Vilnius.js Meetup October 2022

Danske Bank

Reverse Enforced Test Driven Development: Making e2e Tests Easy

Vilius Roškus

Chief IT Software Engineer, Danske Bank

Simple

Danske Bank

Software development

can be simple

Agenda

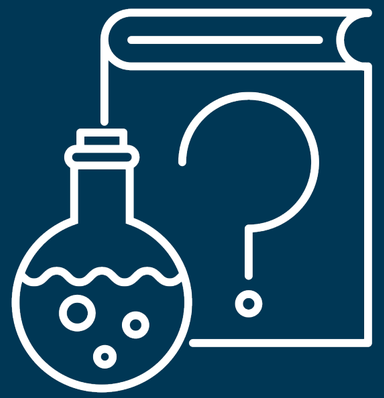
Danske Bank

What to test?

Understanding it easily

How to test?

Getting the maximum from the least effort



How not to forget to test?

Seniors, this one is for you

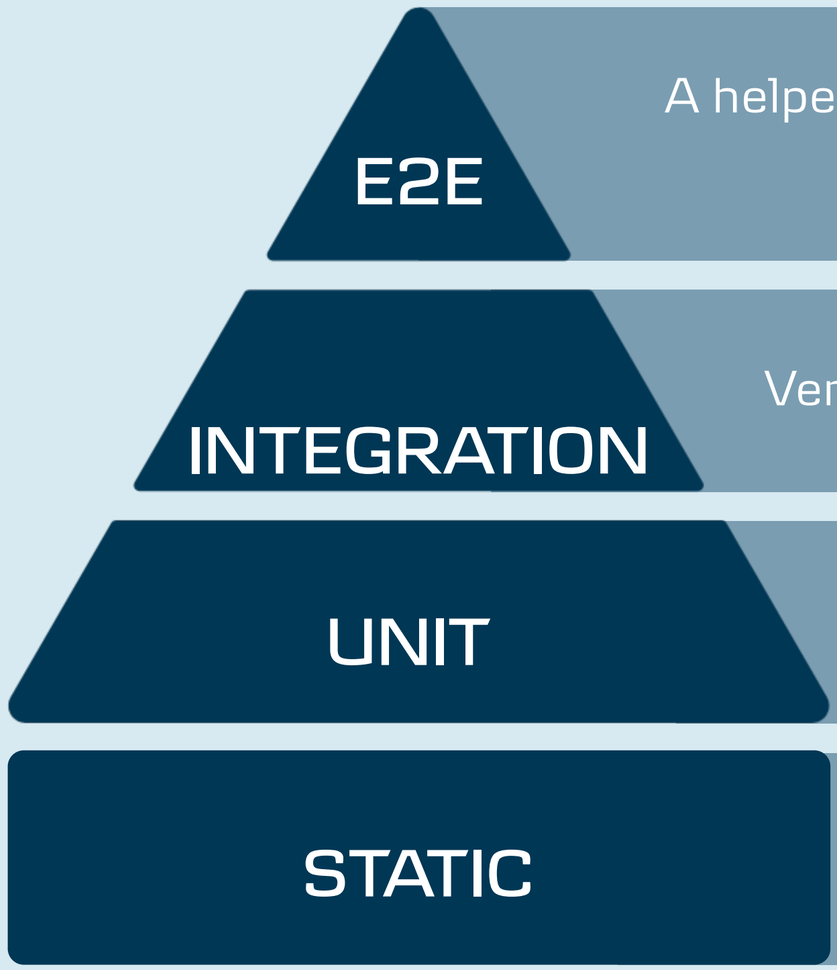
Test types

Danske Bank

Amount of tests needed

Slower, more time to create

Faster, easier to create



A helper robot that behaves like a user to click around the app and verify that it functions correctly.

Verify that several units work together in harmony.

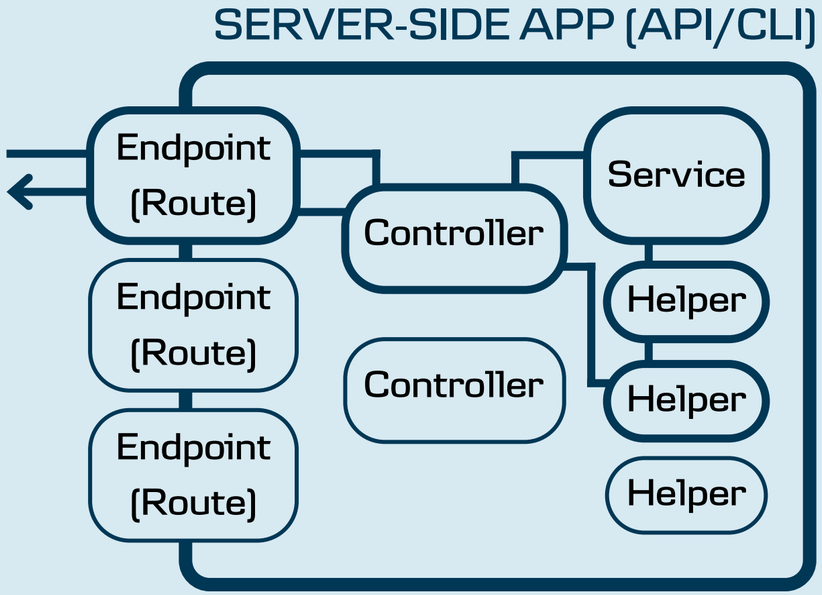
Verify that individual, isolated parts work as expected.

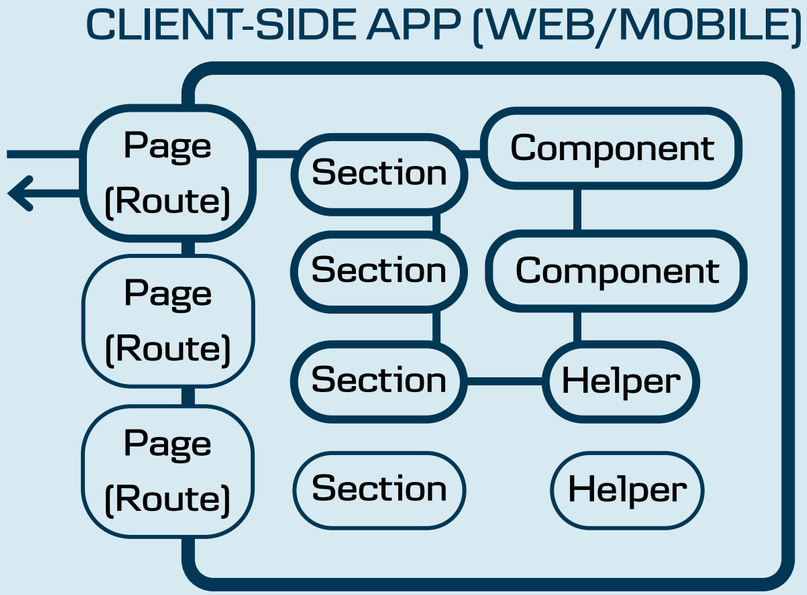
Catch typos and type errors as you write the code.

E2E vs. Unit

E2E

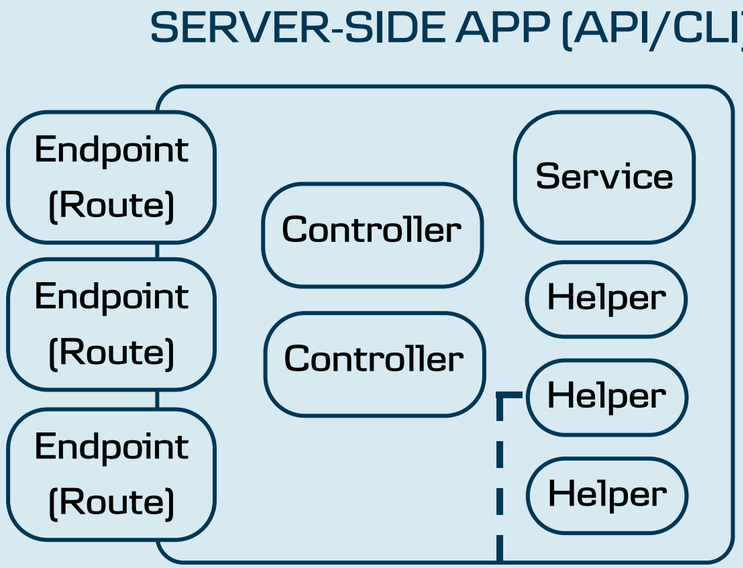
App is launched in test mode and tried as it would behave in real conditions

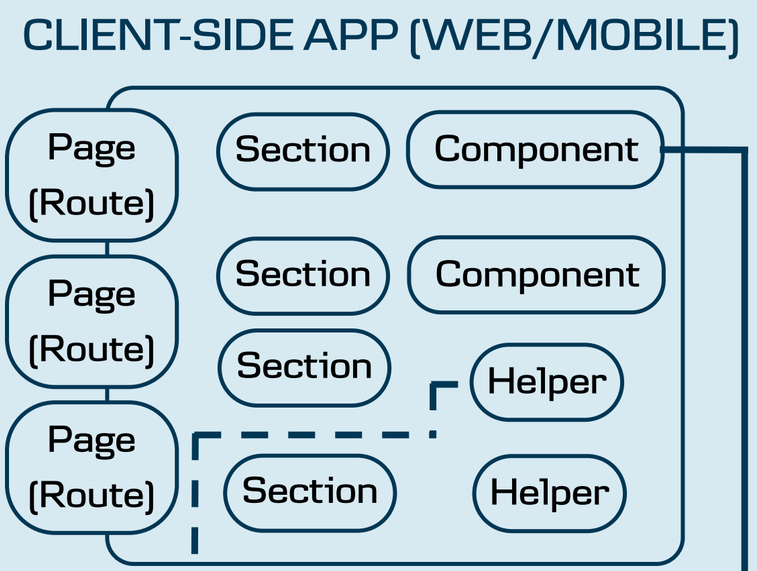


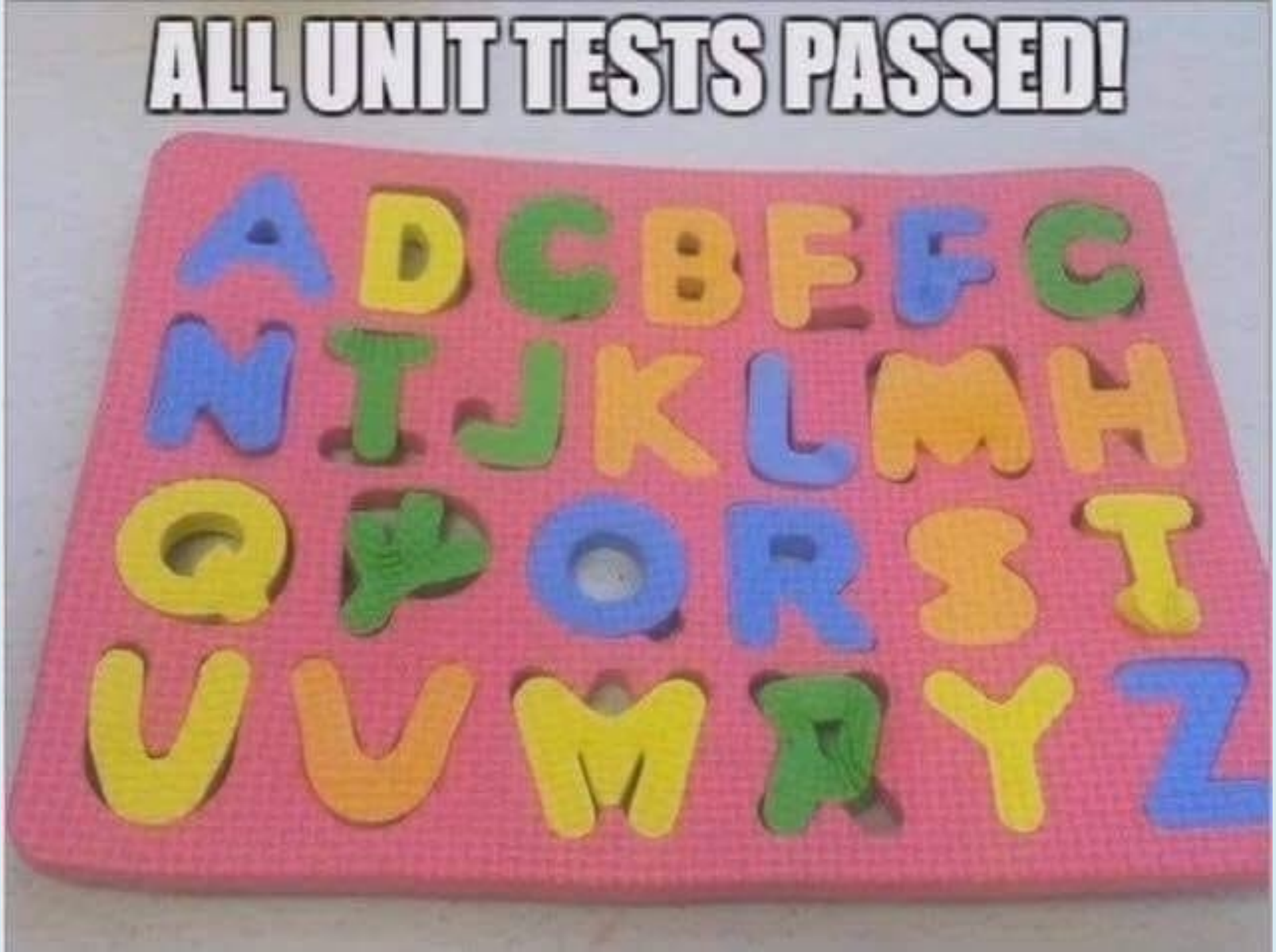


UNIT

A part of application is isolated into testing environment and run independently






ALL UNIT TESTS PASSED!

Test code coverage

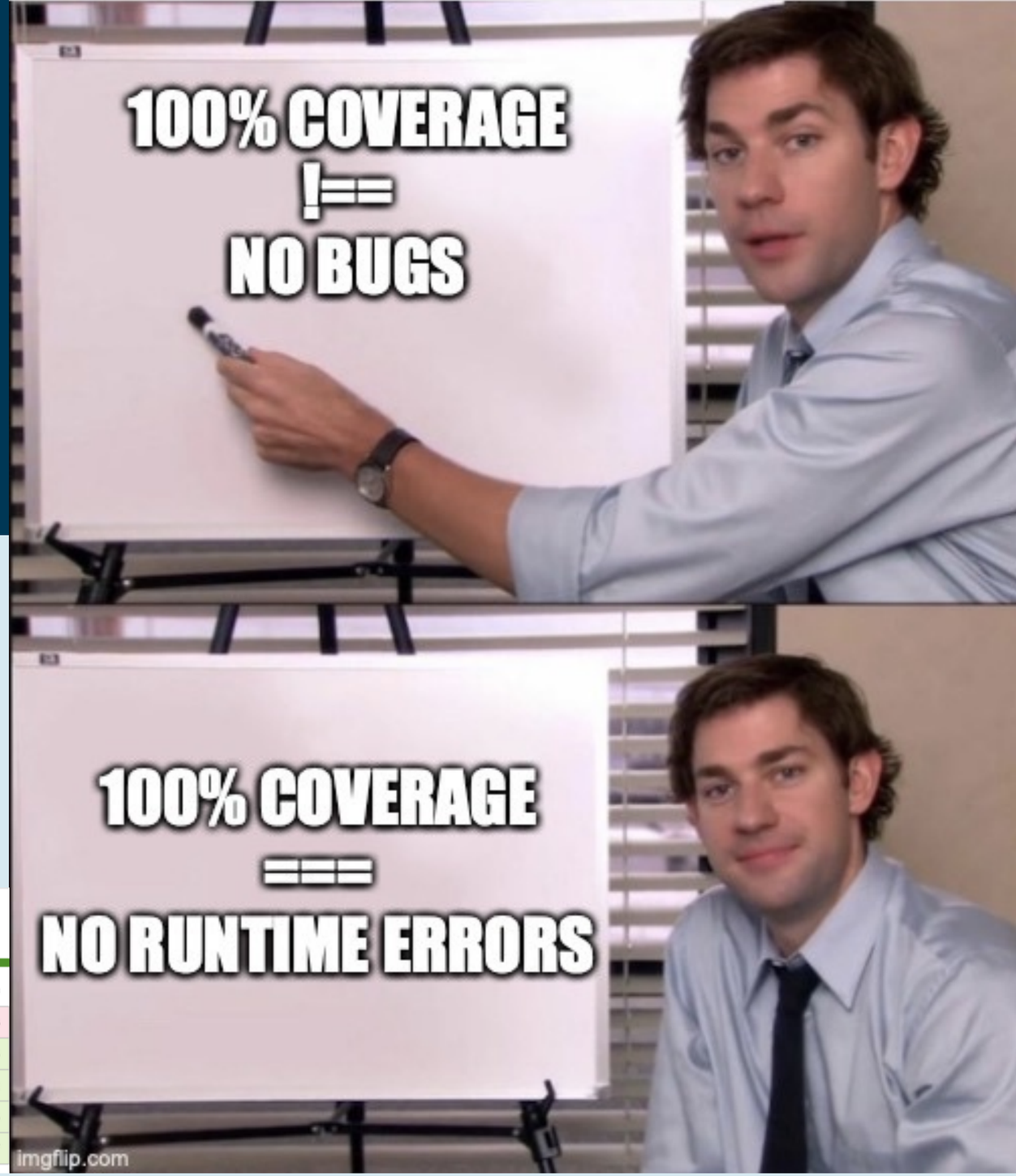
An underrated value that is rarely used beneficially

It can also show the quality of the tests, but the most importantly it shows tested and untested areas of the application

All files

84% Statements 21/25 **100%** Branches 6/6 **100%** Functions 13/13 **84%** Lines 21/25

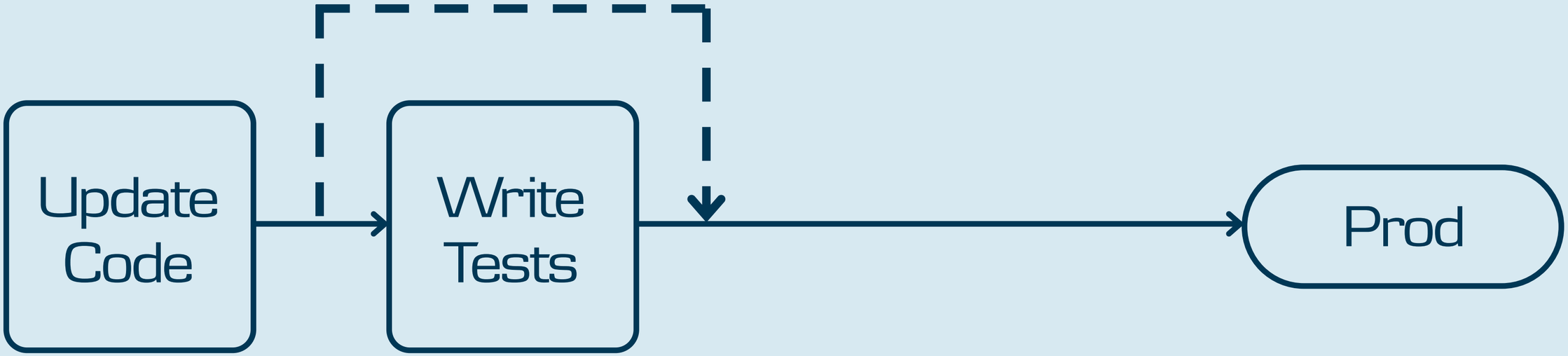
File		Statements		Branches		Functions		Lines	
src		33.33%	2/6	100%	0/0	100%	1/1	33.33%	2/6
src/component/counter		100%	5/5	100%	0/0	100%	5/5	100%	5/5
src/component/header		100%	2/2	100%	0/0	100%	1/1	100%	2/2
src/component/headline		100%	5/5	100%	2/2	100%	2/2	100%	5/5
src/component/login		100%	7/7	100%	4/4	100%	4/4	100%	7/7



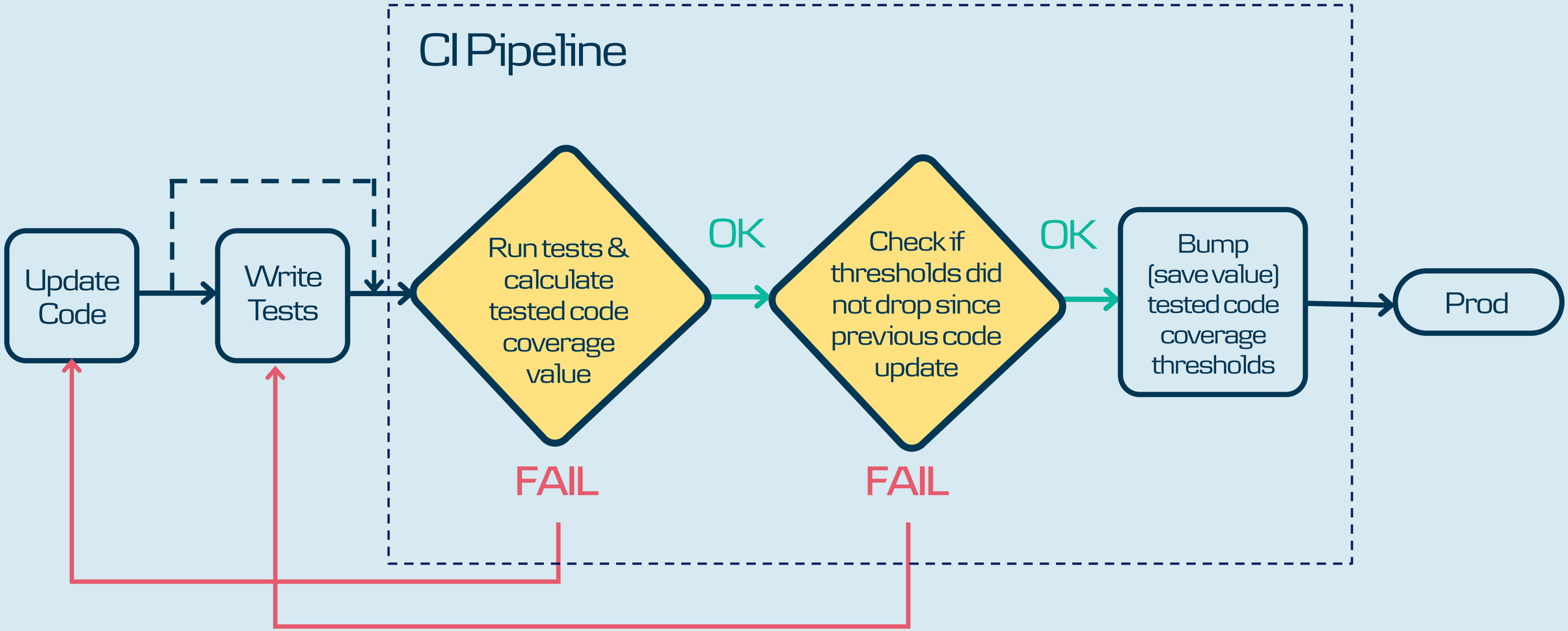
Default Development Flow

Test Driven Development Flow

Reverse Test Driven Development Flow

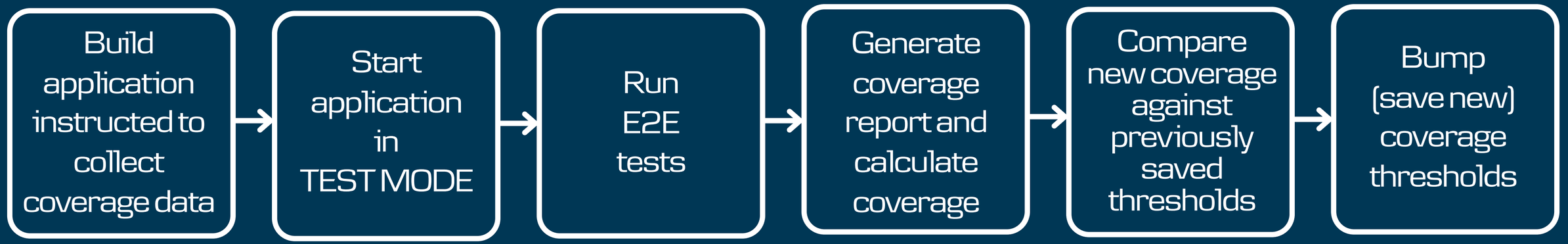


Reverse Enforced Test Driven Development Flow



E2E Test setup

	SERVER-SIDE APP (API / CLI)	CLIENT-SIDE APP (WEB / MOBILE)
Test framework	jest, supertest	Cypress, cra
Code instrumentation	babel-plugin-istanbul (by jest --coverage)	@cypress/instrument-cra (uses babel-plugin-istanbul)
Coverage report generator	nyc (by jest --coverage)	@cypress/code-coverage (uses nyc)
Coverage comparator	jest (by jest --coverage)	@vroskus/compare-coverage-thresholds
Coverage bumper	jest-coverage-thresholds-bumper	jest-coverage-thresholds-bumper



Conclusion

What to test?

The coverage report is a map that helpfully points out things that should be tested.

How to test?

Always try to ensure that your application is compatible for e2e tests, implementing these tests becomes easy task and saves time.

How not to forget to test?

Implement RETDD flow into your CI pipeline and make it mandatory condition for a PR merge or a successful build.

Q&A

Thank you!