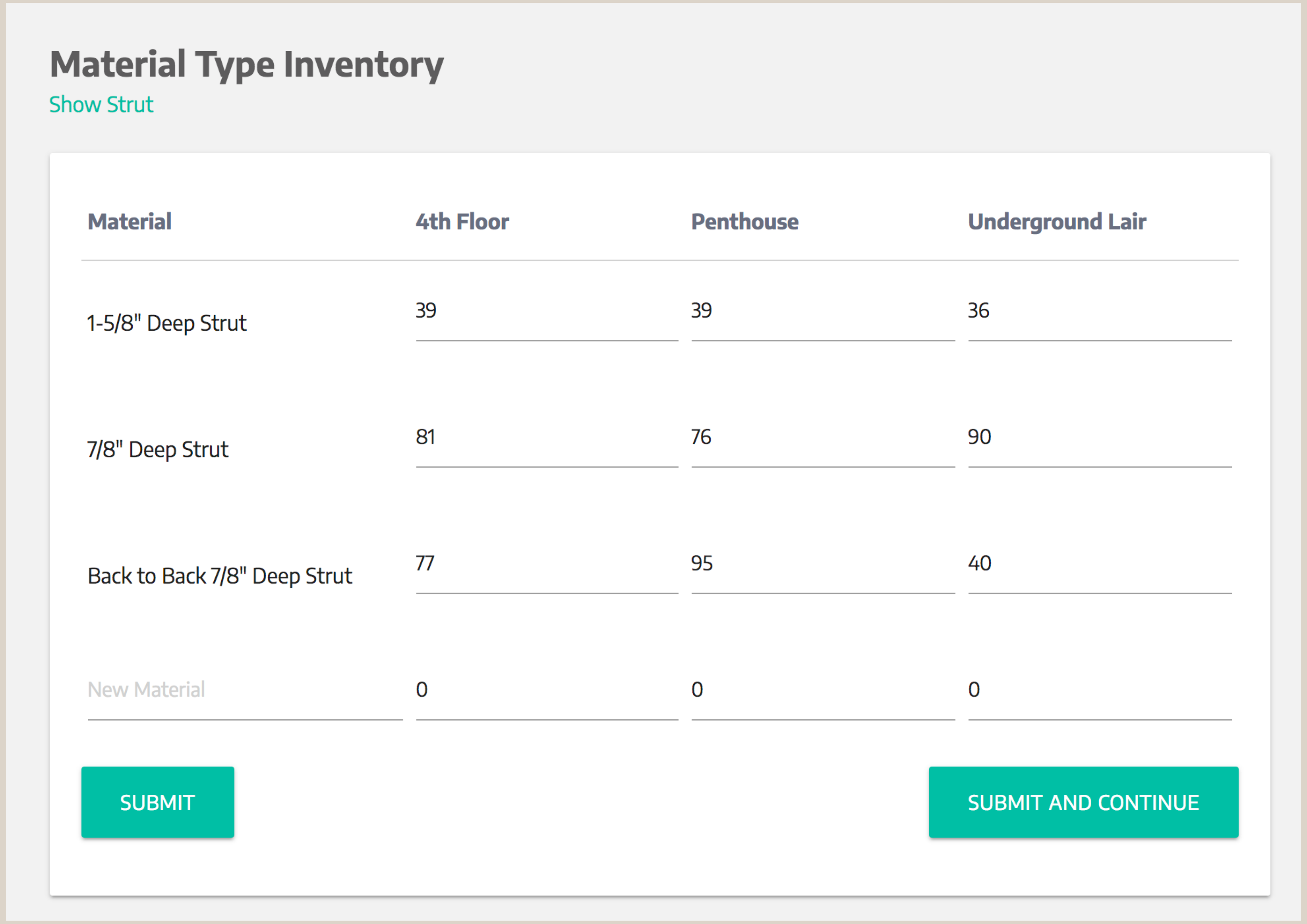
This slide saves me time in the long run even though it took time to create

gifs.com

When I get a

feature request



Material Type Inventory

Show Strut

Material	4th Floor	Penthouse	Underground Lair
1-5/8" Deep Strut	39	39	36
7/8" Deep Strut	81	76	90
Back to Back 7/8" Deep Strut	77	95	40
New Material	0	0	0

SUBMIT

SUBMIT AND CONTINUE

Inventory management software

I need to decide

how to test

I could

Outside-In

TDD

I could just write
integration
tests

I could

not test

I'm making a

decision

Testing will be worth it 💸 ⏰

What does worth it mean?

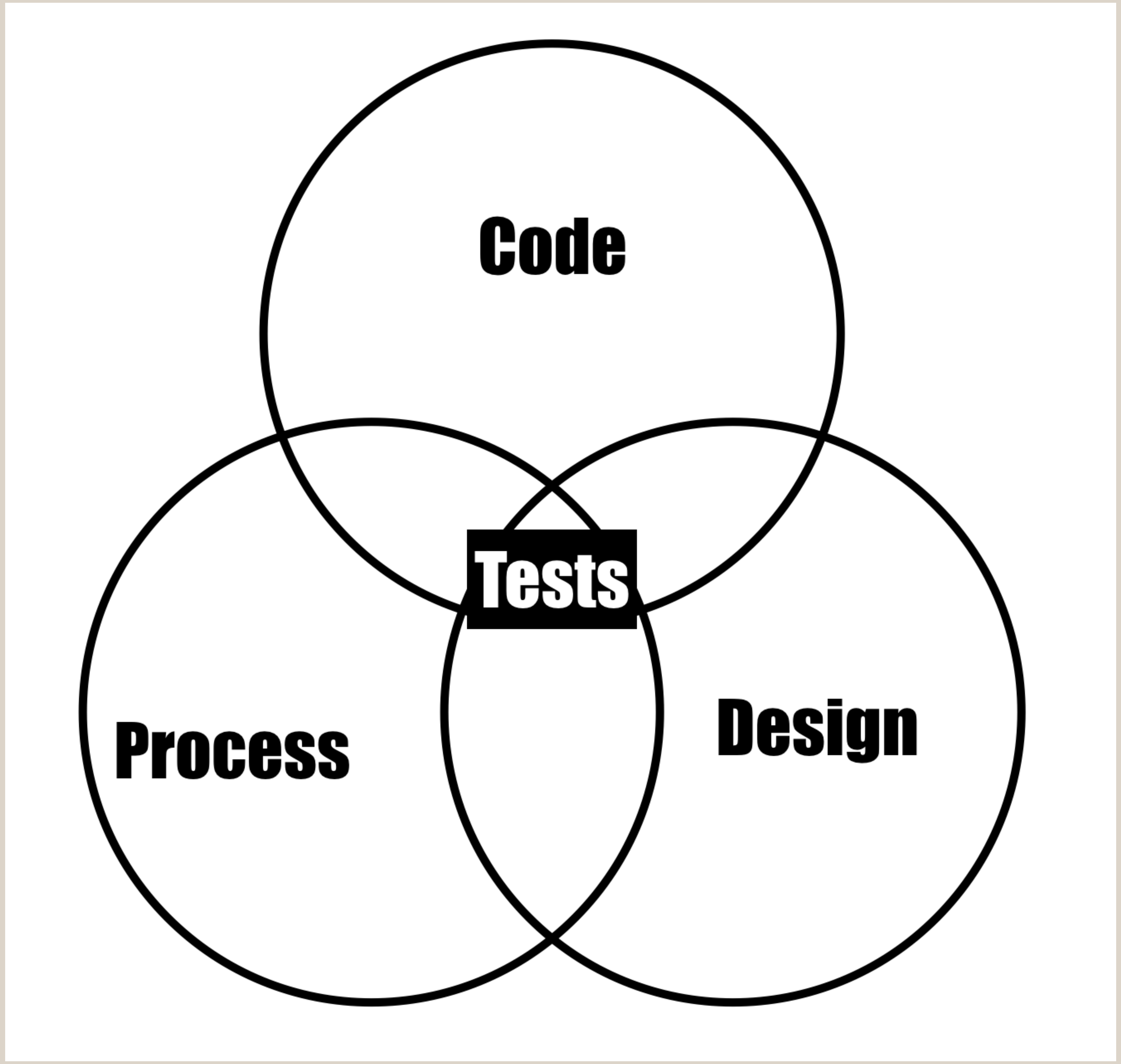
💸 ⏰

How do I know?

High Cost Tests & High Value Tests

Noel Rappin (@noelrap)
https://www.tablexi.com
http://techdoneright.io
http://www.noelrappin.com
http://pragprog.com/book/nrtest3

How can you measure cost and value?



Tests are at the intersection of:

Code
Process
Design

Time

is our metric

How do tests cost time? 💸

You have to 📝

write

the test

The test

runs

A lot

The test needs to be

understood 🤔

The test needs to be

fixed 🔧

How do tests save time? 💾

Writing the test

improves

code design

Running the test

is

faster 🏎️

than manual

testing

The test
validates 🏅
the code

Catches

Bugs

Faster

Cost 📝 🏃 🤔 🔧

vs

Value 💡 🏎️ 🏅 🐞

Dev:

📝 💡 🏎️

Forever:

🏃 🤔 🔧 🏅 🐞

Spoiler alert:

There is no right answer

Strategy

not

Tactics

Some data

End-to-End Integration Capybara

Starts With	Ends With
User Input	HTML Output

Write Time	Run Time
30 mins	0.5 - 3 seconds

Workflow Intermediate Object

Starts	Ends
Params and **workflow.run**	Database changes

Write Time	Run Time
15 mins	0.05 - 0.3 seconds

Unit
One method

Starts	Ends
Call a method	Output of that method

Write Time	Run Time
1-5 mins	0.001 - 0.04 seconds

Type	Specs	Total run time	Avg run time	Write time
System	22	12.72	0.570	~11 hrs
Workflow	40	2.36	0.059	~10 hrs
Unit	119	1.86	0.015	~10 hrs

System tests are 12% of the tests and 75% of the run time

The slowest 4 tests are 40% of the run time

The run times have a wider range

Kind	Min	Max	Variance
Write	1 min	30 min	30x
Run	0.001 sec	3 sec	3000x

Another project

Type	Specs	Total time	Avg time
System	409	579	1.42
Workflow	534	206	0.38
Unit	773	93	0.12
Total	**1716**	**878**	**0.51**

End to end tests are 23% of tests and 66% of run time

What does that suggest?

Balance ⚖️

time spent

As you write similar tests, costs go

down

Short term cost 📝 💡 🏎 is not related to test type

Long term cost

🏃 🤔 🔧 🏅 🐞

is

Long term cost:

runtime failure

In other words: complexity 🤯

Long term savings come from focus

A small fraction of your tests can be the

bulk

of your cost

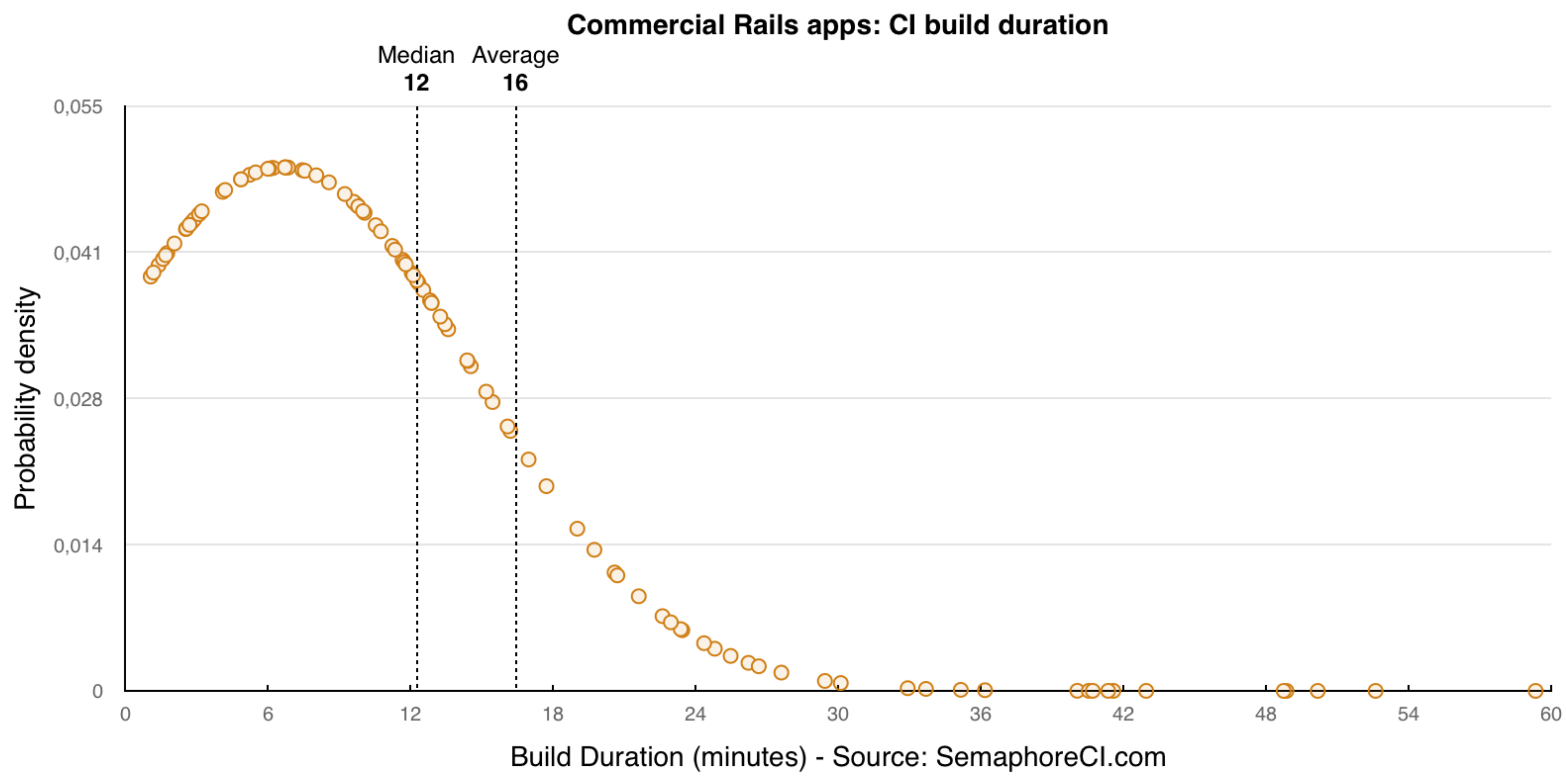
Big payoff in **avoiding** the slowest tests

That's a long way to get to "don't write slow tests"

No individual test causes a slow suite

It's an aggregate set of decisions

Is a slow test suite **inevitable?**



All Rails apps | App size by LOC: up to 5k | 5-10k | 10-20k | 20-40k | over 40k

Test suites get longer as the code gets more complicated.

Only CI runs all the tests... we can throw hardware at that

As long as I can run the tests I'm working on quickly, I'm fine

A short history of the Rails community's thoughts about testing...

1. Testing is great

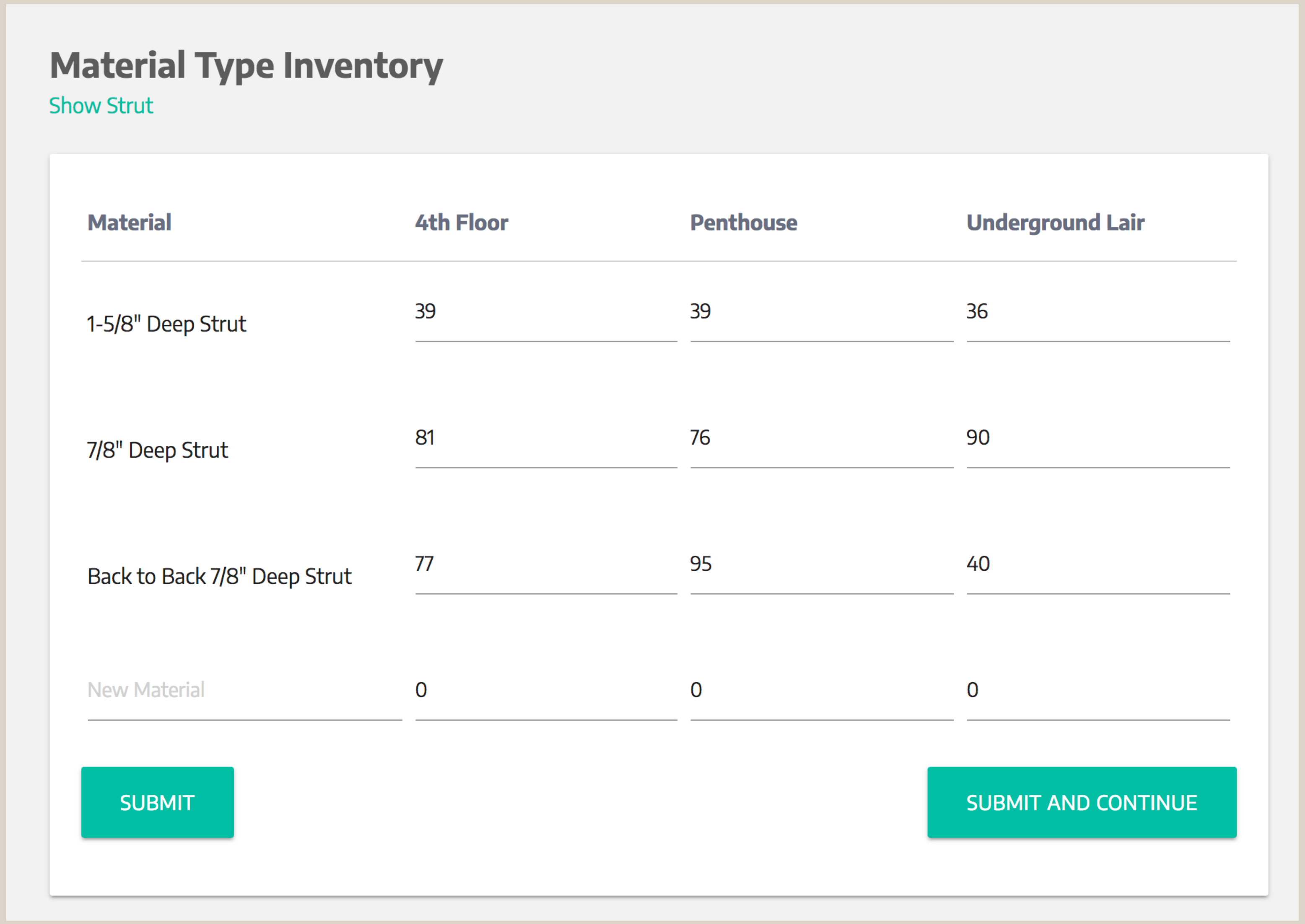
2. Testing is slow

3. Let's try and make testing faster

4. That's hard. Let's throw CI at it.

There's still a cost to a long suite

Giving up causes you to lose the value of tests in improving code

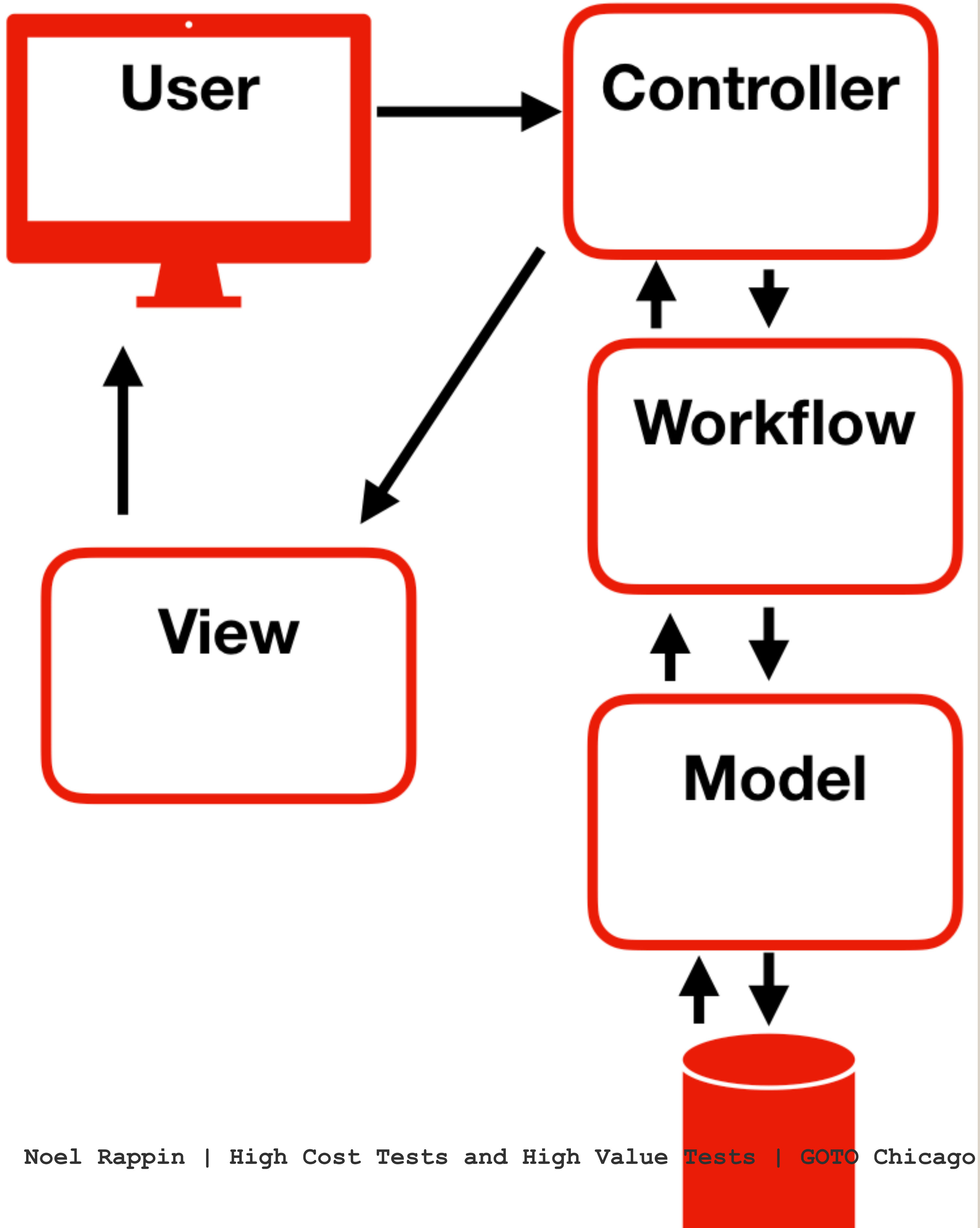


Material Type Inventory

Show Strut

Material	4th Floor	Penthouse	Underground Lair
1-5/8" Deep Strut	39	39	36
7/8" Deep Strut	81	76	90
Back to Back 7/8" Deep Strut	77	95	40
New Material	0	0	0

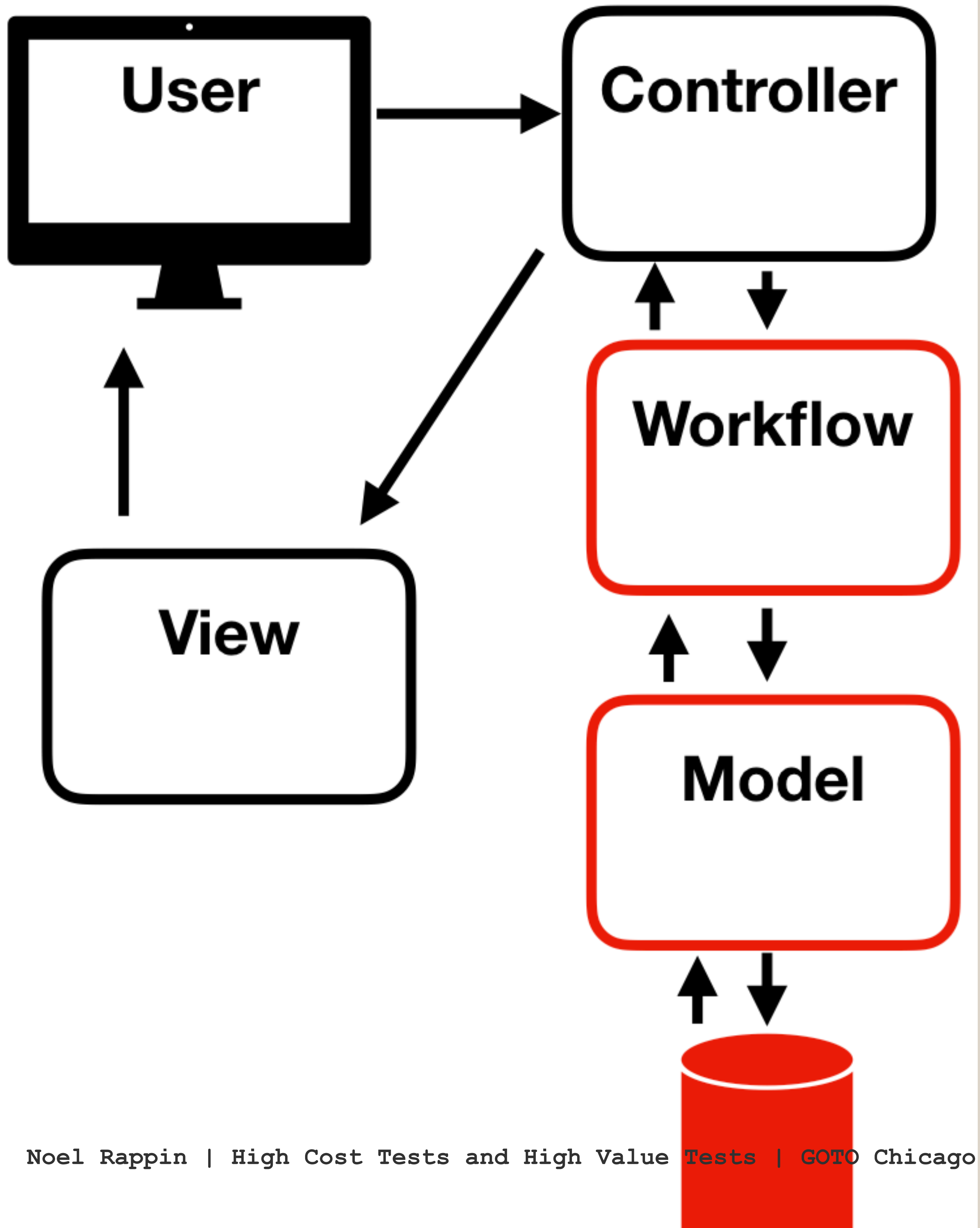
Inventory management software



First test: Capybara integration

Fails on:

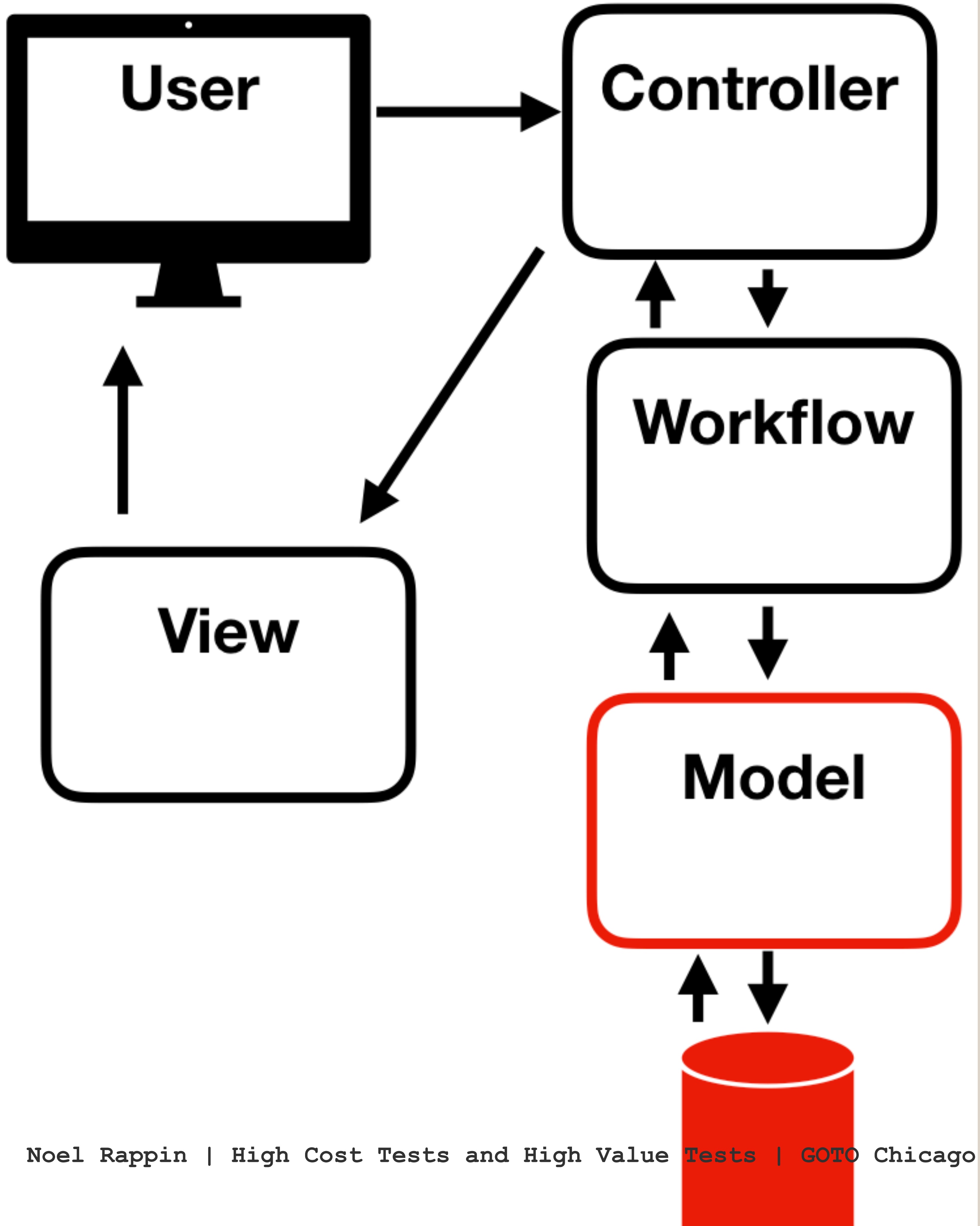
- View failure
- Controller failure
- Handoff from controller to logic failure
- Logic failure
- Database access failure



Next test: workflow object

Fails on:

- Logic failure
- Database access failure



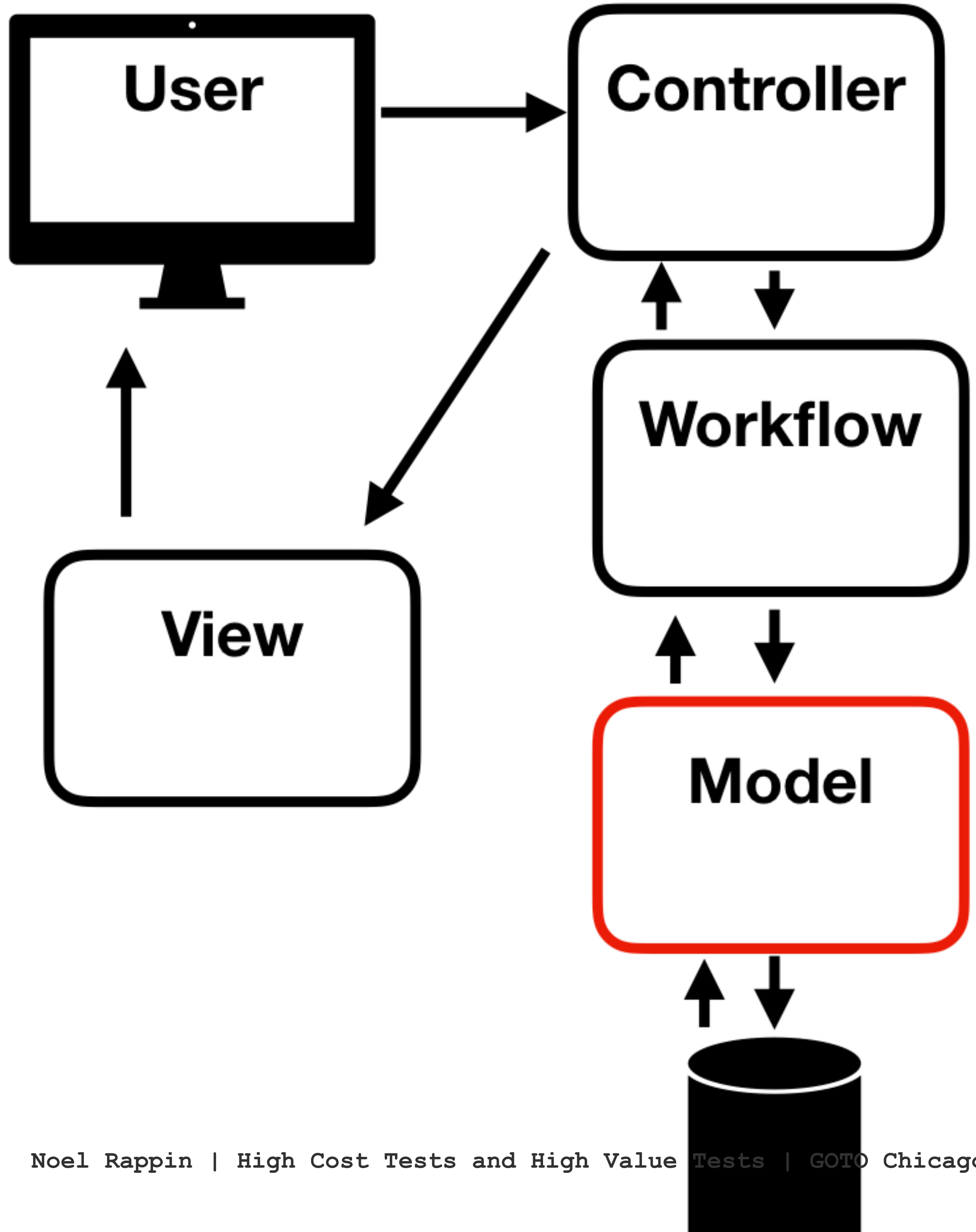
Maybe: Unit test

Fails on

- specific bit of logic
- database access failure

Failure paths: Bad Input "A", -3, ""

System Test
Workflow Test
Unit test

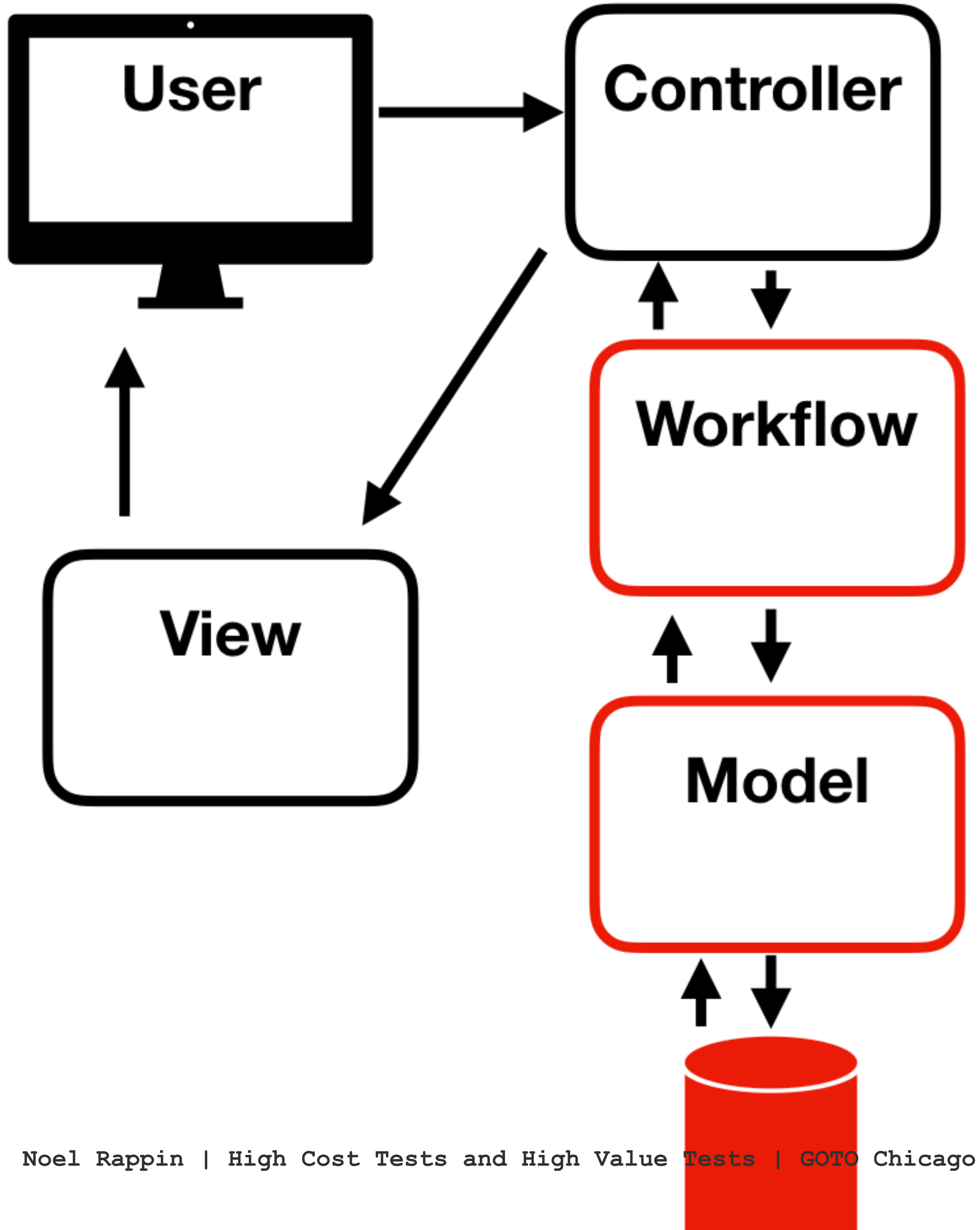


Unit test

A partial test of the workflow
Or a model test
Or a logic object

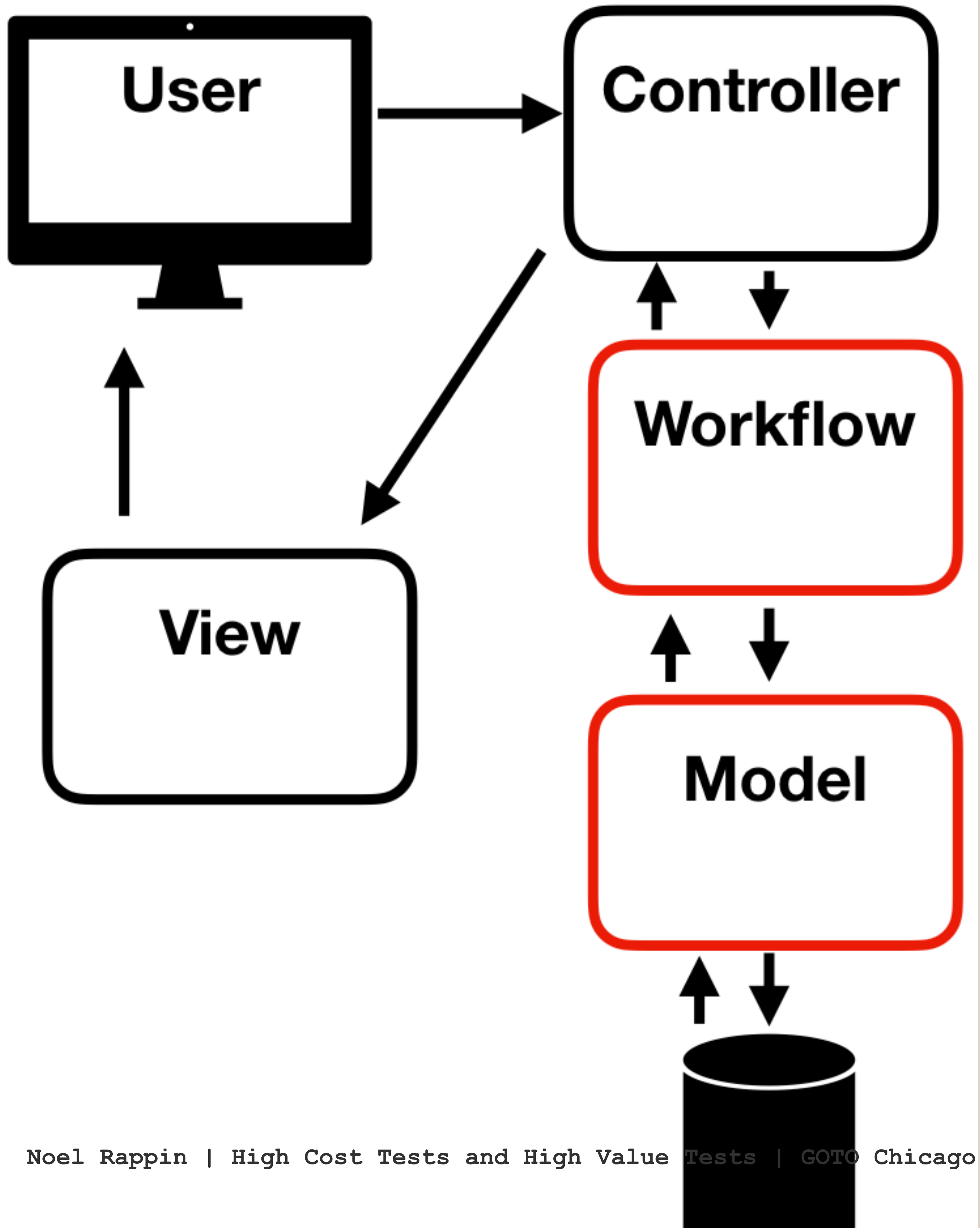
New feature

blank row that can become a new item



Workflow test

New bug a new row that duplicates a name



Not a system test workflow or partial workflow test

What if you don't like unit tests?

"TDD is Dead"

Argument against unit tests:

Unit tests cost

too much 📝

to write

Often true in a legacy context

True if framework doesn't handle tests

You might hear:

Unit tests cause hard to understand 🤔 💡 designs

True if you don't like small units

You might just be writing

bad

unit tests

A lot of Copy/Paste 🍝

A lot of unrelated assertions

Logic change is

far away

from method under test

Capybara is not a

unit test

framework

Lack of unit tests:

Good

legacy

code strategy

At the cost of harder to **diagnose** tests 🔧

Strategies

What will make a test fail?

If it can't fail

uniquely

Do you need it?

Create the minimum amount of objects needed to trigger the failure

Use multiple test failures as an opportunity

Sometimes you can

delete

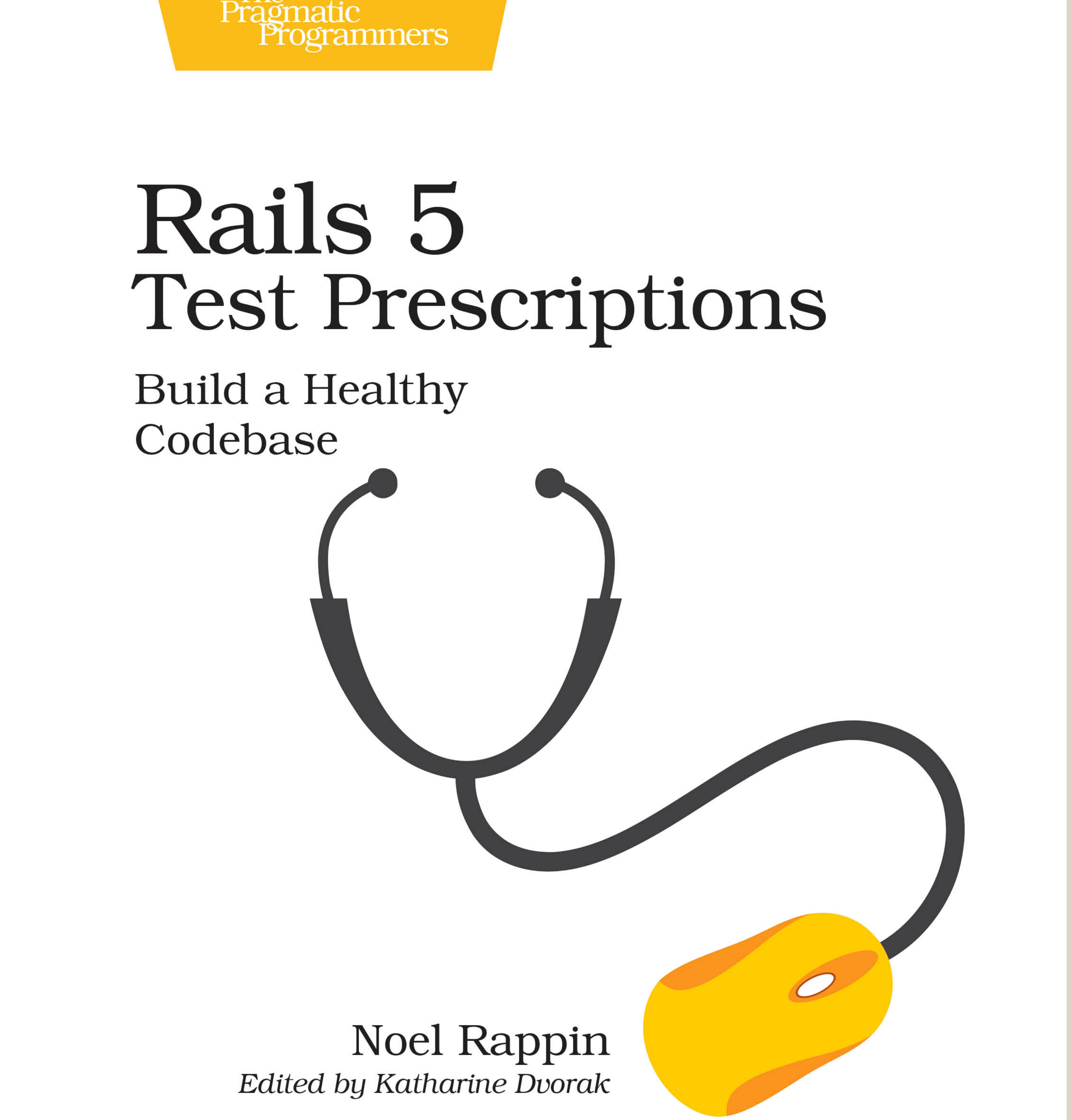
tests

Use integration tests to save development time

Tests have costs

Tests have

value



Pragmatic Programmers

Rails 5 Test Prescriptions

Build a Healthy Codebase

Noel Rappin

Edited by Katharine Dvorak

Noel Rappin (@noelrap)

http://pragprog.com/book/nrtest3
workshops@tablexi.com
http://techdoneright.io/