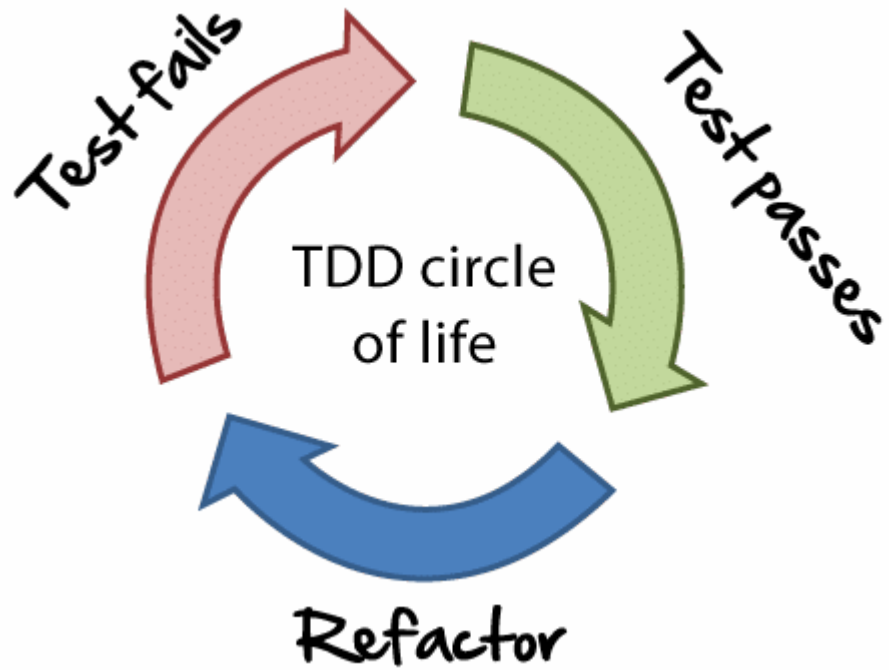
TDD against the odds

Three stories

ThoughtWorks ❤️ Test Driven Development


Test fails
Test passes
TDD circle
of life
Refactor

Let's look at an example

```
data class Sum(private val value: Int) {
    operator fun plus(other: Sum): Sum
}
```

```
data class Sum(private val value: Int) {
    operator fun plus(other: Sum): Sum
}
```

```
Sum(3) + Sum(2) // Sum(5)
```

```
@Test
fun `adds two values`() {
    expectThat(Sum(3) + Sum(2))
        .isEqualTo(Sum(5))
}
```

```
@Test
fun `identity property`() {
    expectThat(Sum(3) + Sum(0))
        .isEqualTo(Sum(3))
}
```

```
@Test
fun `associative property`() {
    expectThat(Sum(3) + Sum(2))
        .isEqualTo(Sum(2) + Sum(3))
}
```

```
data class Sum(private val value: Int) {
    operator fun plus(other: Sum): Sum {
        return Sum(value + other.value)
    }
}
```

Reality is messy

Frontend

Backend

Infrastructure

One case study for each

Frontend

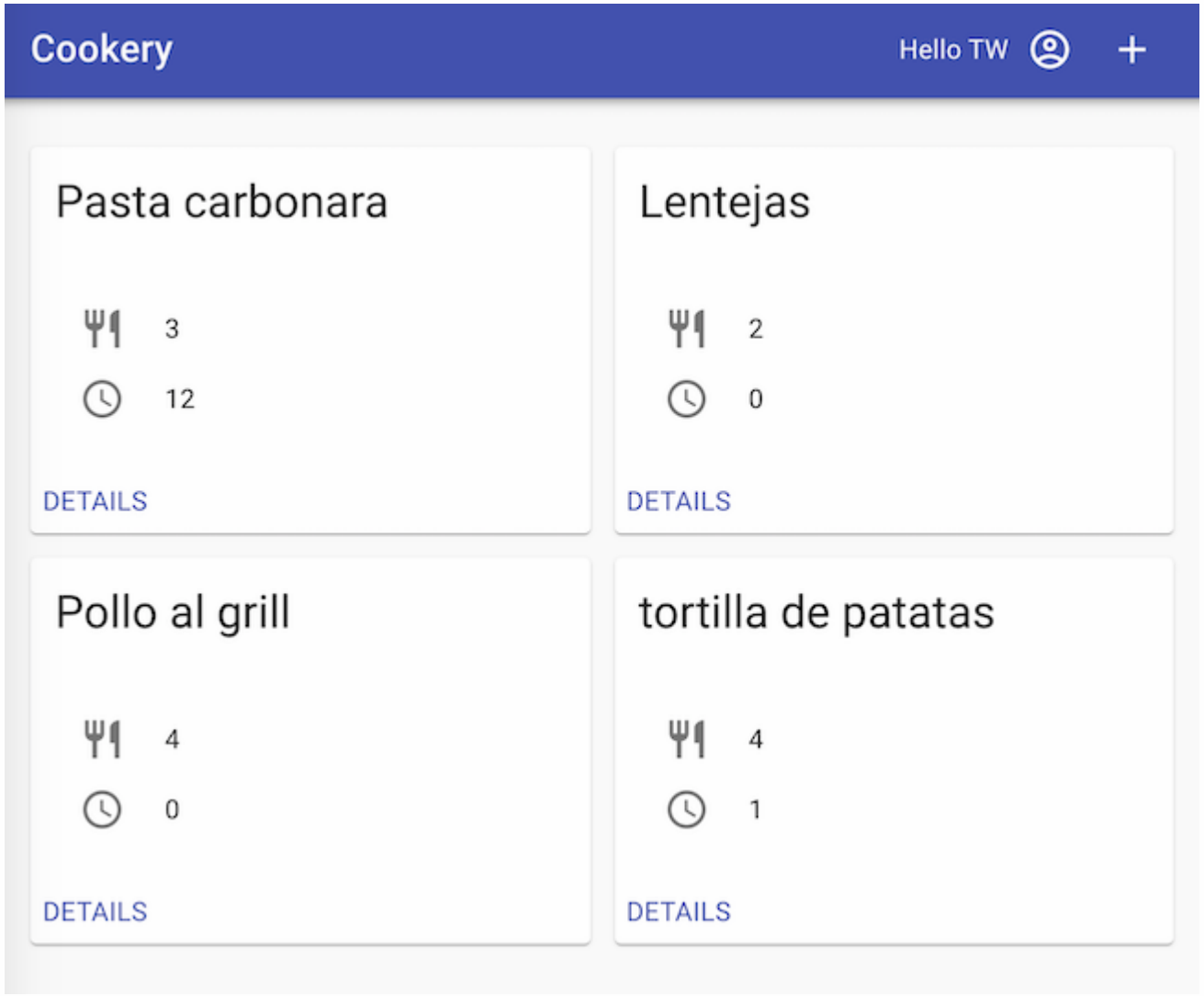
Backend

Infrastructure

Frontend

Frontend = SPA

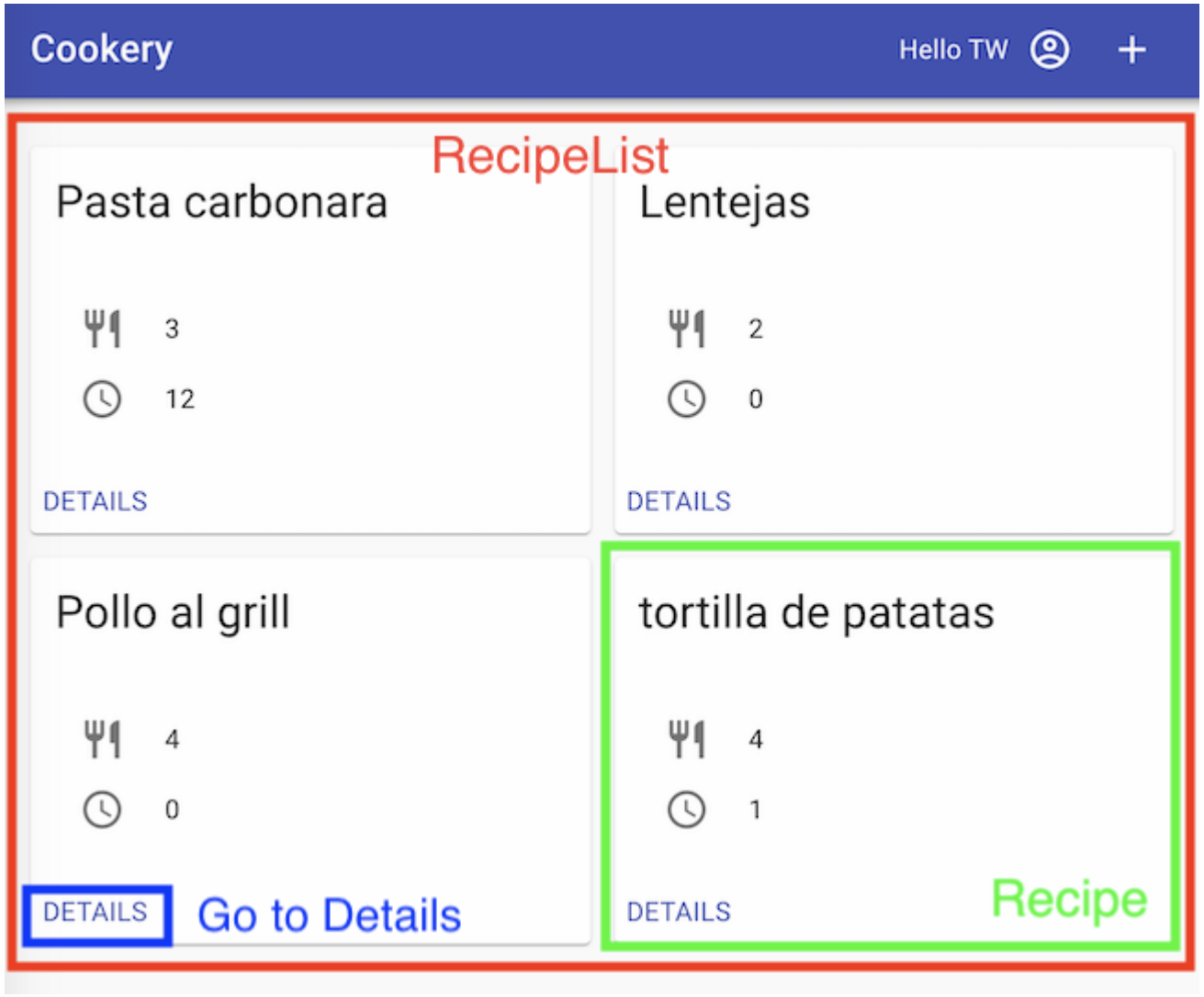
These days


Cookery
Hello TW
Pasta carbonara
3
12
DETAILS
Lentejas
2
0
DETAILS
Pollo al grill
4
0
DETAILS
tortilla de patatas
4
1
DETAILS

How do you do TDD in React?


Jest


ENZYME


Cookery
Hello TW
RecipeList
Pasta carbonara
3
12
DETAILS
Lentejas
2
0
DETAILS
Pollo al grill
4
0
DETAILS
Go to Details
tortilla de patatas
4
1
DETAILS
Recipe

```
it('renders a recipe list', () => {
  const wrapper = shallow(<RecipeList {...props} />)
})
```

```
it('renders a recipe list', () => {
  const wrapper = shallow(<RecipeList {...props} />)

  expect(wrapper.find('Recipe').props().toEqual({
    title: 'Pasta Carbonara'
  }))
})
```

This is not great

kentcdodds.com/blog/why-i-never-use-shallow-rendering

React Testing Library

Simple and complete React DOM testing utilities that encourage good testing practices.

Read The Docs | Edit the docs

> The more your tests resemble the way your software is used, the more confidence they can give you.

```
it('renders a recipe list', async () => {
  const { getByTestId, getByText, getAllByText } = fullRender(<App />, {
    route: '/recipes',
  })
})
```

```
it('renders a recipe list', async () => {
  const { getByTestId, getByText, getAllByText } = fullRender(<App />, {
    route: '/recipes',
  })
})
```

```
jest.mock('recipe-list/recipeList.service')
jest.mock('recipe-details/recipeDetails.service')
```

```
it('renders a recipe list', async () => {
  const { getByTestId, getByText, getAllByText } = fullRender(<App />, {
    route: '/recipes',
  })

  // Navigate to list of recipes
  await waitForElement(() => getByTestId('recipe-list'))
})
```

```
it('renders a recipe list', async () => {
  const { getByTestId, getByText, getAllByText } = fullRender(<App />, {
    route: '/recipes',
  })

  // Navigate to list of recipes
  await waitForElement(() => getByTestId('recipe-list'))

  // Check recipe
  await waitForElement(() => getByText('Pasta Carbonara'))
})
```

```
it('renders a recipe list', async () => {
  const { getByTestId, getByText, getAllByText } = fullRender(<App />, {
    route: '/recipes',
  })

  // Navigate to list of recipes
  await waitForElement(() => getByTestId('recipe-list'))

  // Check recipe + Navigate to it
  await waitForElement(() => getByText('Pasta Carbonara'))
  userEvent.click(getAllByText('DETAILS')[0])
  await waitForElement(() => getByText('egg'))
})
```

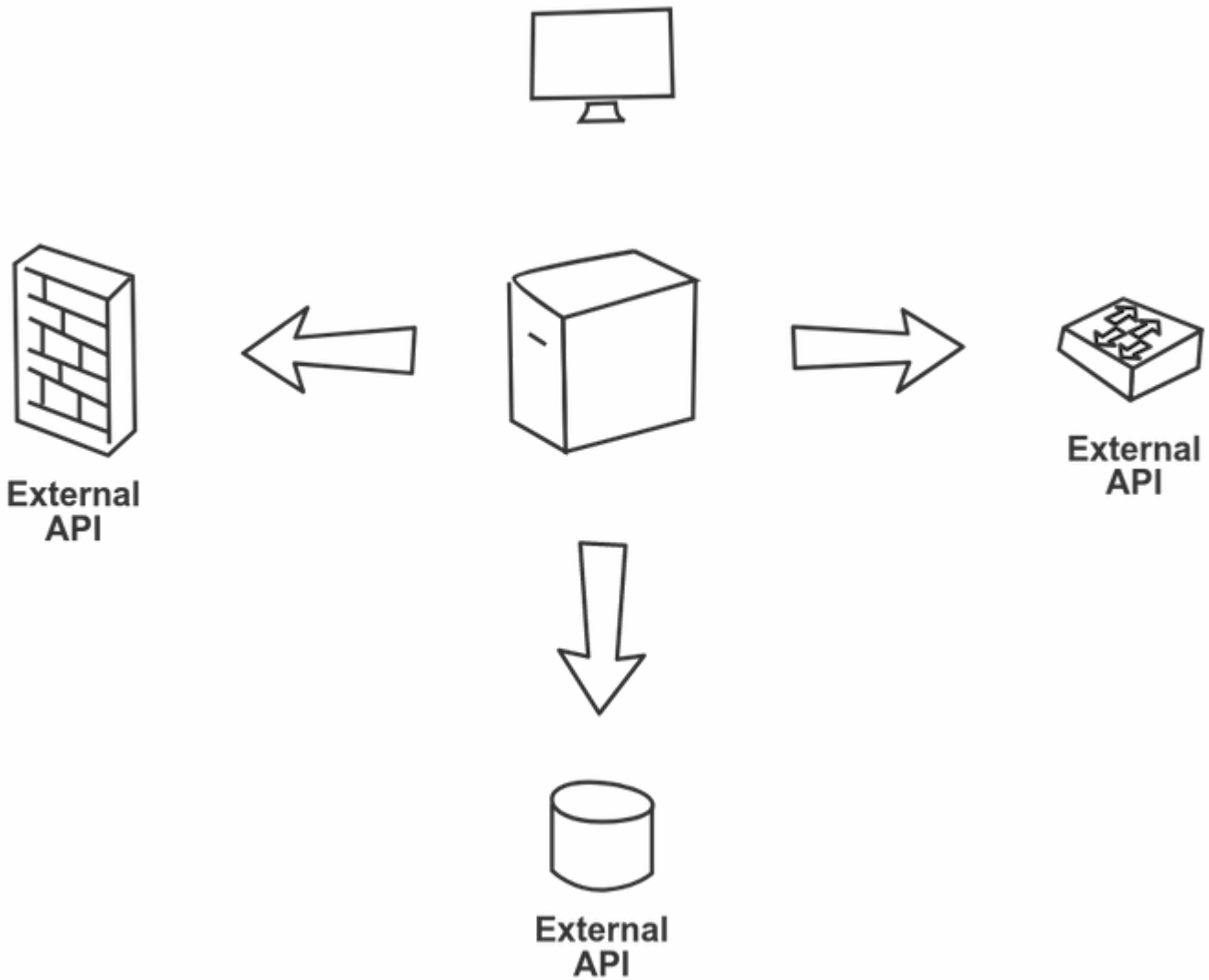
```
it('renders a recipe list', async () => {
  const { getByTestId, getByText, getAllByText } = fullRender(<App />, {
    route: '/recipes',
  })

  // Navigate to list of recipes
  await waitForElement(() => getByTestId('recipe-list'))

  // Check recipe + Navigate to it
  await waitForElement(() => getByText('Pasta Carbonara'))
  userEvent.click(getAllByText('DETAILS')[0])
  await waitForElement(() => getByText('egg'))

  // Go back
  userEvent.click(getByText('Back'))
  await waitForElement(() => getByTestId('recipe-list'))
})
```

```
PASS  src/App.test.tsx
  App
    ✓ renders without crashing (111ms)
    ✓ renders new recipe form (107ms)
    ✓ renders a recipe list (148ms)
    ✓ renders recipe details (62ms)
    ✓ redirects to recipes list (39ms)

Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        5.534s, estimated 10s
Ran all test suites related to changed files.

Watch Usage: Press w to show more.
```

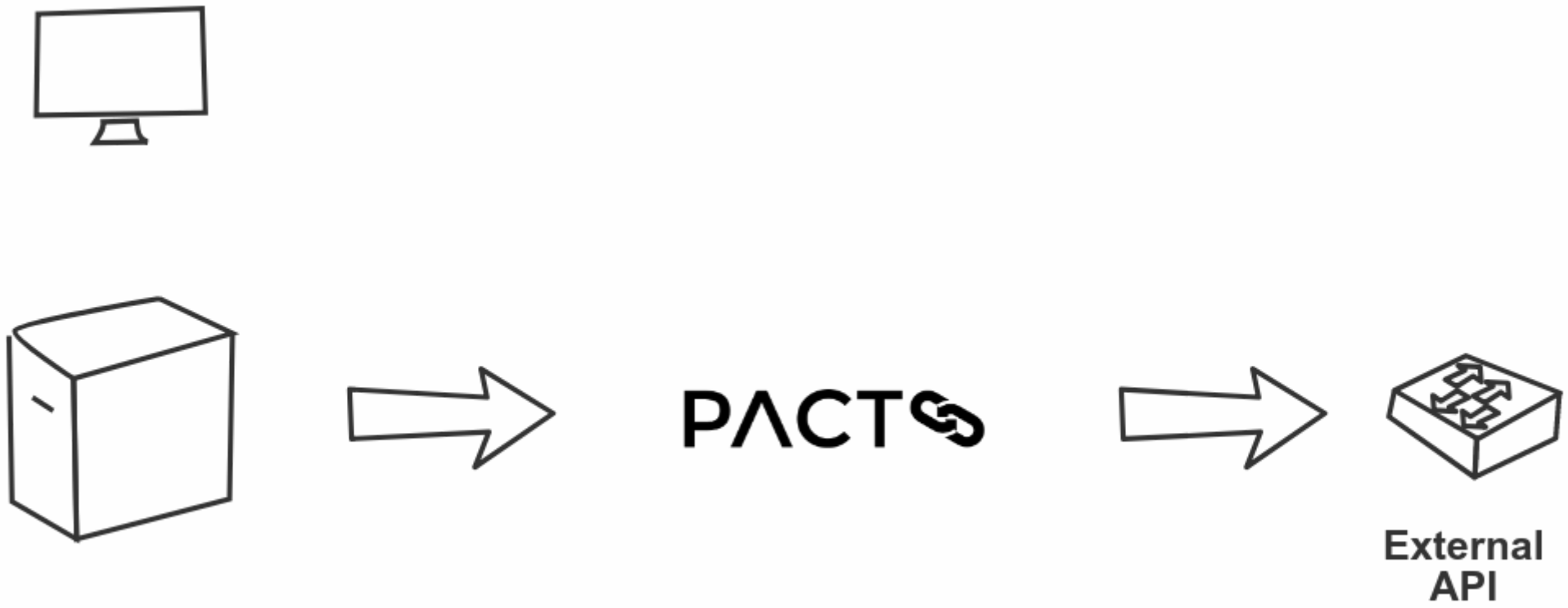
Closer to integration tests than to unit tests

But more meaningful results

Backend


External
API
External
API
External
API

Consumer Driven Contract Testing

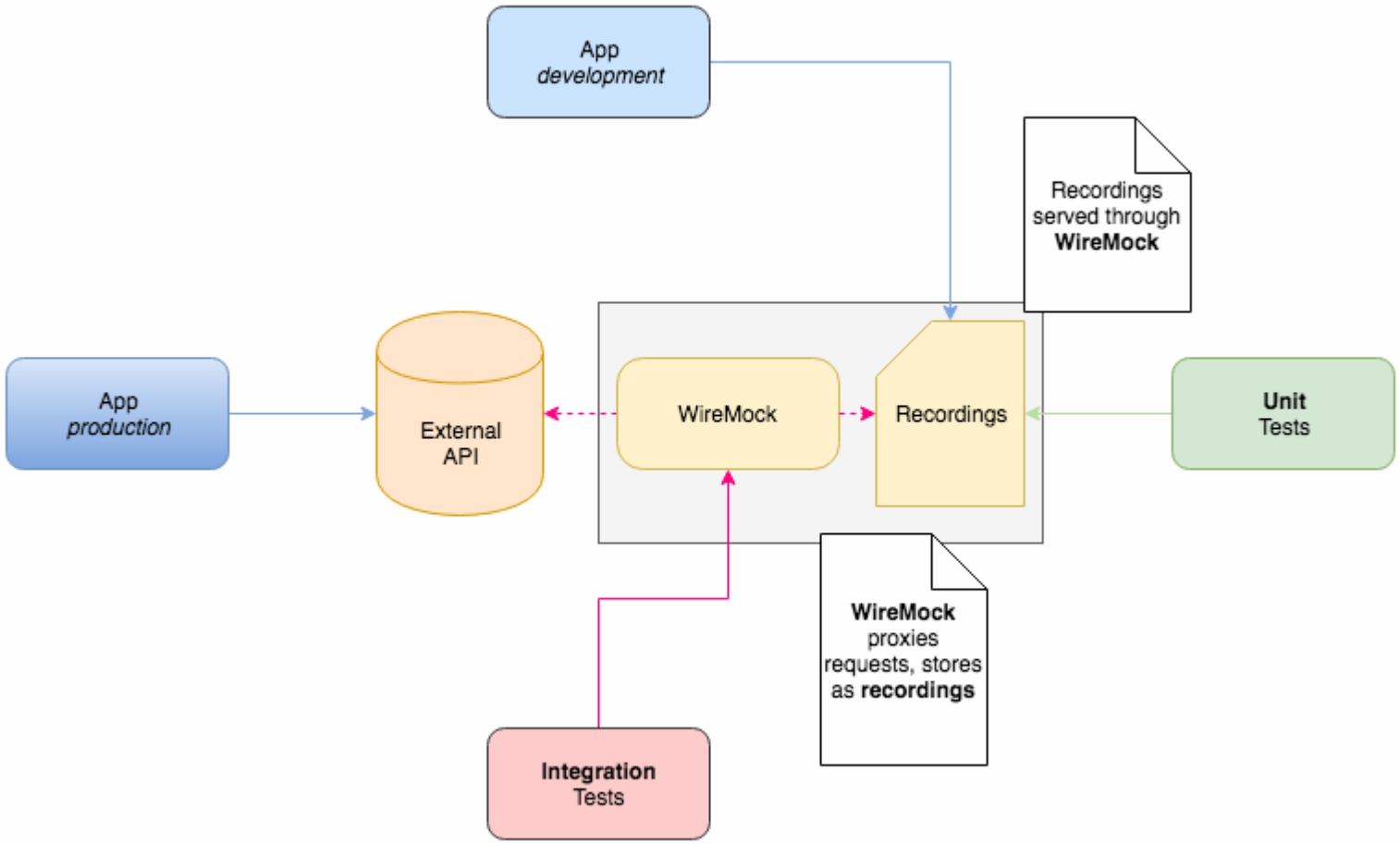


Only works if both parties can fulfill the contract as part of CI/CD

Alternative

Recordings


WIREMOCK


App
development
Recordings
served through
WireMock
App
production
External
API
WireMock
Recordings
Unit
Tests
WireMock
proxies
requests, stores
as recordings
Integration
Tests

Workflow

Write basic API client that makes a call

```
@Component
class JsonPlaceholder {
    @Autowired
    lateinit var template: RestTemplate

    fun todos(): JsonNode {
        val response = template.exchange(
                "/todos",
                HttpMethod.GET,
                null,
                JsonNode::class.java)
        return response.body
    }
}
```

Write integration test that calls the real endpoint and stores the result

```
@Category(IntegrationTest::class)
@RunWith(SpringRunner::class)
@SpringBootTest
class JsonPlaceholderIntegrationTest : RecordingTest() {
    public override fun recordingServerUrl(): String {
        return "https://jsonplaceholder.typicode.com"
    }

    @Autowired
    lateinit var subject: JsonPlaceholder

    @Test
    fun todos() {
        subject.todos()
    }
}
```

Use recording in unit test and refine call

```
@RunWith(SpringRunner::class)
@SpringBootTest
class JsonPlaceholderTest : RecordedTest() {
    @Autowired
    lateinit var subject: JsonPlaceholder

    @Test
    fun todos() {
        expectThat(subject.todos())
            .isNotEmpty()
    }
}
```

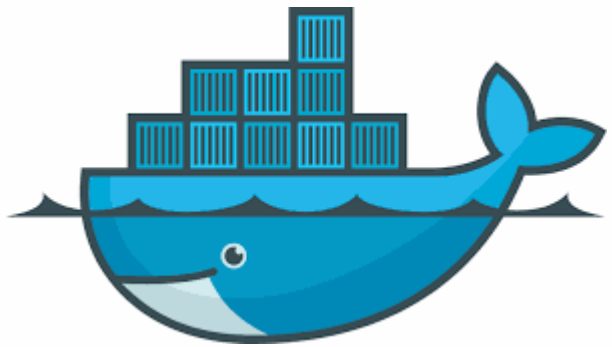
```
usage: ./go <goal>

    goal:

    test-unit                   -- Run unit tests
    test-integration            -- Run integration tests

    refresh-recordings          -- Recreate recordings
```

hceris.com/recording-apis-with-wiremock

Infrastructure

For us, infrastructure mostly means cloud infrastructure provisioned with code

That, however, is a very broad topic



TDD for containers

serverspec.org

Basic setup

```
require 'serverspec'
require 'docker'
require 'rspec/wait'

set :backend, :docker
set :docker_image, 'example-openjdk'

RSpec.configure do |config|
  config.wait_timeout = 60 # seconds
end
```

OS Version

```
describe file('/etc/alpine-release') do
  its(:content) { is_expected.to match(/3.8/) }
end
```

Java Version

```
describe command('java -version') do
  its(:stderr) { is_expected.to match(/1.8/) }
end
```

```
FROM openjdk:8-jre-alpine3.8
```

JAR

```
describe file('gs-rest-service.jar') do
  it { is_expected.to be_file }
end
```

```
FROM openjdk:8-jre-alpine3.8

WORKDIR /app
ENV VERSION="0.1.0"

COPY build/libs/gs-rest-service-${VERSION}.jar gs-rest-service.jar
```

This is not just about static checks

Runtime checks are also possible

App is running

```
describe process('java') do
  it { is_expected.to be_running }
  its(:args) { is_expected.to contain('gs-rest-service.jar') }
end
```

Bound to right port

```
describe 'listens to correct port' do
  it { wait_for(port(8080)).to be_listening.with('tcp') }
end
```

```
FROM openjdk:8-jre-alpine3.8

WORKDIR /app
EXPOSE 8080
ENV VERSION="0.1.0"

COPY build/libs/gs-rest-service-${VERSION}.jar gs-rest-service.jar

CMD ["java", "-jar", "gs-rest-service.jar"]
```

Not running under root

```
describe process('java') do
  its(:user) { is_expected.to eq('runner') }
end
```

```
FROM openjdk:8-jre-alpine3.8

WORKDIR /app
EXPOSE 8080
ENV VERSION="0.1.0"

COPY build/libs/gs-rest-service-${VERSION}.jar gs-rest-service.jar

RUN adduser -D runner
USER runner

CMD ["java", "-jar", "gs-rest-service.jar"]
```

```
rspec spec/container_spec.rb
Randomized with seed 61858
.......

Top 7 slowest examples (12.86 seconds, 99.9% of total time):
  Application Container java listens to correct port should be listening
    7.82 seconds ./spec/container_spec.rb:20
  Application Container java Process "java" should be running
    4.14 seconds ./spec/container_spec.rb:14
  Application Container java Command "java -version" stderr should match
    0.3948 seconds ./spec/container_spec.rb:10
  Application Container java Process "java" args should contain "gs-rest
    0.15328 seconds ./spec/container_spec.rb:15
  (more output ...)

Finished in 12.87 seconds (files took 1.69 seconds to load)
7 examples, 0 failures
```

github.com/sirech/talks/blob/master/2019-01-tw-tdd_containers.pdf

Conclusion

TDD can be used in situations you didn't think about

Get creative

Don't get stuck at the definition of a *Unit Test*

Mario Fernandez

ThoughtWorks