

Our test instability prevent us from delivering

Sofía LESCANO CARROLL



@SofLesc





Raise your hand if your tests make you angry or are painful to write.



I have the solution!



No tests, no problem !

(Obviously, it's a joke ;))

Sofía Lescano Carroll

Senior Developer

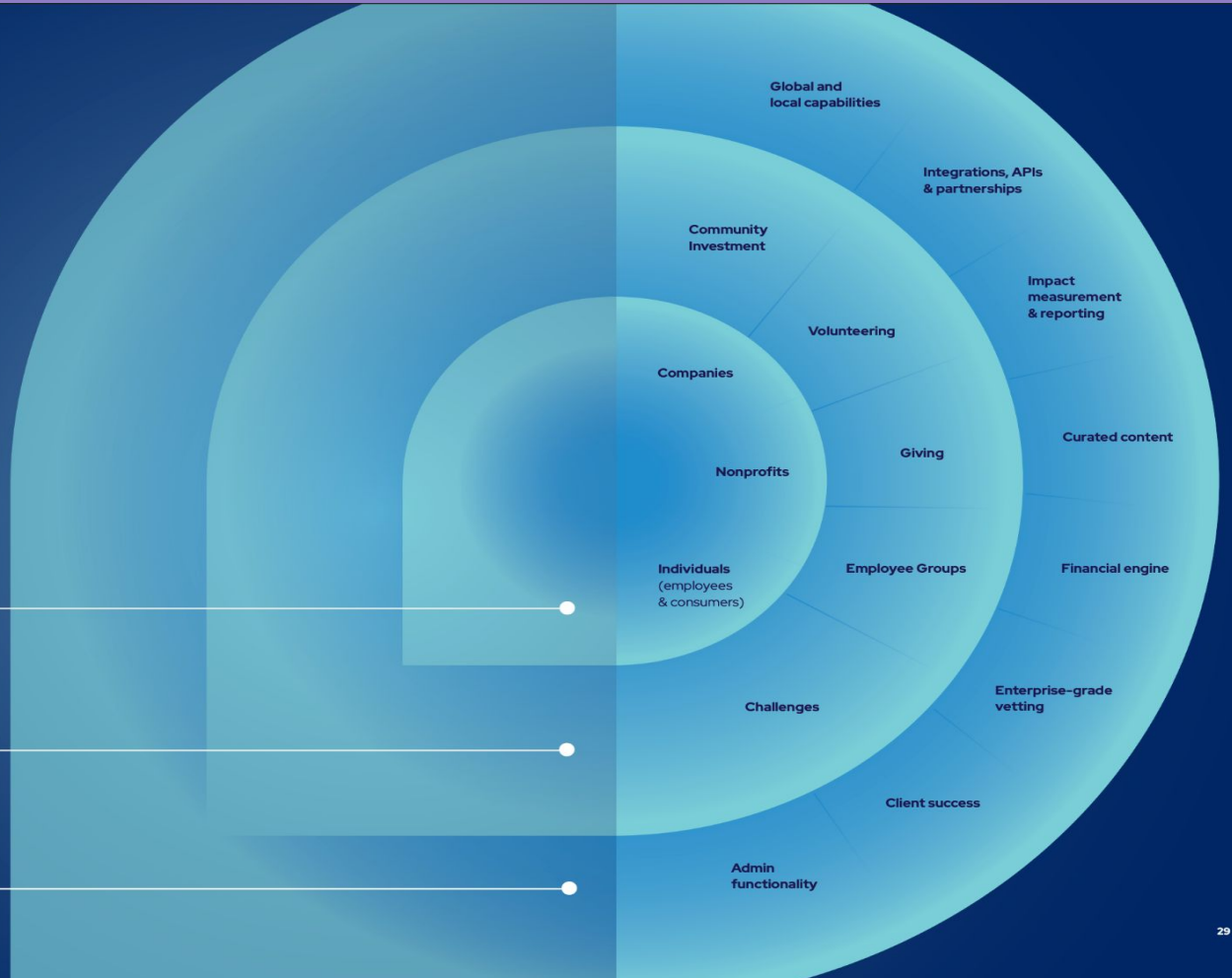


Meet the Benevity platform

Who we serve

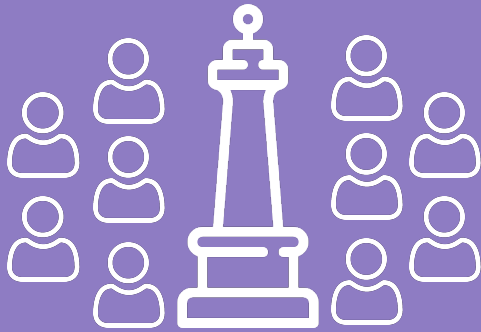
Solutions

Capabilities



Why do we even write tests ?

Why it is useful to write tests ?



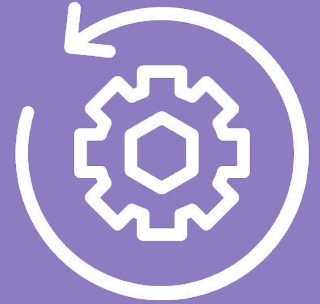
Difficulty knowing the whole application



Verify the app is working as expected

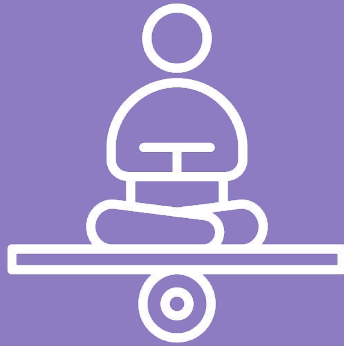


Prevent regressions



Automation

Why it is useful to write tests ?

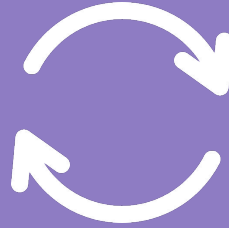
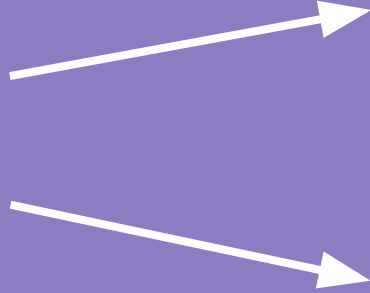


Peace of mind during development
and confidence in the code

If the tests are correctly written



Failing test



Test update



Detected an issue in
the app



What if I can't
trust my tests ?

What is a flaky test?



Flaky test

A test is said to be flaky or unreliable when it gives both positive and negative results despite no changes to the code or the test.

Flaky test

Since the result of the flaky test is non-deterministic, reproducing and correcting the error case can be very complex.

Flaky test

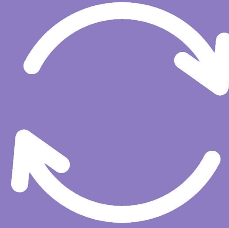
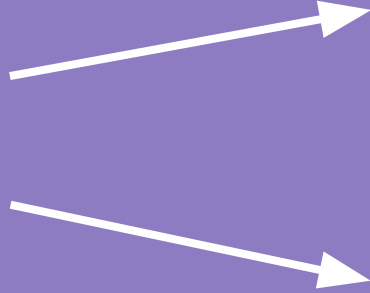
A flaky test can pass

$\frac{1}{2}$, $\frac{3}{4}$, $\frac{199}{200}$...

If tests are correctly written



Test does not pass

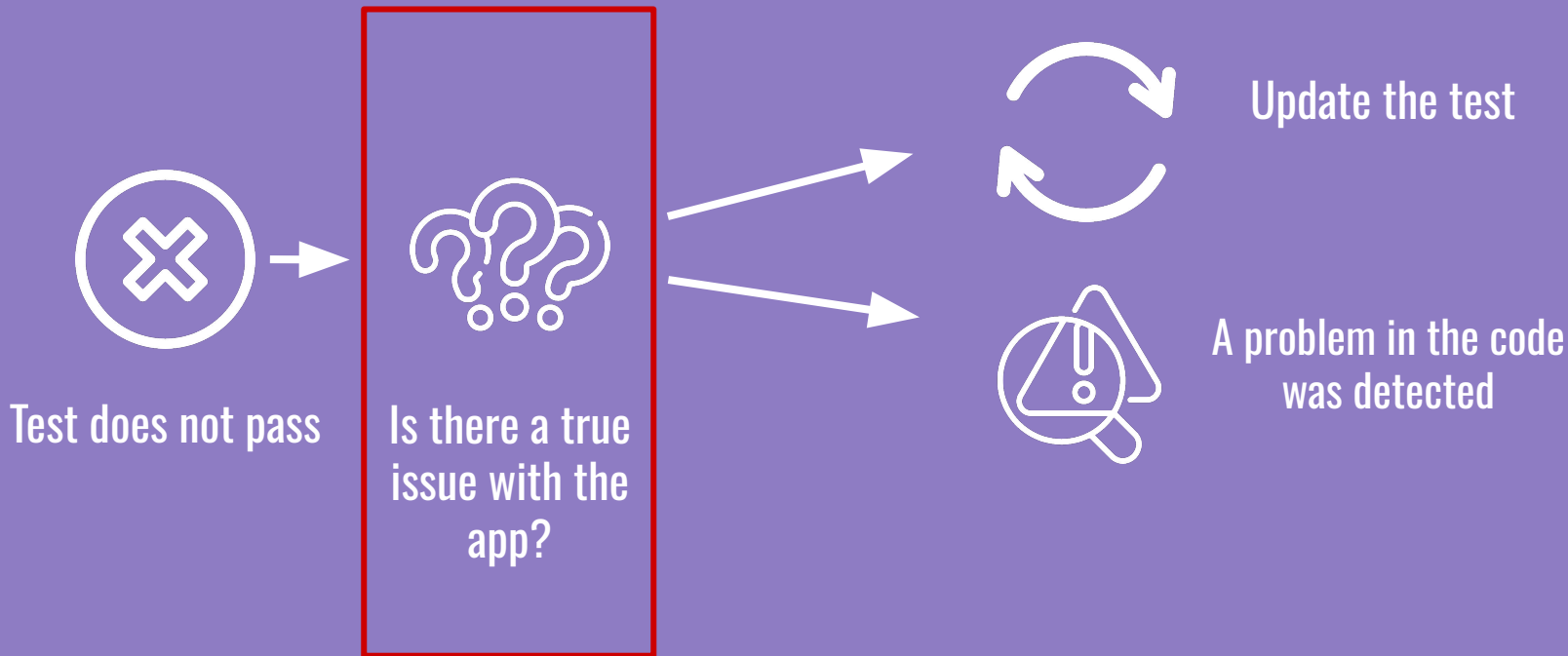


Update the test



A problem in the
code was detected

What if I can't trust my tests ?

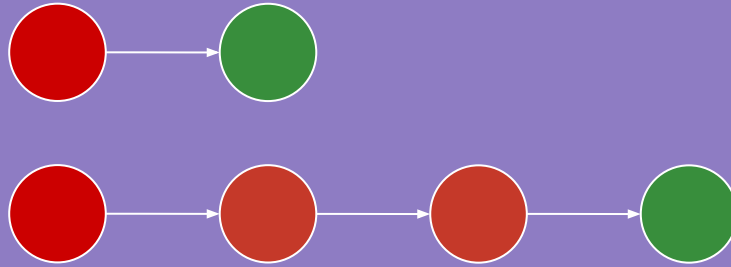


Is there a true problem in my code ?



Multiple executions do not
always allow for a
conclusion

Is there a true problem in my code ?



Multiple executions do not always allow for a conclusion

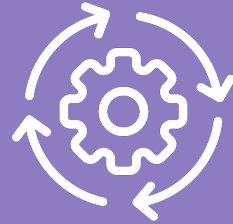
Is there a true problem in my code ?



Multiple executions do not
always allow for a
conclusion

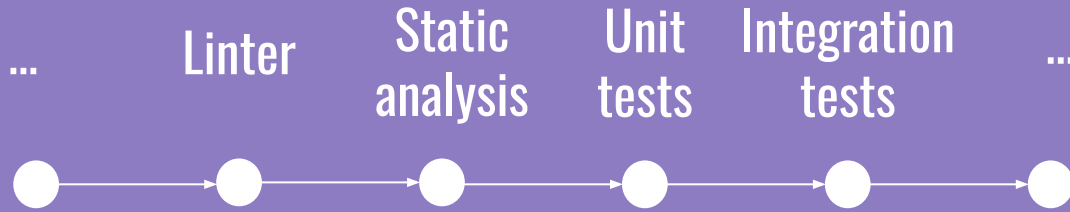


Understand the goal of the
test even when it was not
written by me



Time and ability to run
the tests

Development cycle



We merge and deploy only when all tests pass

The impact of flaky tests



Uncertainty about the
status of the
application



Degradation of
capacity to deliver
value



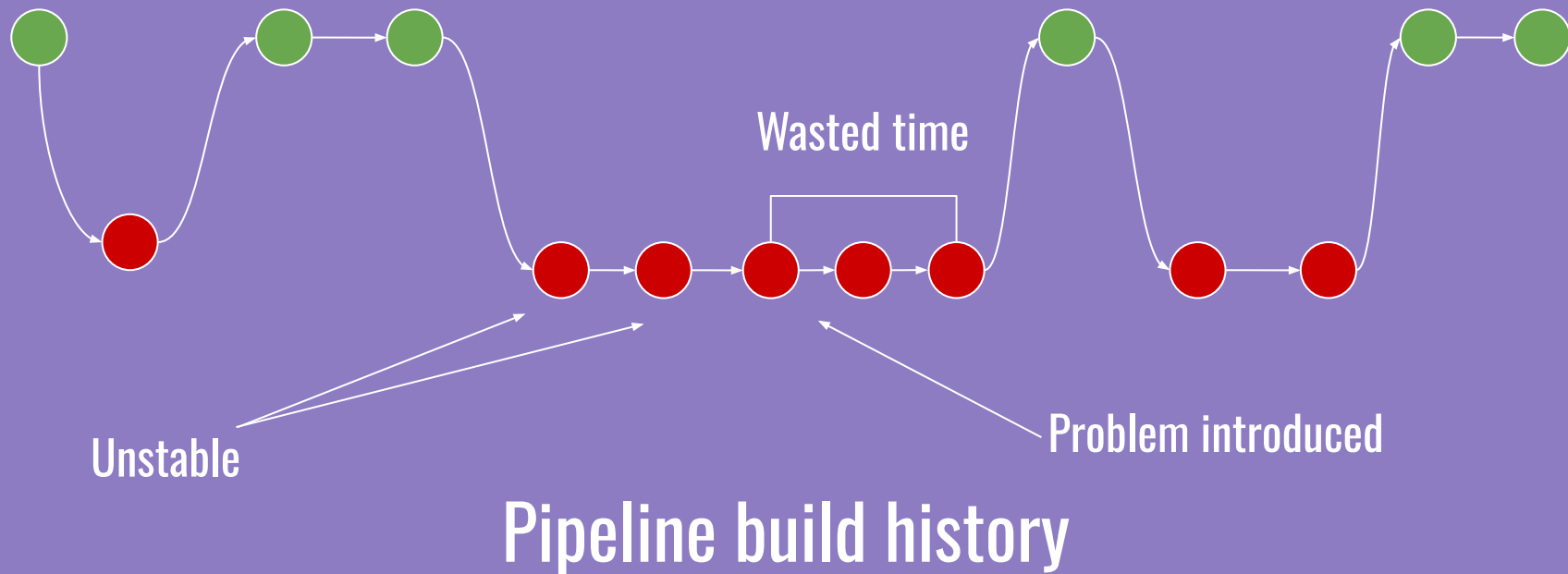
Developer
frustration

The impact of flaky tests



There is so much noise that we no longer take real problems seriously.

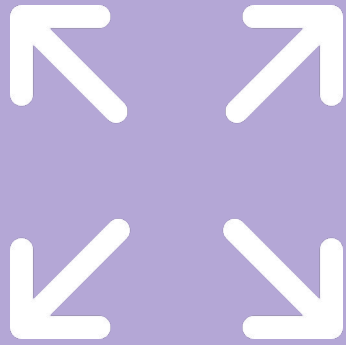
The impact of flaky tests



The impact of flaky tests



- The test does not pass
- It's normal ... it may just be flaky



A known problem
that can spread very
quickly



Raise your hand if from time to time when you encounter a problem you pretend not to have seen it

tech-backend-chapter – Jan 6th

damn you flaky test!

tech – Feb 23rd

2:21 PM
Oh, flaky tests strike again...

For those who's watching

- We disabled the Manta test because the behaviour is actually flaky

... now more

👍 1 🗨️

11:43 AM

These flaky tests are getting out of hands :

< 1 minute ago

the flaky test monster is a hydra, you cut of one head and two grow in its place!

Sep 7th, 2023 at 2:33 PM

yeah, it's very flaky, now it went the other way around

tech – Feb 23rd

I restarted the workflow on CircleCI, apparently it was just *another* flaky test

😄 1 reaction

know

Sorry for the delay, I hate flaky tests, it's the 4th time in a row I launch tests execution and they are failing due to flaky tests !!

what a waste of time 🙄 (edited)

This seems to be a flaky test

when flaky test will have a good mood for my PR ><

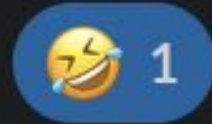
- Build failed.
- The test has previously been reported as flaky. - <https://pm.>

We will go in production in 15 minutes, we just fix a flaky test

A known problem: perseverance and patience

14 minutes ago

after 6 try, test are ok now





Raise your hand if you've ever copied code that existed in your project to create new code.

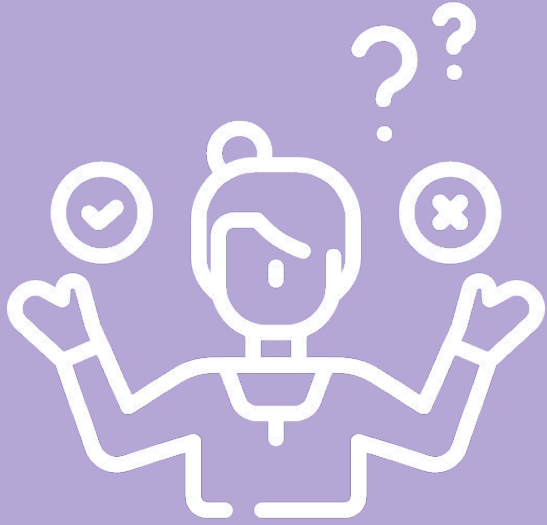
A worsening problem



Create new tests based
on old ones

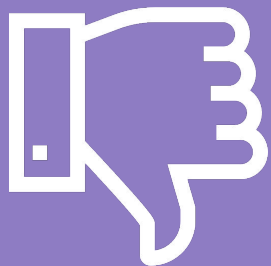


New features > stability of the
technical stack

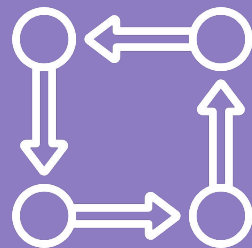


How to Write a Good Flaky Test

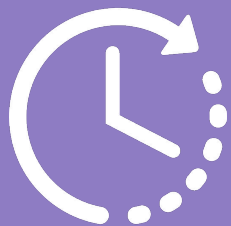
Main causes of flaky test



Poorly written tests (dates,
business rules, etc.)



dependencies and
order problems



Timing issues



External dependencies

Main causes of flaky test



Poorly written tests (dates,
business rules, etc.)

Main causes of flaky test



Not so great
tests



Not so great
results

Main causes of flaky tests: poorly written tests

```
public function testRenewedBudgetEnd()
{
    // Arrange.
    $budget = self::budgetWithNoLimits();

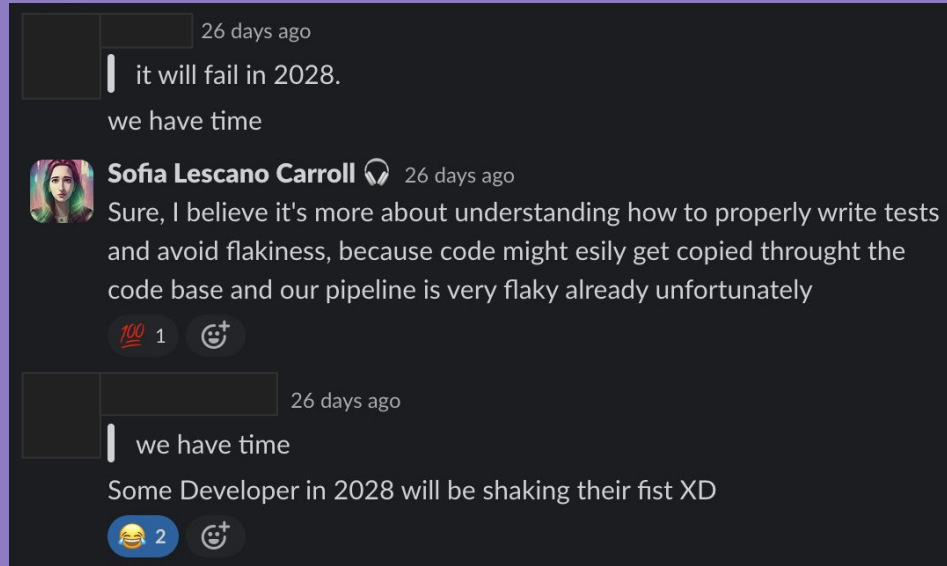
    // Act.
    $newStart = $budget->getRenewedBudgetEnd();
    $dateDiff = $budget->getDuration()->getEndTimeStamp()->diff($newStart);
    $daysInterval = $dateDiff->days;

    // Assert.
    $this->assertEquals(366, $daysInterval, 'Renewed budget end date should 1 year more than original budget end date');
}
```



The test that stops working on March 1st

Main causes of flaky tests: poorly written tests



A problem for the developer of the future

Main causes of flaky tests: poorly written tests

```
public function testExportInvoices(){
```

```
...
```

```
$firstInvoice = Invoice::factory()->for($brand)->create(['created_at' => Carbon::now()->subMonths(2)]);
```

```
$secondInvoice = Invoice::factory()->for($brand)->create(['created_at' => Carbon::now()->subMonth()]);
```

```
$thirdInvoice = Invoice::factory()->for($brand)->create(['created_at' => Carbon::now()]);
```

```
$subject->exportInvoices($brand);
```

```
...
```

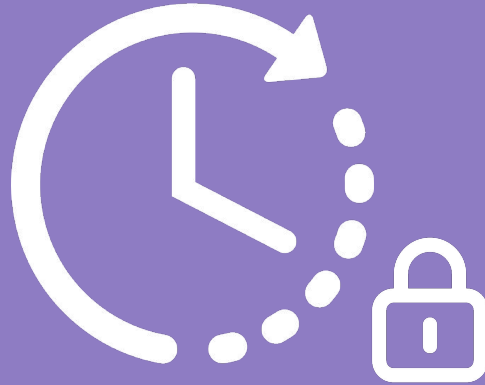
```
}
```

31 march -> 31 february ?!

->SubMonthNoOverflow

"No, but Sofia, don't you know that we're not allowed to test on the 31st of each month?"

Main causes of flaky tests: poorly written tests



Solution track: using a clock interface

Main causes of flaky tests: poorly written tests

```
// Des règles business à prendre en compte : pas de vente entre France et Angleterre
public function testAddItemToCart(){
    $client = Client::factory()->create(); // Le pays est aléatoire
    $brand = Brand::factory()->create(); // Le pays est aléatoire
    $product = Product::factory()->for($brand)->create();
    $subject->addItemToCart($client, $product);
    self::assertCount(1, $client->cart);
    ...
}
```

Random data yes, but not always

Main causes of flaky tests: poorly written tests

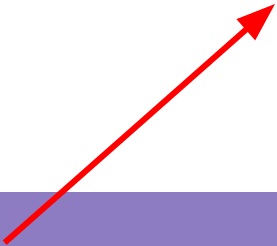
```
public function myTest() {  
    ...  
    $firstVariable = uniqid(); // 65691234abcde  
    $secondVariable = uniqid(); // 6569123456789  
    $subject->functionToTest($firstVariable, $secondVariable);  
    ...  
}                               uniqid('first_variable') -> first_variable_6569123456789
```

Randomness with `uniqid()`: string that could be cast as an int

Main causes of flaky tests: poorly written tests

```
// Les ids de mysql ne sont pas remis à zéro entre les tests

public function testCreateBrand(){
    $brand = $subject->createBrand('mybrand@mycompany.com', 'FR')
    $logger->shouldReceive('info')->with('New brand created with id 1')->once();
    ...
}
```



Sqlite Entity IDs -> MySQL

1 test to correct ok, 234 tests to correct... not fun

Dependency with a Feature Flag

1/ Introduces a feature behind a Feature Flag

2/ The test is not explicit about the value of the FF

3/ If we change the FF on the environment the test no longer passes

Main causes of flaky test



Timing issues

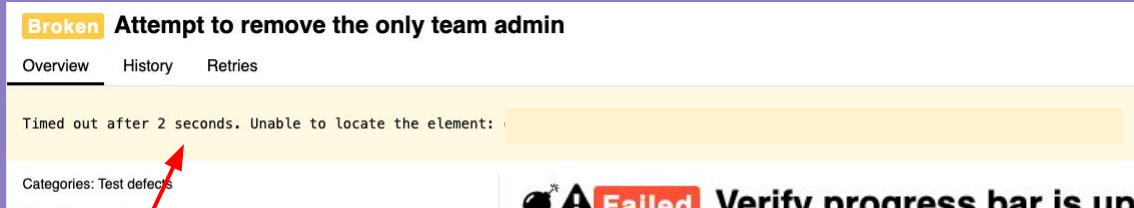
Main causes of flaky test: timing issues

Broken Attempt to remove the only team admin

Overview History Retries

Timed out after 2 seconds. Unable to locate the element: ...

Categories: Test defects



Failed Verify progress bar is updated with giving account donation

Overview History Retries

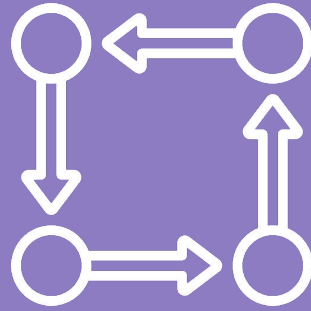
The following 4 assertions failed:

- 1) [the friendraiser total] expected:<"\$[23]6.11"> but was:<"\$[18]6.11">
- 2) [the friendraiser donation percentage] expected:<"[7]4"> but was:<"[9]4">
- 3) [the friendraiser matching percentage] expected:<"[2]5"> but was:<"[]5">
- 4) [the friendraiser funded message] expected:<"1[5]% Funded"> but was:<"1[2]% Funded">



Waiting time exceeded or reading ahead

Main causes of flaky test



**Test dependency
and ordering**

Main causes of flaky testing: dependent tests

```
// Une base de données partagée par vos tests

public function testCreateBrand(){
    $brand = $subject->createBrand(self::BRAND_EMAIL, self::BRAND_COUNTRY);
    ...
}

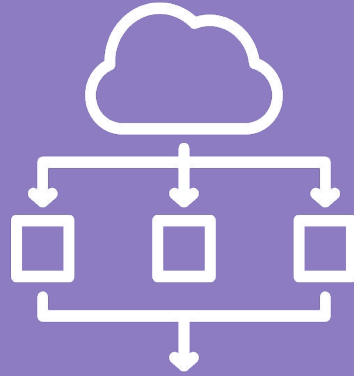
public function testEditBrand(){
    $brandToEdit = $repository->findByEmail(self::BRAND_EMAIL);
    $editedBrand = $subject->editBrand($brandToEdit, ['country' => 'ES']);
    ...
}

public function testListBrands(){
    $brands = $subject->listBrands();
    self::assertEquals('ES', $brands[0]->country);
    ...
}
```

Main causes of flaky testing: dependent tests

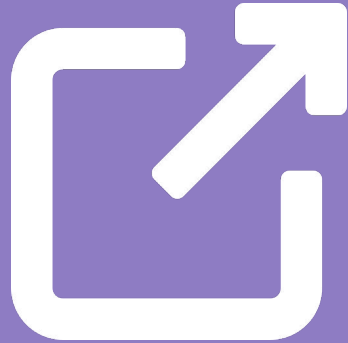


Lack of clean up
(data, configurations)



Concurrency

Main causes of flaky test



External dependencies

Main causes of flaky testing: external dependencies



Call for a payment sandbox that may be unstable

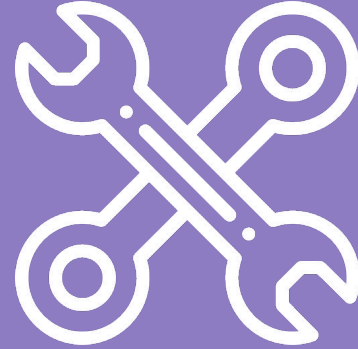


How to
tackle this
problem

Understanding the problem



A human
problem



A technical
problem

Being aware



With a large code base and lots of developers, this will happen sooner or later.

How to tackle the problem



Understanding the
impacts



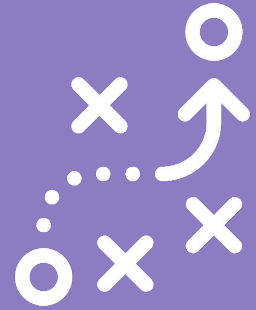
Improving testing
practices



Measure



Have the right
tools



Establish a
strategy

How to tackle the problem



Developers

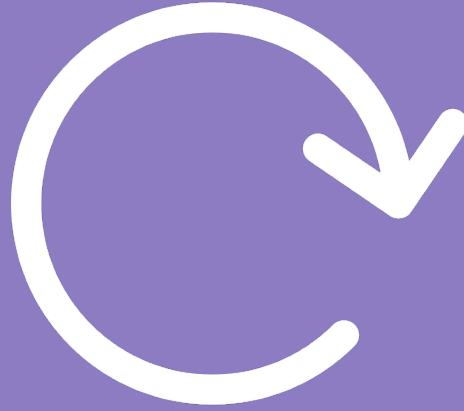


Understanding the
impacts



Product &
management

At the developer level



It's easier to replay the test and hope it passes.

At the company level



A flaky test can potentially fail on every running workflow (PR, deployment, etc.) until it is fixed or ignored.

At the company level

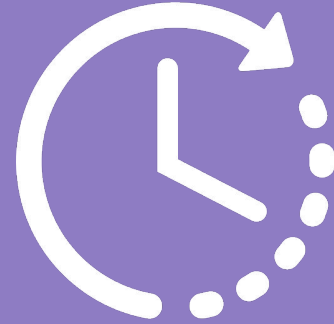


Waste of time and human and software costs

At the company level



A collective effort



Allocate time

Quantify the problem

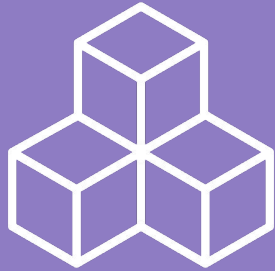


"If you can't measure it, you can't manage it" - Peter Drucker

Quantify the problem: make it visible



Time spent
on it



Occurrences



Consequence
on releases



Mesure
evolution

How to tackle the problem



Improving testing practices

Improving testing practices



**Train
developers**



**Share the errors
encountered**



**Review the tests,
not just the code**

How to tackle the problem



Having the right tools & learning how
to use them

Having the right tools & learning how to use them



Continuous Integration
and Deployment



Infrastructure and
application monitoring



Automation test
reporting tool

Detecting Flaky Tests with Allure and CircleCI

Detecting different results
with the same code


Broken Verifies that Giving Opportun
See bug DEV-32802

Overview History Retries

Success rate 99.28% (558 of 562)

broken	3/5/2024 at 10:46:43
passed	3/4/2024 at 18:43:27
passed	3/4/2024 at 17:13:07
passed	3/4/2024 at 15:12:14
broken	3/4/2024 at 13:13:47
passed	3/4/2024 at 11:14:42
passed	3/1/2024 at 21:02:22
passed	3/1/2024 at 18:55:44
skipped	3/1/2024 at 17:11:18
passed	3/1/2024 at 15:10:51
passed	3/1/2024 at 14:46:52
passed	3/1/2024 at 11:55:02
passed	3/1/2024 at 9:08:58
passed	2/29/2024 at 16:28:31
passed	2/28/2024 at 18:35:43
passed	2/28/2024 at 18:02:26
passed	2/28/2024 at 15:58:53
passed	2/28/2024 at 15:36:32
passed	2/28/2024 at 14:33:39
passed	2/28/2024 at 10:15:39

! 1 test failed out of 16481

▼ testBrandCanViewOrder with data set "status-brand-confirmed"  **FLAKY**

Detecting Flaky Tests with CircleCI

ⓘ Tests metrics are calculated based on 100 most recent workflow runs

Filtering by date range does not apply to tests.

Avg. tests per run

24,611

[See more ↓](#)

Flaky tests detected

346

[See more ↓](#)

Test failures

9

[See more ↓](#)

Slowest test P95 Duration

53s

[See more ↓](#)

Most Failed Tests

Among 100 most recent workflow runs

← Most failed

Test	Job	Duration (p95)	Runs	Success Rate
testAlertLowStripeBalanceWithInsufficientBalance with data set "GBP" FLAKY	test-php-sta...	6s	98	93%
testAlertLowStripeBalanceWithSufficientBalance with data set "GBP" FLAKY	test-php-sta...	6s	98	94%
testCreate FLAKY	test-php-sta...	1s	97	99%
testDirtyDiscountRate FLAKY	test-php-sta...	15s	97	99%

Monitoring the overall status of the application

Detect Flaky Tests with Datadog

Commits

Search by commit SHA

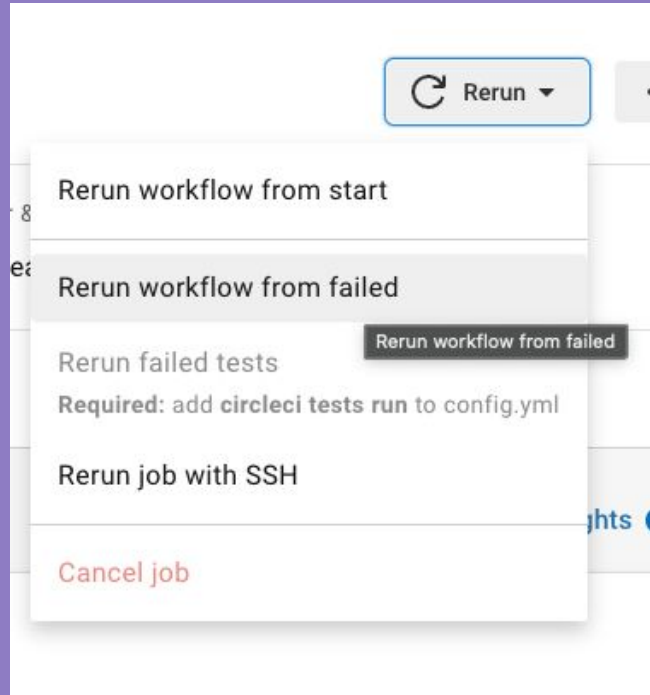
Commits per page: 5

COMMIT	FAILED	PASSED	SKIPPED	NEW FLAKY	REGRESSIONS	FAILURE %	TOTAL TESTS	TOTAL TIME	LAST UPDATED
Revert: Add new payment method with b... 7cdbdac LATEST	3	159	0	1	0	1.85%	162	24 min 12 s	1 hour ago
Improve performance on integration tests 4041638	3	159	0	0	0	1.85%	162	23 min 7 s	1 hour ago
New feature: order products by available stock 56f304f	3	159	0	0	0	1.85%	162	24 min 26 s	1 hour ago
Update dependency used to minify css assets ceaafb4	3	159	0	1	0	1.85%	162	23 min 24 s	6 hours ago
Improve performance on integration tests 94ec285	3	159	0	1	0	1.85%	162	23 min 40 s	6 hours ago

Ignore flaky tests

Ignore a test detected by mistake

Restart flaky tests with CircleCI



Rerun a subset
of tests or the
failing step to
save time

Avoiding certain behaviors through code

**Using a static analyzer
(PHPStan)**

**Implementing our own rules to secure the
code and help your developers**

Avoiding certain behaviors through code

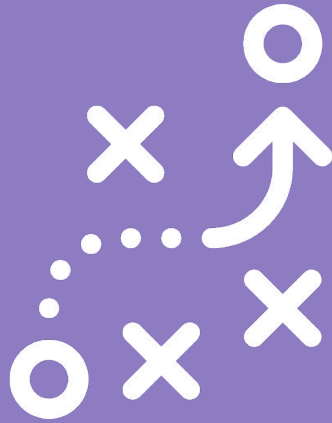
```
public function myTest {  
  ...  
  
  $date = Carbon::now();  
  $previousMonth = $date->subMonth(); // This will trigger the rule.  
  $safePreviousMonth = $date->subMonthNoOverflow(); // This will not trigger the rule.  
  
  ...  
}
```

----- line xxx -----

Avoid using `subMonth()`. Use `subMonthNoOverflow()` instead to prevent unexpected date changes.

💡 Use `subMonthNoOverflow()` instead.

How to tackle the problem

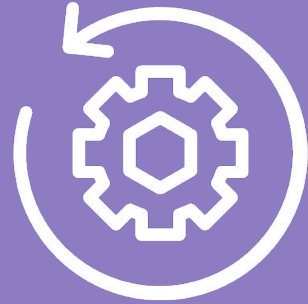


Workaround strategies

How is deployment done?



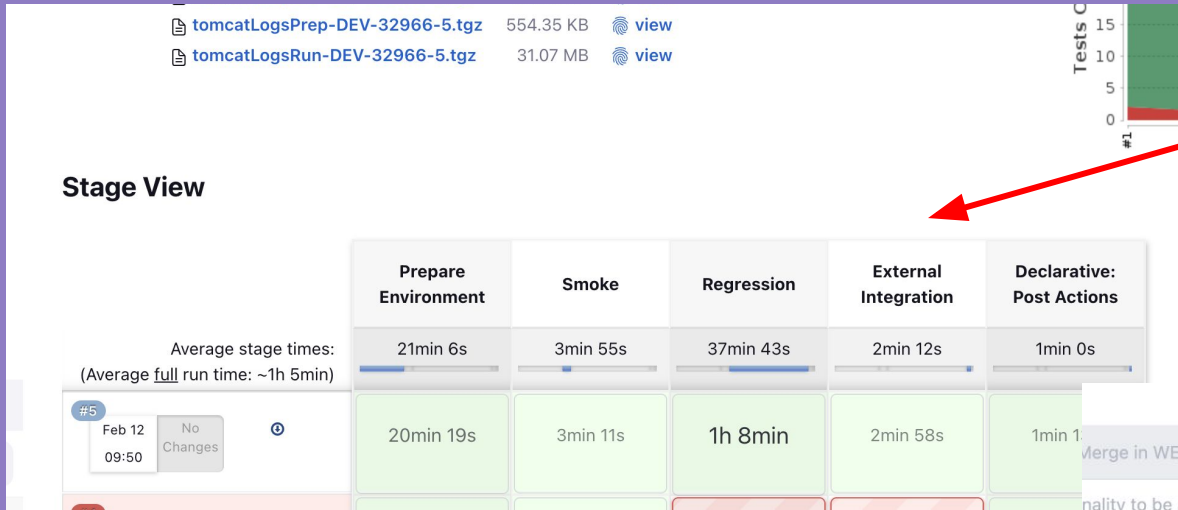
Manual



Automated

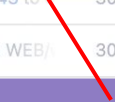
Manual deployment: separate Test Suites

Separate risky tests



	Commit date	Issues	Builds
Merge in WEB/wpg from DEV-3560...	30 Jan 2024	DEV-35605	⬇
ality to be able to retriev	2024.01.30.761	DEV-31165	⬇
2024.01.30.761_FailedExternalInte	30 Jan 2024	DEV-35943	⬇
egration	30 Jan 2024	DEV-35943	⬇
e day. Merge in WEB/wpg from DEV-35943 to	30 Jan 2024	DEV-35943	⬇
e Emails in Budget Notifications Merge in WEB/	30 Jan 2024	DEV-35950	⬇

2024.01.30.761
2024.01.30.761_FailedExternalInte
egration



A specific tag

Manual deployment force deployment



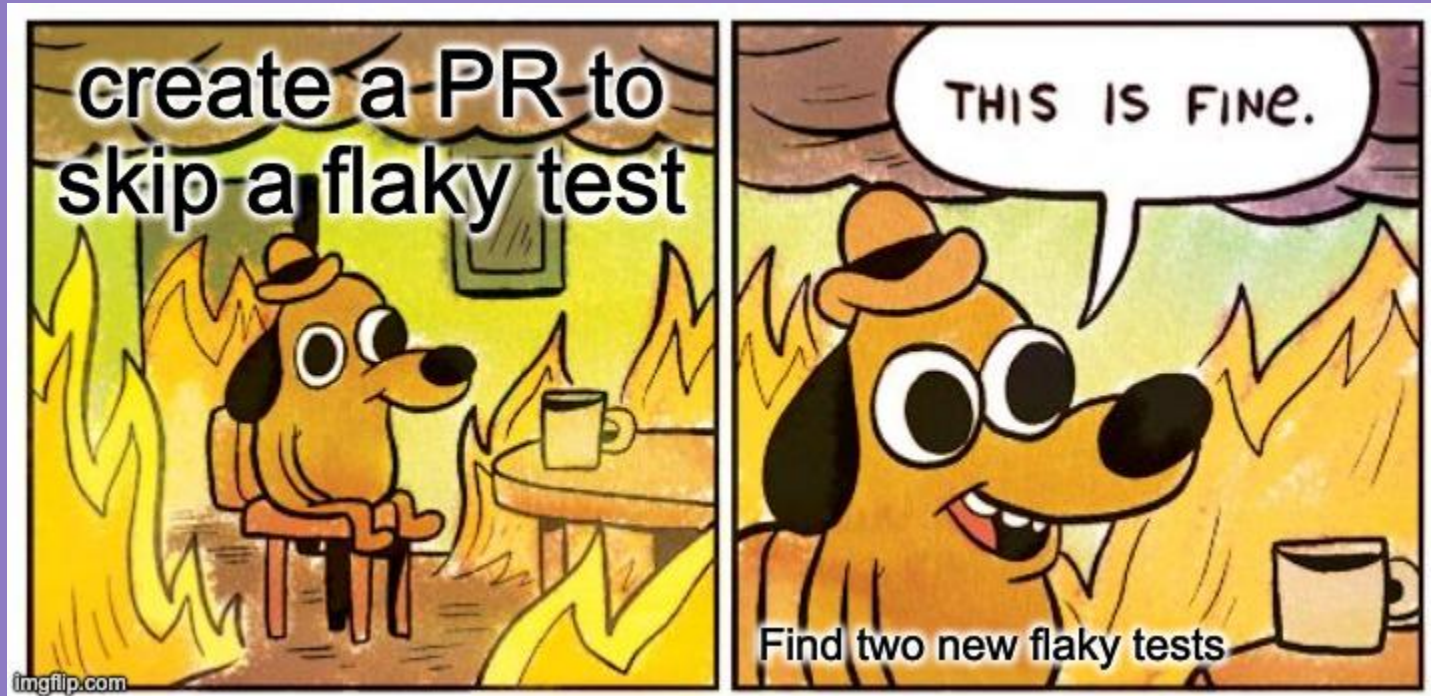
Force merge

Manual deployment force deployment



The fun of unstable testing: having another unstable test that prevents it from being fixed

Manual deployment force deployment



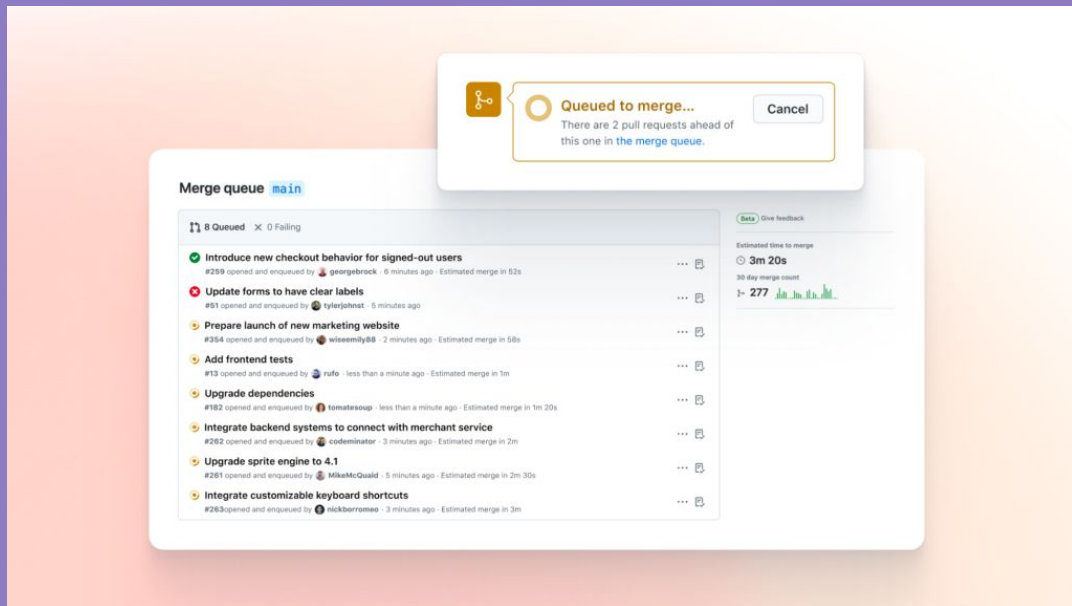
Manual deployment force deployment



Manual verification

With great power comes great responsibility

Automated deployment : github merge queue

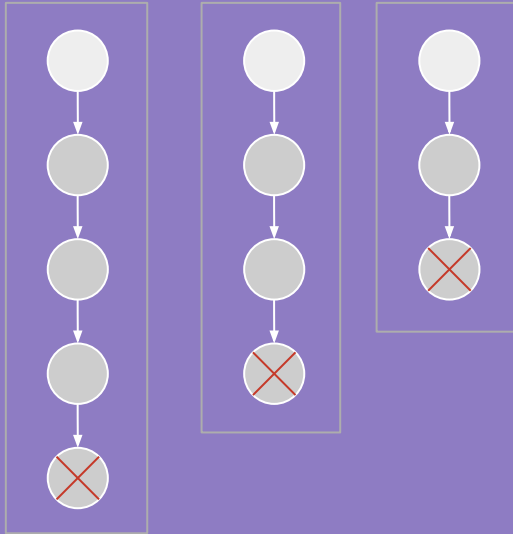


Automation of the merge
and release creation
process

Relies on pipeline and
testing

The problem can no longer
be solved manually.

Automated deployment : merge queue



Find PRs that can be merged together

Will remove a PR if the test suite does not pass

The PRs are compatible but a flaky test is messing with the results

A single flaky test
can block the merge
queue for hours

Avoid blocking the merge queue with a retry strategy

```
//mergequeue/retry-circleci-workflow.sh
```

```
TRIGGER_WORKFLOW="$(curl -s --retry 3 --retry-all-errors  
"https://circleci.com/api/v2/workflow/${WORKFLOW_ID}/rerun" \  
  --header "Circle-Token: $CIRCLE_TOKEN" \  
  --header 'Content-Type: application/json' \  
  --data '{"from_failed": true}')"  
echo "$TRIGGER_WORKFLOW"
```

Give another chance to tests that have not
passed

How to tackle the problem



**A retry strategy increases pipeline execution time
and hides the problem**

How to tackle the problem

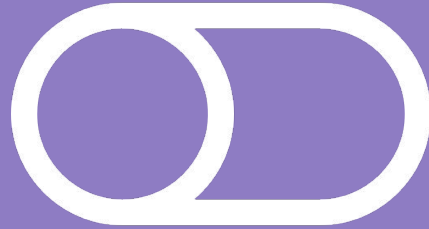


Define a process

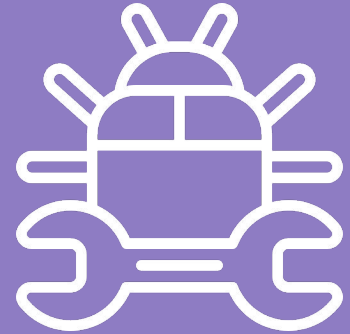
A clear process



**Report a new flaky
test**



**Deactivate the test
/!\ test coverage**



**Urgent technical
debt**



Challenges and limitations

Take control of the situation as soon as possible



The more the problem progresses, the more difficult it is to fix.

Not everyone follows the rules



We have a process but it is easier to replay
the test

Not everyone follows the rules



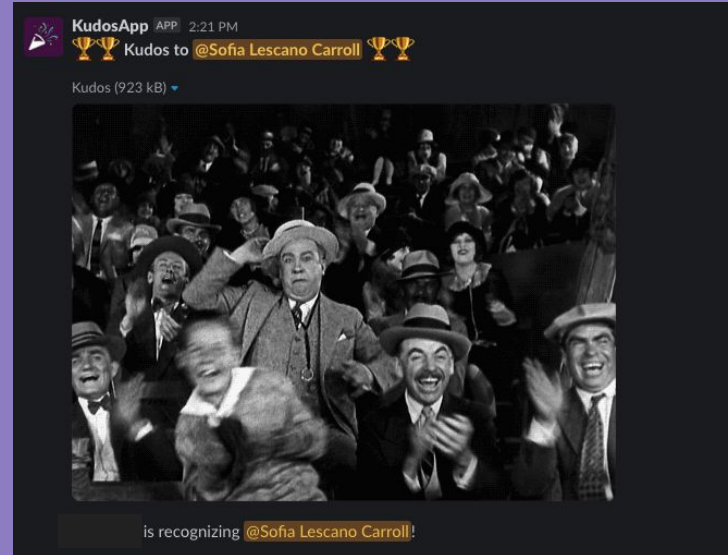
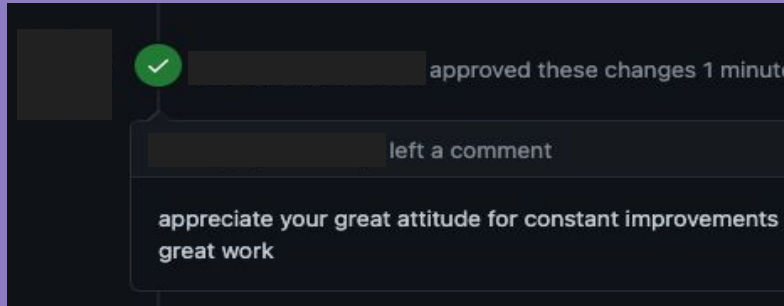
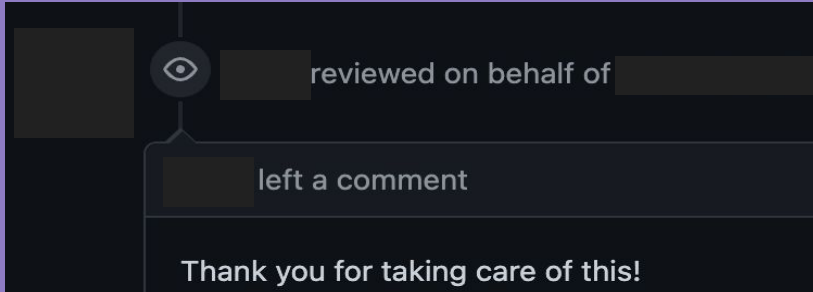
Inspire by example rather than denounce

Not everyone follows the rules



Value the work of the people involved

Not everyone follows the rules



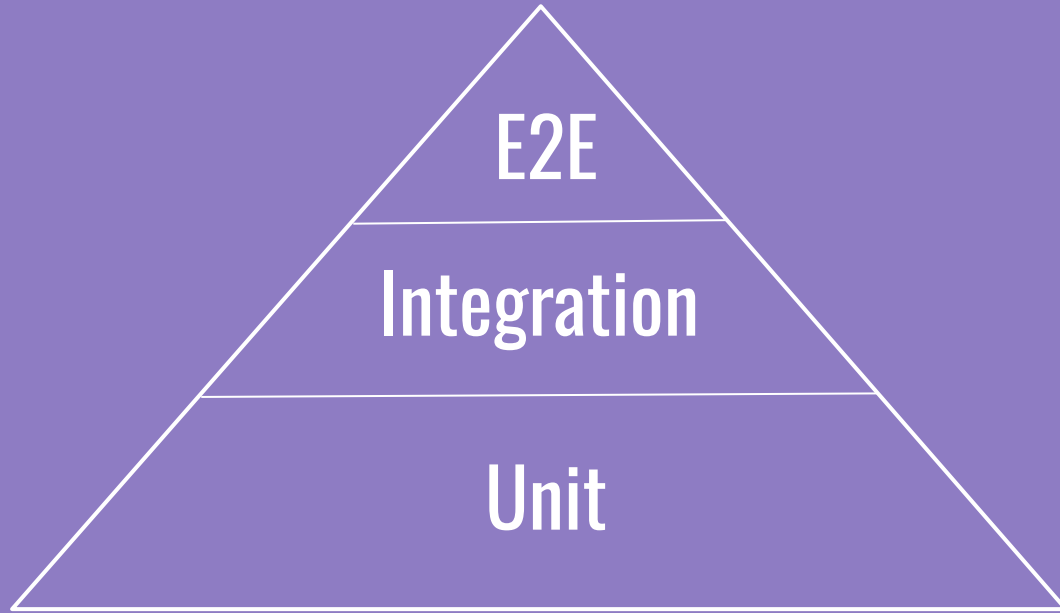
Value the work of the people involved

Instabilities that are difficult to address



Maybe the answer is ultimately to delete the test or test differently.

Instabilities that are difficult to address



Maybe the answer is ultimately to delete the test or test differently.

**When nothing works,
you can at least make
fun of it**

When nothing works: make fun of it

FLAKY TESTS

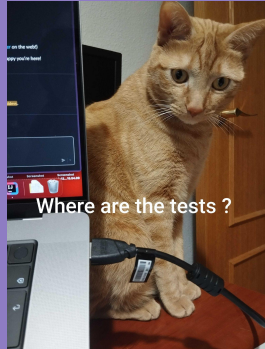


1

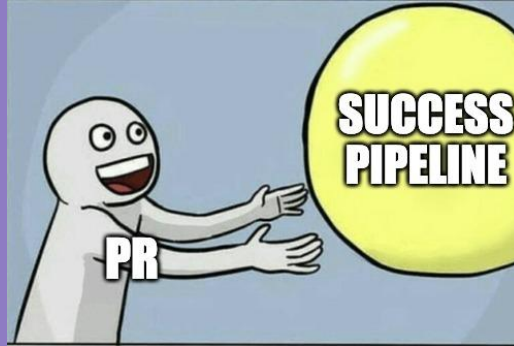
Sofia



0



Where are the tests ?



A picture is worth a thousand words

juicy_fish **Handicon** **Berkahicon**
Freepik
Parzival' 1997 **Iutfix**
Pixel perfect **Iconixar** **Rukanicon**
Thank you !  **Wahyu Adam**
Geotatah **nangicon** **srip**
Monkik **Smashicons** **M Karruly** **mynamepong**

Icons made by the artists mentioned, from www.flaticon.com



Thank you !

Our test instability prevent us from delivering

Sofía LESCANO CARROLL



@SofLesc



#fullRemote #PHP #Laravel #lifeInSpain #startup #doGood #quality