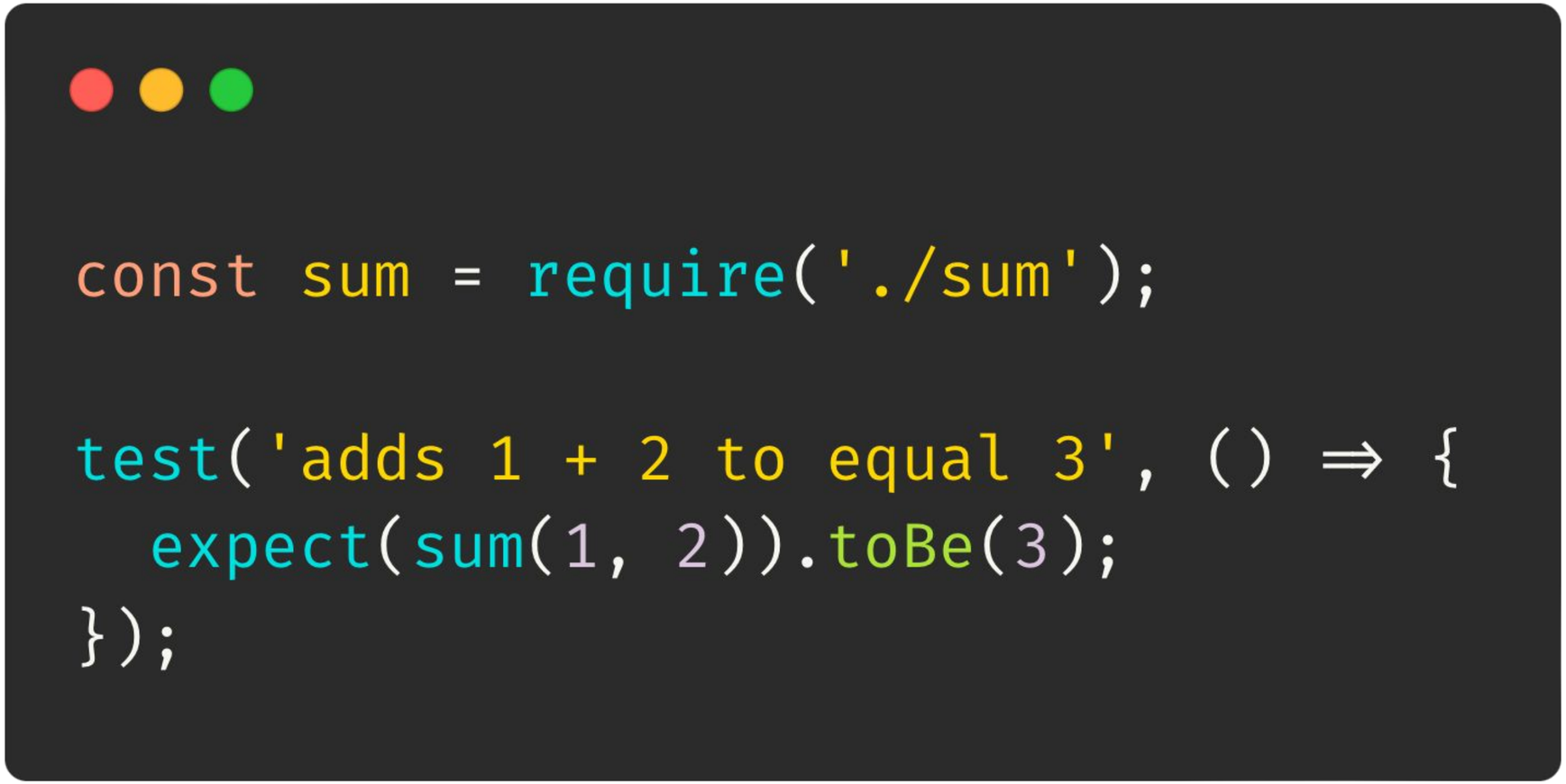
What is testing?

Testing in software development allows us to examine individual functions or methods and ensure they behave as expected under various conditions.

Jest is a popular testing framework for JavaScript



elastic



```
const sum = require('./sum');

test('adds 1 + 2 to equal 3', () => {
  expect(sum(1, 2)).toBe(3);
});
```

What is end-to-end (E2E) testing?

E2E testing checks an entire software application from beginning to end, mimicking fundamental user interactions and data, verifying that a software system performs as intended, covering the complete flow of the application.

```
import { test, expect } from '@playwright/test';

test('has title', async ({ page }) => {
  await page.goto('https://playwright.dev/');

  // Expect a title "to contain" a substring.
  await expect(page).toHaveTitle(/Playwright/);
});

test('get started link', async ({ page }) => {
  await page.goto('https://playwright.dev/');

  // Click the get started link.
  await page.getByRole('link', { name: 'Get started' }).click();

  // Expects page to have a heading with the name of Installation.
  await expect(page.getByRole('heading', { name: 'Installation' })).toBeVisible();
});
```

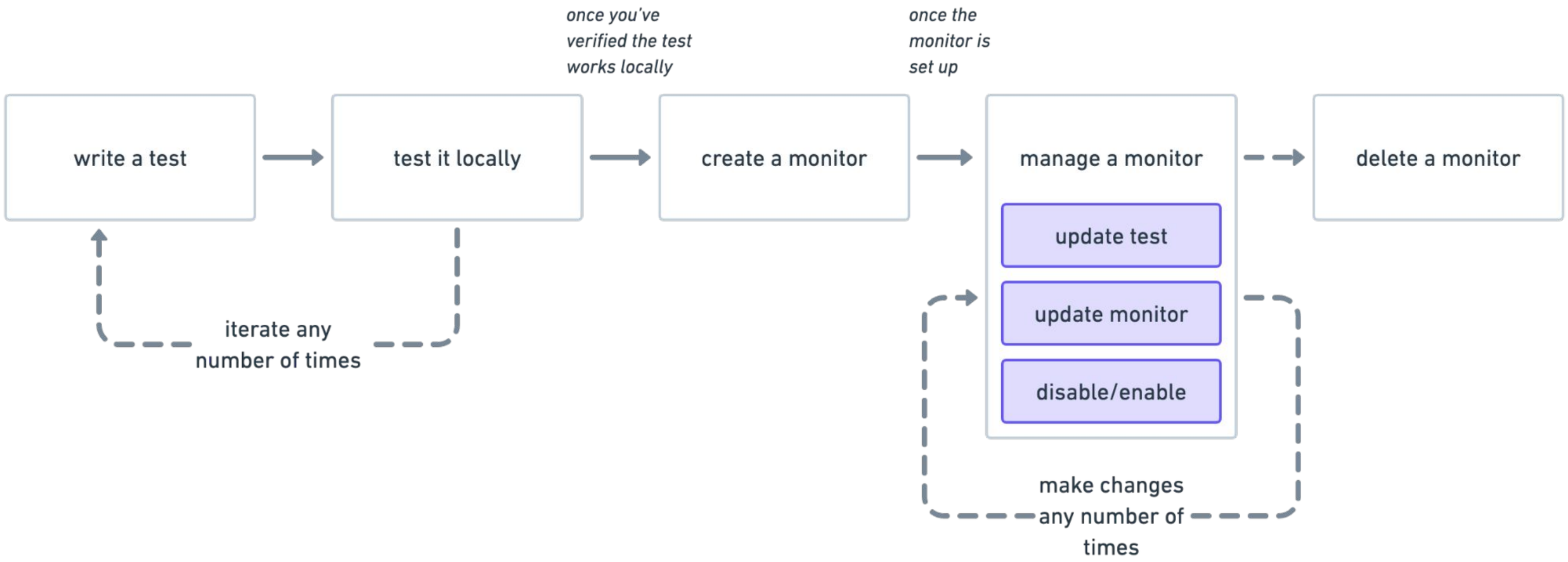
elastic

What is Synthetic Monitoring?

Synthetic monitoring allows you to track user pathways using global testing infrastructure, emulating the full user path to gauge the influence of web applications.

How does synthetic monitoring relate to testing?

Both testing and synthetic monitoring emulate the user path. This tooling can be used to build tests that can also provide production monitoring in web applications.


once you've verified the test works locally
once the monitor is set up
write a test
test it locally
create a monitor
manage a monitor
delete a monitor
update test
update monitor
disable/enable
iterate any number of times
make changes any number of times

elastic

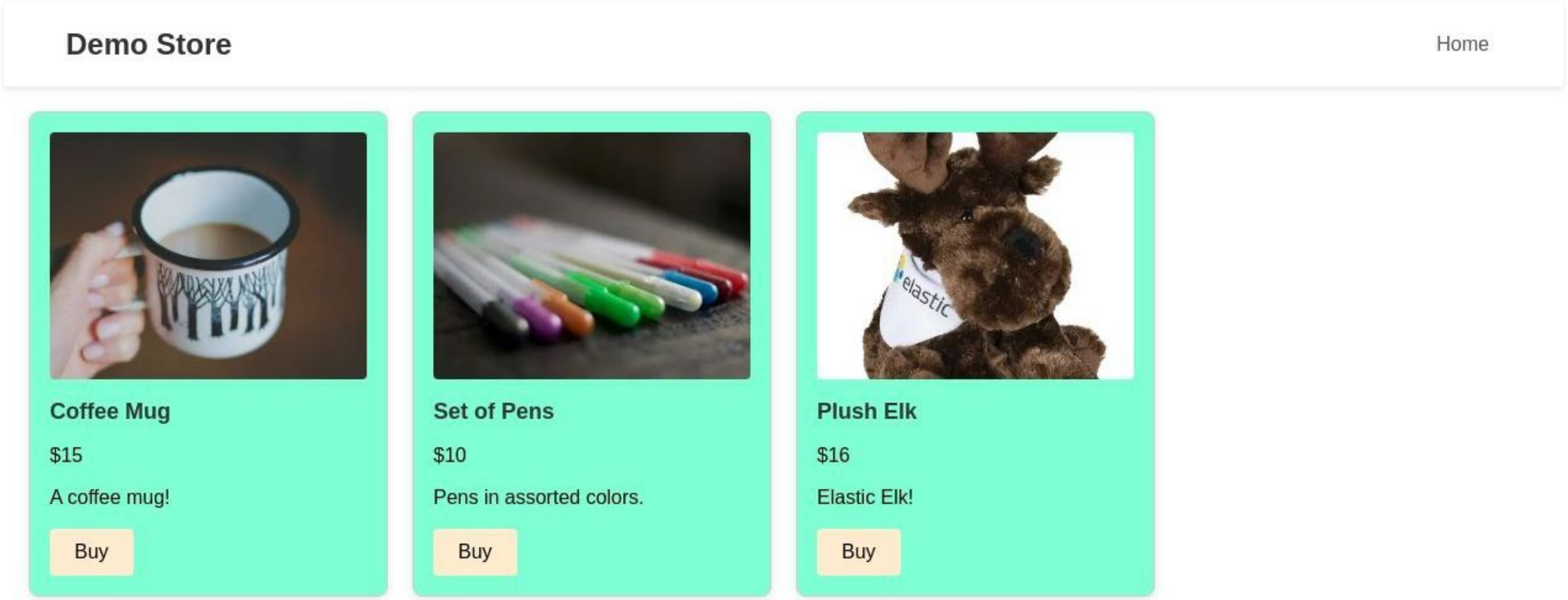
Demos

elastic

Next.js

elastic

https://demo-store-ivory.vercel.app/



elastic

https://github.com/JessicaGarson/demo-store

elastic

Monitors

Create Monitor | Refresh

Overview | Management

Search by name, URL, host, tag, project or location

Up | Down | Disabled | Type 2 | Location 1 | Tags 0 | Frequency 2 | Project 1

Current status

3 Up | 0 Down | 0 Disabled

Last 6 hours

0 Errors

Last 12 hours

0 Alerts

Showing **3 Monitors**

Sort by Status | **Group by** None

Lightweight demo store
North America - US East
September 24, 2024 3:05 PM
Duration
47 ms

Recorded journey Demo
North America - US East
Duration
1 s

Simplified Test for Demo Store
North America - US East
Duration
999 ms

elastic

"Simplified Test for Demo Store" (North America - US East) is down - Elastic Synthetics

No Reply - Elastic Alerts <noreply@alerts.elastic.co>
to me

Wed, Sep 25, 12:35 PM (8 days ago)

"Simplified Test for Demo Store" is down from North America - US East. - Elastic Synthetics

Details:

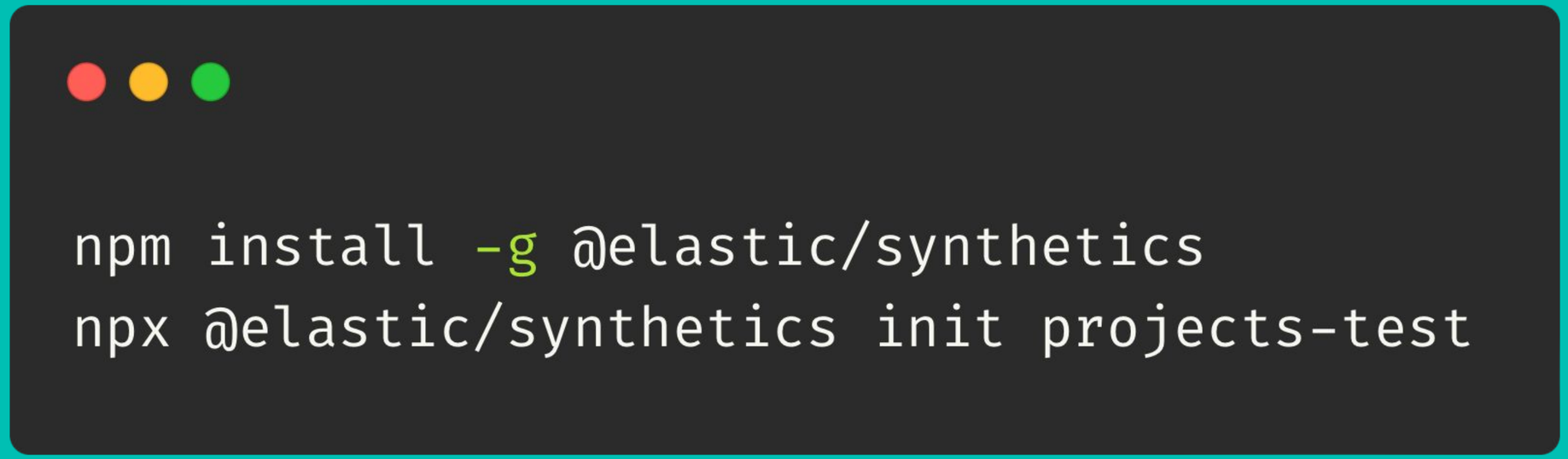
- Monitor name: Simplified Test for Demo Store
- URL: https://demo-store-ivory.vercel.app/
- Monitor type: browser
- Checked at: Sep 25, 2024 @ 16:34:48.900
- From: North America - US East
- Error received: error executing step: Timed out 5000ms waiting for expect(locator).toBeVisible()
- Link: https://getting-started.kb.us-east4.gcp.elastic-cloud.com:9243/app/synthetics/monitor/bf435caa-83eb-45b1-a876-a6c20b8035aa/errors/us-east4-a-1922a090bc4-0?locationId=us-east4-a

---

This message was sent by Elastic. View rule in Kibana.

elastic

How to get started



elastic

```
changed 144 packages in 3s

26 packages are looking for funding
  run `npm fund` for details
> Initializing Synthetics project in 'live-demo'
✔ Enter Elastic Kibana URL or Cloud ID · your_cloud_id==
✔ What is your API key · ************************************************************
✔ Select the locations where you want to run monitors · us_east
✔ Set default schedule in minutes for all monitors · 10
✔ Choose project id to logically group monitors · live-demo
✔ Choose the target Kibana space · default
```

elastic

```
> Setting up project using NPM ...
Wrote to /Users/jessgarson/rehearsal-synth-mon/live-demo/package.json:

{
  "name": "live-demo",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}

> Installing @elastic/synthetics library ...
```

```
> Writing live-demo/synthetics.config.ts.
> Writing live-demo/journeys/example.journey.ts.
> Writing live-demo/lightweight/heartbeat.yml.
> Writing live-demo/journeys/advanced-example-helpers.ts.
> Writing live-demo/journeys/advanced-example.journey.ts.
> Writing live-demo/.github/workflows/run-synthetics.yml.
> Writing live-demo/README.md.
> Writing live-demo/package.json.

All set, you can run below commands inside: /Users/jessgarson/rehearsal-synth-mon/live-demo:

  Run synthetic tests: npm run test

  Push monitors to Kibana: SYNTHETICS_API_KEY=<value> npm run push

  Configure API Key via `SYNTHETICS_API_KEY` env variable or --auth CLI flag.

Visit https://www.elastic.co/guide/en/observability/current/synthetic-run-tests.html to learn more.
```

elastic

Key commands to keep in mind

Run synthetic tests in the journeys folder :

```
npm run test
```

Push monitors to Kibana:

```
SYNTHETICS_API_KEY=<value> npm run push
```

elastic

Heartbeat

```
heartbeat.monitors:
- type: http
  name: Lightweight demo store
  id: demo-store-lightweight
  enabled: true
  urls: "https://demo-store-ivory.vercel.app"
  schedule: '@every 3m'
  timeout: 16s
  alert.status.enabled: true
```

elastic

Elastic Synthetics 🤝 Playwright

elastic

```
import { journey, step, monitor, expect } from '@elastic/synthetics';

journey('Simplified Test for Demo Store', ({ page, params }) => {
  monitor.use({
    id: 'demo-store-check',
    schedule: 10,
  });

  step('Launch application', async () => {
    await page.goto(params.url);
  });

  step('Check for loading state', async () => {
    const loadingText = page.getByText('Loading...');
    if (await loadingText.isVisible({ timeout: 5000 })) {
      await loadingText.waitFor({ state: 'hidden', timeout: 60000 });
    }
  });
```

elastic

```
  step('Validate page content', async () => {
    const header = page.getByRole('heading', { level: 1 });
    await expect(header).toHaveText('Demo Store');

    const firstProductCard = page.getByTestId('product-card').first();
    await expect(firstProductCard).toBeVisible();

    const buyButton = firstProductCard.getByTestId('buy-button');
    await expect(buyButton).toBeVisible();

    await page.screenshot({ path: 'page-content.png' });
  });
```

```
step('Interact with the Buy button', async () => {
  page.once('dialog', async dialog => {
    const message = dialog.message();
    console.log('Alert message:', message);
    expect(message).toBe('Added to cart!');
    await dialog.accept();
  });

  const firstProductCard = page.getByTestId('product-card').first();
  const buyButton = firstProductCard.getByTestId('buy-button');
  await buyButton.click();

  await page.screenshot({ path: 'after-click.png' });
});
```

```
  step('Monitor page performance', async () => {
    const performance = await page.evaluate(() => {
      return JSON.stringify(window.performance);
    });
    console.log(performance);
  });
});
```

Let’s run our tests! 🧪

elastic

‹ Monitors

Simplified Test for Demo Store

Overview | History | Errors

Location
North America - US East

Status
Succeeded

Last run
9 seconds ago

Run test manually

Edit monitor | Refresh

Monitor details

Enabled (all locations)	
URL	https://demo-store-ivory.vercel.app/
Last run	Oct 1, 2024 @ 2:26:14 PM
Last modified	Sep 25, 2024 @ 1:19:31 PM
Project ID	projects-test
Monitor ID	demo-store-check-projects-test-default
Monitor type	Journey
Frequency	Every 10 minutes
Locations	North America - US East
Tags	--

Summary To date

100.0% Availability

1.0 s Median duration

1 Errors

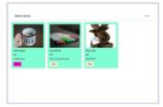
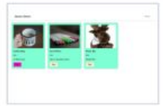
Simplified Test for Demo Store-Test results

North America - US East COMPLETED Took 1 s

View test result details

5 steps completed

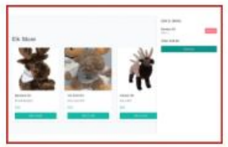
Steps

#	Screenshot	Step name	Result	Duration
1		Launch application	Complete	688 ms
2		Check for loading state	Complete	102 ms
3		Validate page content	Complete	184 ms
4		Interact with the Buy button	Complete	238 ms
5		Monitor page performance	Complete	47 ms

elastic

What happens when a test fails

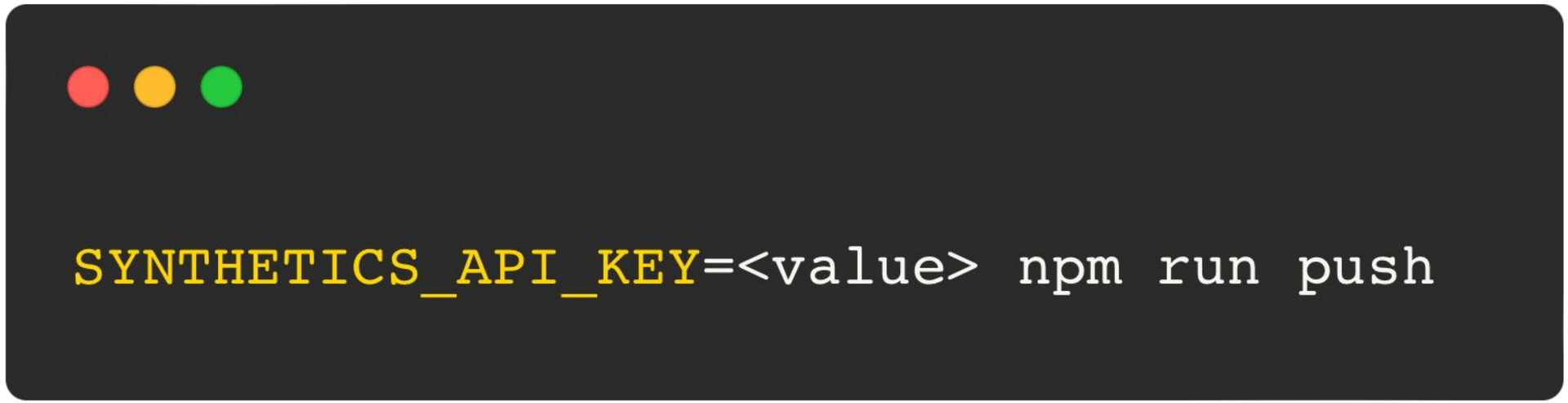
No Reply - Elastic . 3	**"Shopping Cart Test for React Elk Store" (North America - US East) is now up - Elastic Synthetics** - The alert for "Shopping Cart Test fo...	1:58 PM
No Reply - Elastic . 3	**"Shopping Cart Test for React Elk Store" (North America - US East) is down - Elastic Synthetics** - "Shopping Cart Test for React Elk Stor...	1:53 PM



Feb 20, 2025 @ 1:40:05 PM | Failed | error executing step: Cannot read properties of undefined (reading 'add') | 1.6 sec

elastic

Making changes



elastic

Making changes

```
> projects-test@1.0.0 push
> npx @elastic/synthetics push

⚠ Lightweight monitor schedules will be adjusted to their nearest frequency supported by our synthetics in
frastructure.
> Pushing monitors for 'projects-test' project in kibana 'default' space
> preparing 5 monitors
> Monitor Diff: Added(0) Updated(1) Removed(0) Unchanged(4)
> creating or updating 1 monitors (425ms)
✓ Pushed: https://239b09d47e334af58a8179f048fb7361.us-east4.gcp.elastic-cloud.com:443/app/synthetics/monit
ors
```

elastic

Another option 

elastic

Monitors

Create Monitor | Refresh

Overview | Management

Search by name, URL, host, tag, project or location

Up | Down | Disabled | Type 2 | Location 1 | Tags 0 | Frequency 2 | Project 1

Current status

3 Up

0 Down

0 Disabled

Last 6 hours

0 Errors

Last 12 hours

0 Alerts

Showing **3 Monitors**

Sort by Status **Group by** None

elastic

3 Add a script

Use Elastic Synthetics Recorder to generate a script and then upload it. Alternatively, you can write your own Playwright script and paste it in the script editor.

Launch Synthetics Recorder Download Synthetics Recorder

Upload script | Script editor

Select or drag and drop a .js file

Parameters

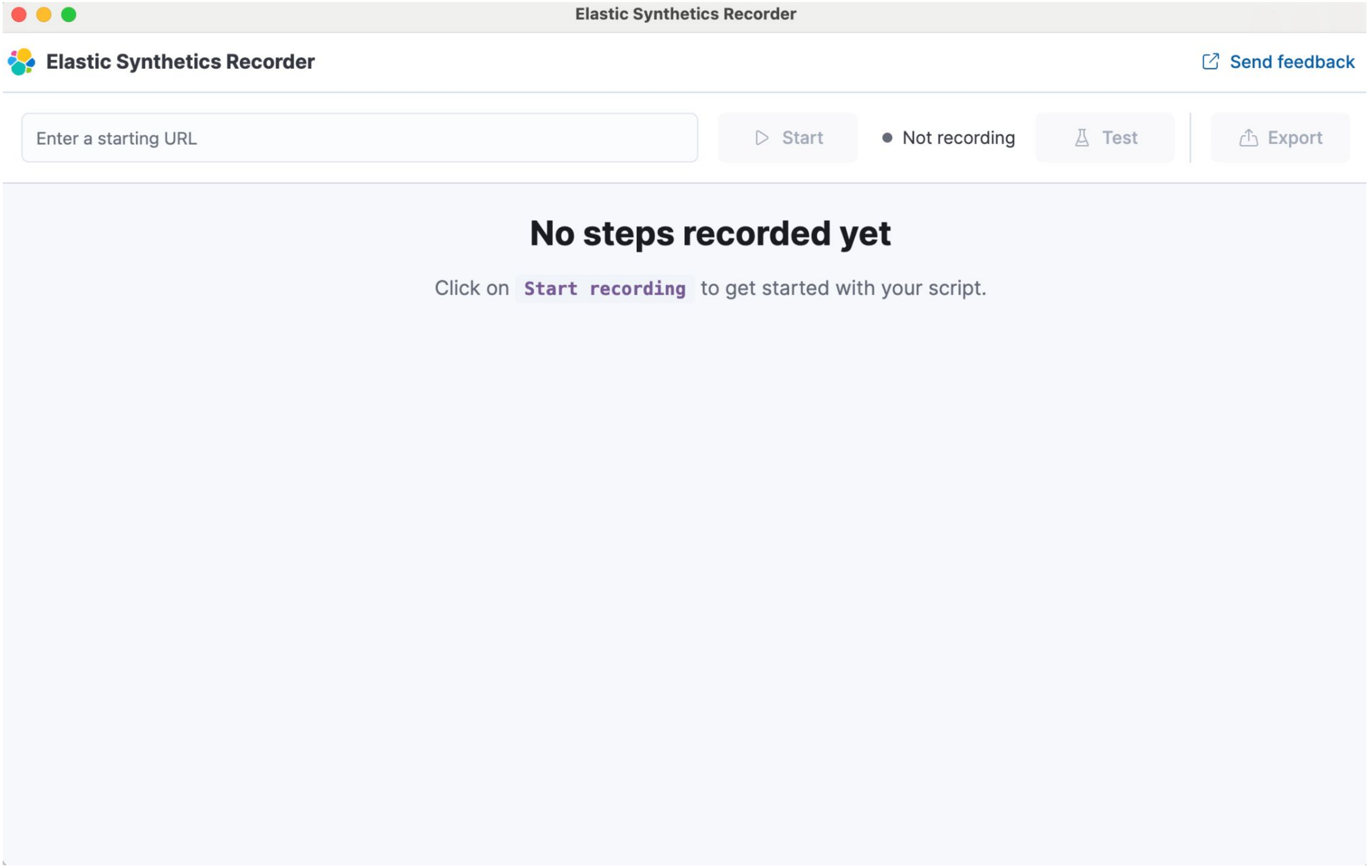
```
1
```

Use JSON to define parameters that can be referenced in your script with `params.value`

> Advanced options

Cancel Run test Create monitor

elastic


Elastic Synthetics Recorder
Elastic Synthetics Recorder
Send feedback
Enter a starting URL
Start
Not recording
Test
Export
No steps recorded yet
Click on Start recording to get started with your script.

demo-store-ivory.vercel.app/products

Incognito

Demo Store

Home

Coffee Mug

$15

A coffee mug!

Buy

Set of Pens

$10

Pens in assorted colors.

Buy

Plush Elk

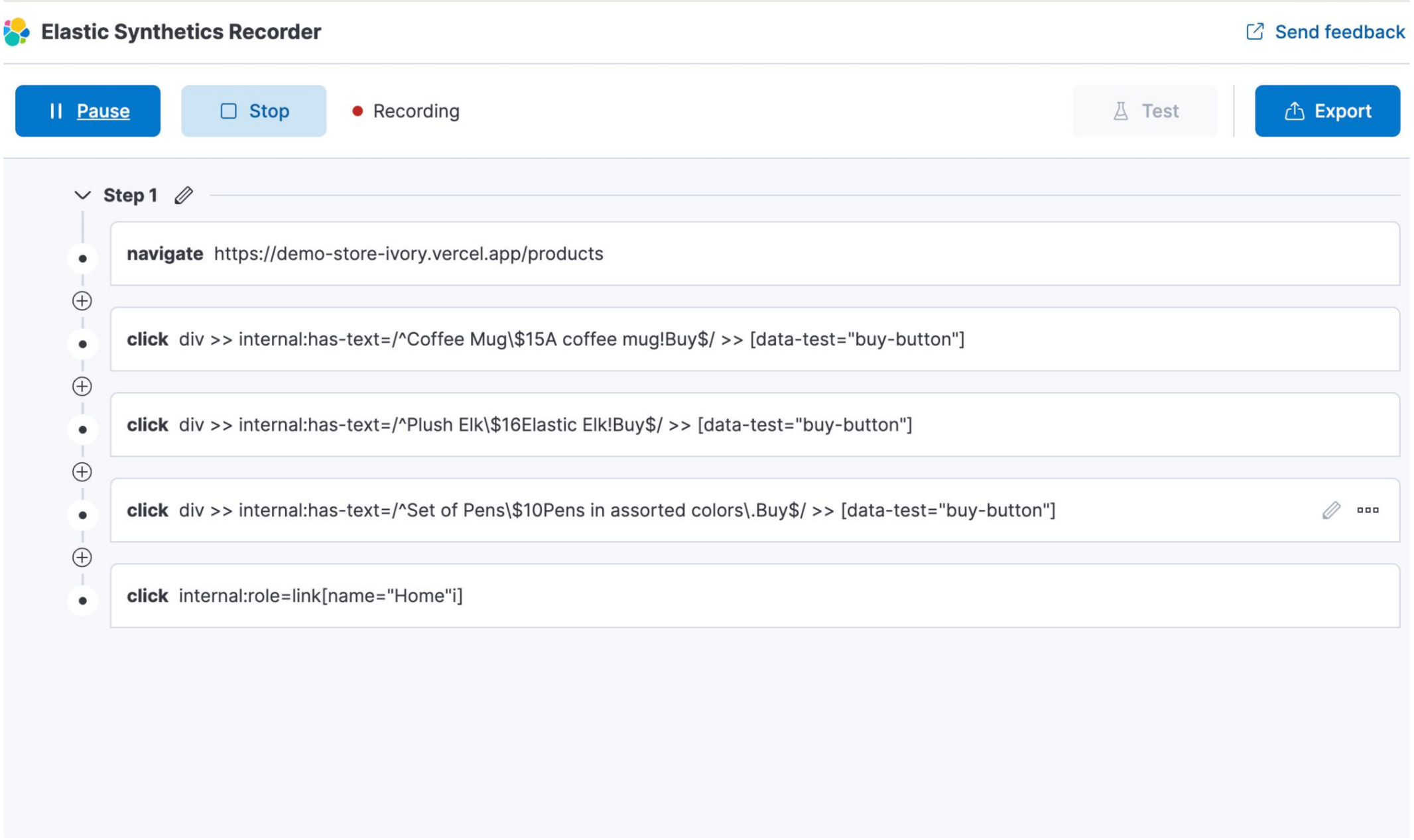
$16

Elastic Elk!

Buy

```
getByText('Coffee Mug$15A coffee mug!BuySet of Pens$10Pens in assorted
colors.BuyPlush Elk$')
```

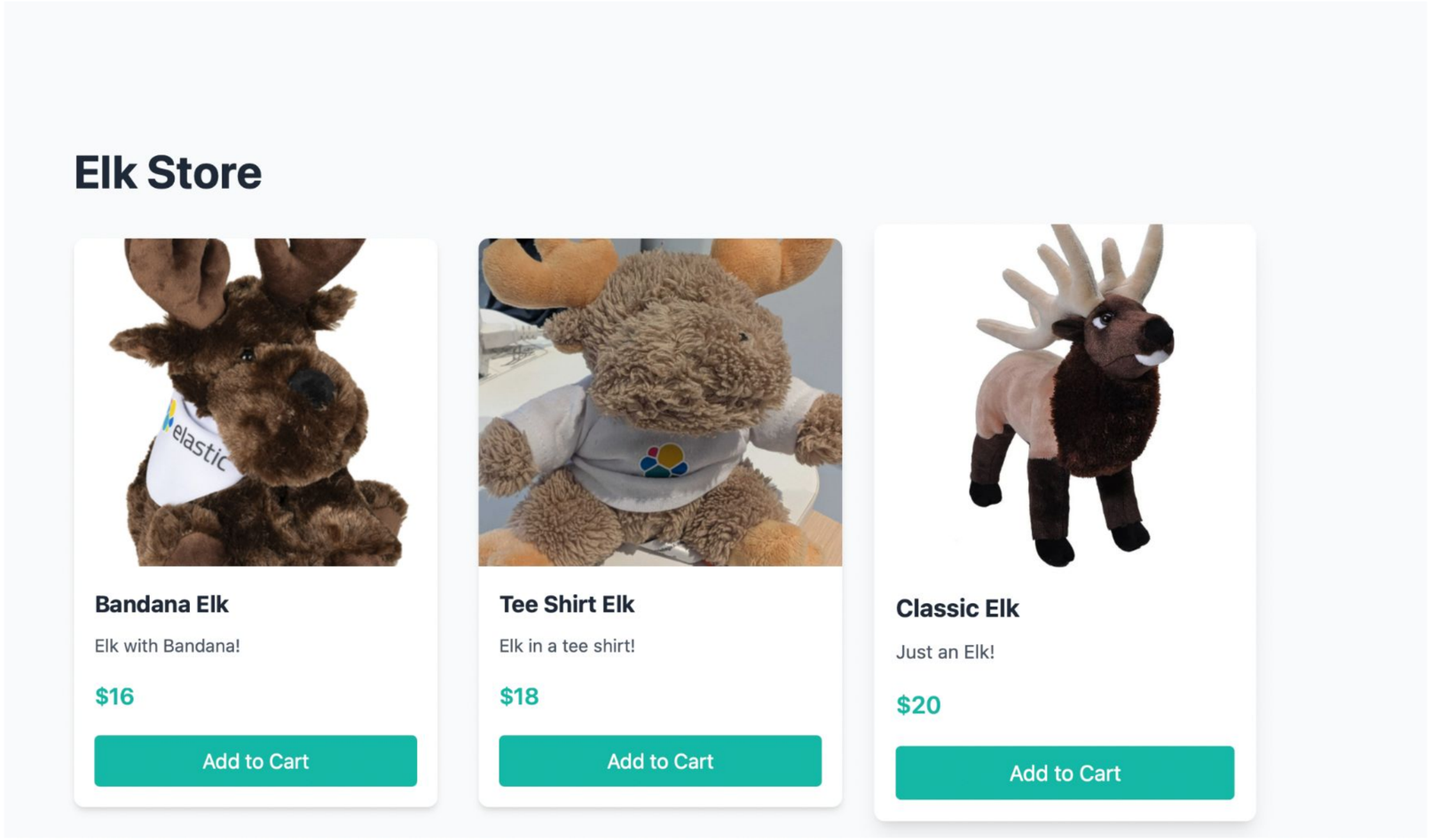
elastic


Elastic Synthetics Recorder
Send feedback
Pause
Stop
Recording
Test
Export
Step 1
navigate https://demo-store-ivory.vercel.app/products
click div >> internal:has-text=/^Coffee Mug\$15A coffee mug!Buy$/ >> [data-test="buy-button"]
click div >> internal:has-text=/^Plush Elk\$16Elastic Elk!Buy$/ >> [data-test="buy-button"]
click div >> internal:has-text=/^Set of Pens\$10Pens in assorted colors\.Buy$/ >> [data-test="buy-button"]
click internal:role=link[name="Home"i]

Tradeoffs with this option ⚠️

Is this the same for React?

https://react-elk-store.vercel.app/

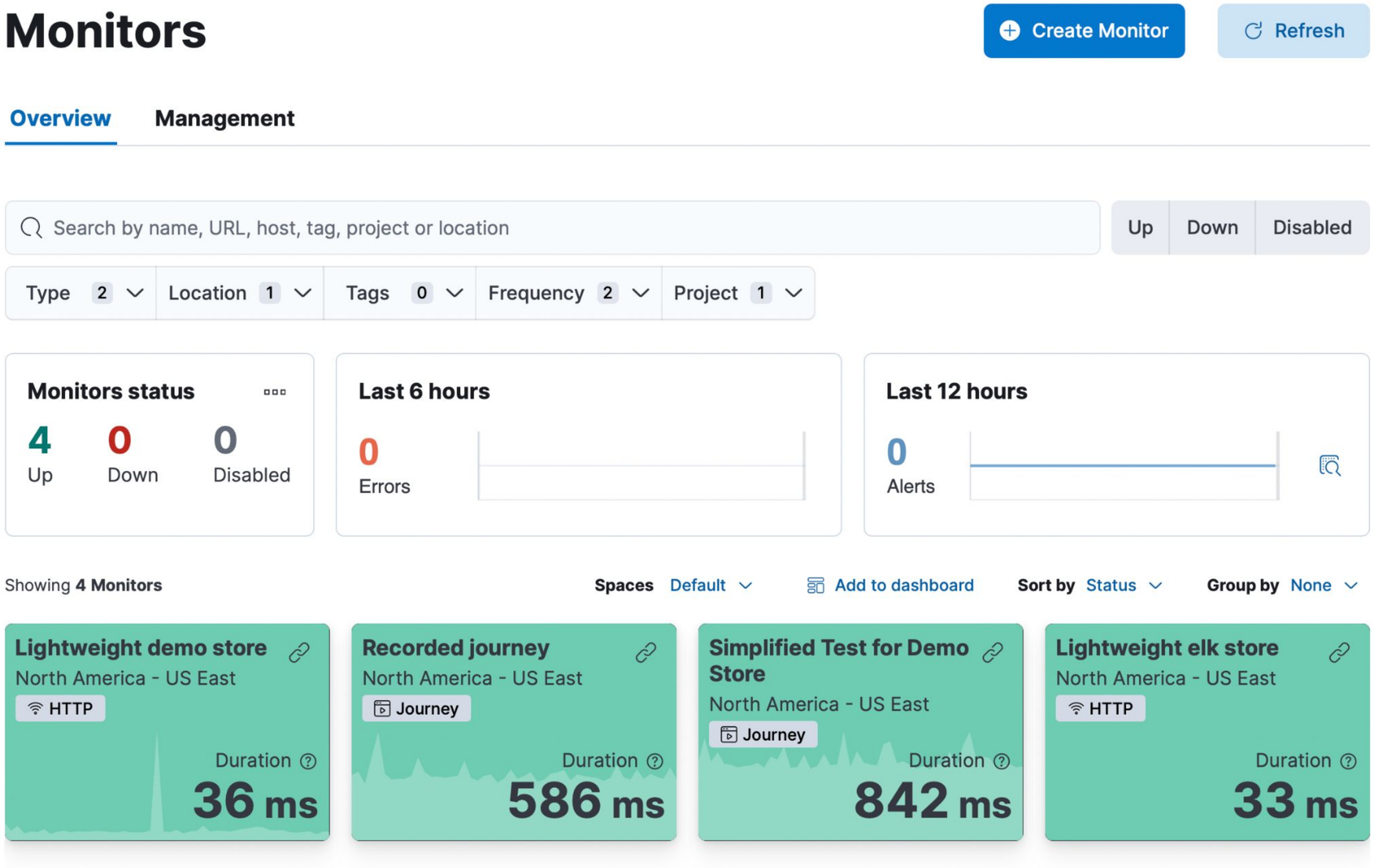


elastic

https://github.com/JessicaGarson/react-elk-store

elastic

Let’s add another heartbeat monitor

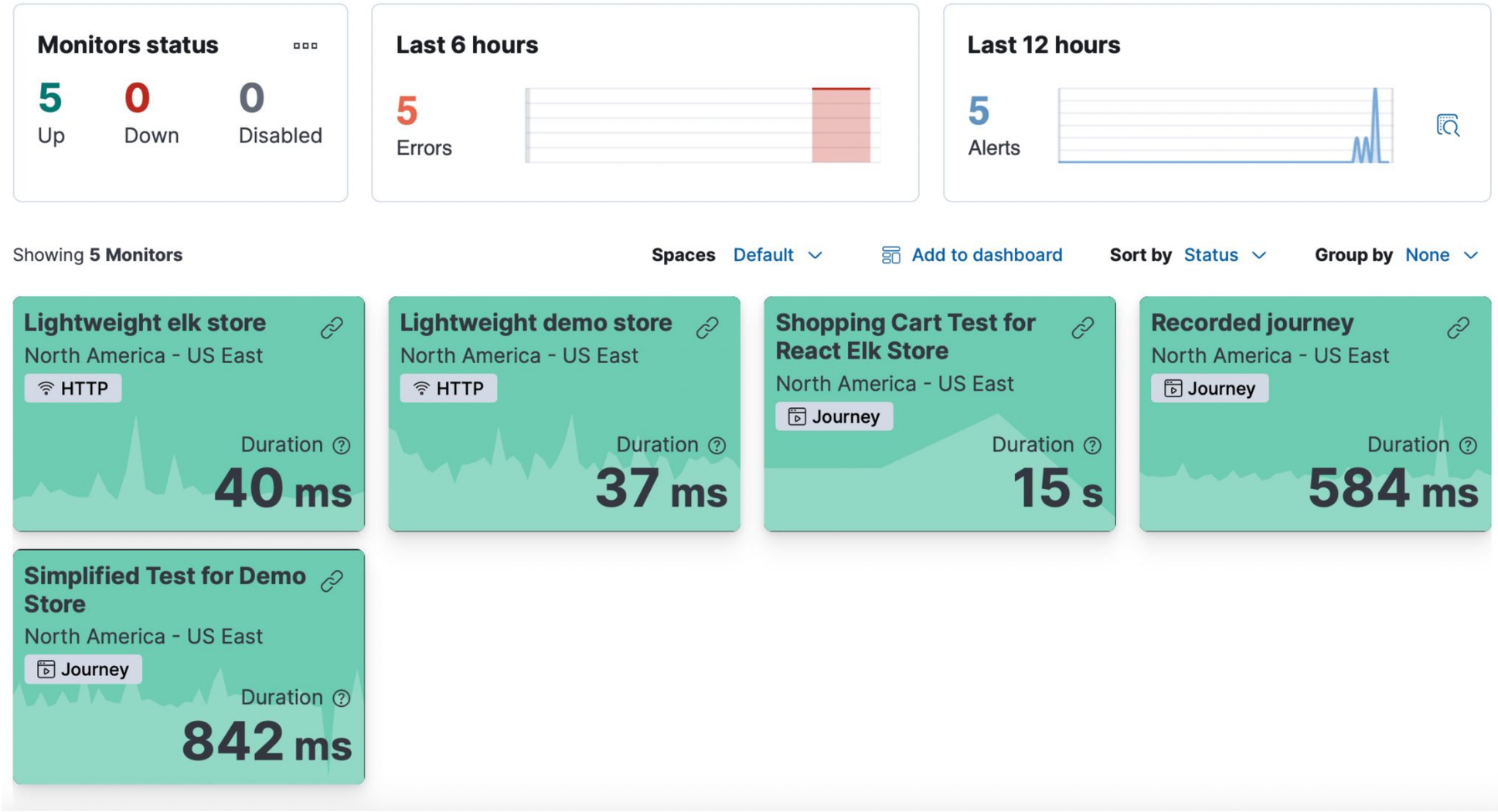


lightweight/heartbeat.yml updated

```yaml
heartbeat.monitors:
- type: http
  name: Lightweight demo store
  id: demo-store-lightweight
  enabled: true
  urls: "https://demo-store-ivory.vercel.app"
  schedule: '@every 3m'
  timeout: 16s
  alert.status.enabled: true
- type: http
  name: Lightweight elk store
  id: elk-store-lightweight
  enabled: true
  urls: "https://react-elk-store.vercel.app"
  schedle: '@every 3m'
  timeout: 16s
  alert.status.enabled: true
```

elastic

Let’s add another journey



```
import { journey, step, monitor, expect, Page } from '@elastic/synthetics';

const TEST_URL = 'https://react-elk-store.vercel.app/';

journey('Shopping Cart Test for React Elk Store', ({ page }: { page: Page }) => {
  monitor.use({
    id: 'react-elk-store-cart-check',
    schedule: 10,
  });

  step('Launch the application', async () => {
    await page.goto(TEST_URL, { waitUntil: 'load', timeout: 15000 });
  });

  step('Check for page load', async () => {
    const storeHeader = page.getByRole('heading', { level: 1 });
    await storeHeader.waitFor({ state: 'visible', timeout: 10000 });
    await expect(storeHeader).toHaveText('Elk Store', { timeout: 10000 });
  })
```

elastic

```
step('Add item to cart and check cart count', async () => {
   // Find first product card and its Add to Cart button
   const firstProductCard = page.getByTestId('product-card').first();
   const addToCartButton = firstProductCard.getByTestId('add-to-cart-button');
   await addToCartButton.waitFor({ state: 'visible', timeout: 7000 });

   // Click the Add to Cart button
   await addToCartButton.click();

   // Look for the cart button and check its badge
   const cartButton = page.getByTestId('cart-button');

   // Wait for the cart quantity to appear in the button
   await page.waitForFunction(() => {
     const cartBtn = document.querySelector('[data-testid="cart-button"]');
     return cartBtn && cartBtn.querySelector('span.absolute');
   }, { timeout: 7000 });

   // Verify the cart badge shows "1"
   const cartQuantity = await cartButton.evaluate(button => {
     const badge = button.querySelector('span.absolute');
     return badge ? badge.textContent.trim() : null;
   });

   expect(cartQuantity).toBe('1');
 });
```

elastic

```
  step('Proceed to checkout', async () => {
    // Open the cart first
    const cartButton = page.getByTestId('cart-button');
    await cartButton.click();

    // Wait for the checkout button to appear in the cart sidebar
    const checkoutButton = page.getByText('Checkout');
    await checkoutButton.waitFor({ state: 'visible', timeout: 7000 });
    await expect(checkoutButton).toBeVisible();
    await checkoutButton.click();
  });

  step('Monitor page performance', async () => {
    const performance = await page.evaluate(() => JSON.stringify(window.performance));
    console.log(performance);
  });
});
```

Next steps

elastic

- Enhancing the coverage of monitors
- Adding a slack notification when monitors fail
- Adding authentication tests

Key takeaways

- It’s easy to get up and running with extending the setup wizard
- It can let you know if parts of the user experience are not working as expected
- An SRE focused tool that provides visibility into critical flows

elastic

https://github.com/JessicaGarson/Synthetic-Monitoring-Unpacked

elastic


Thank you!