



# Командное тестирование

 @tolkv

 @lavcraft



@- 

@esharova 





не тестировать всем одно и то же по несколько раз



сделать так, чтобы тесты увеличивали обзорность вашей системы



сделать так, чтобы тесты увеличивали обзорность вашей системы упростить поддержку кода тестов



избавиться от лока на конкретные технологии в тестировании.

# Командное тестирование

A full-body image of Darth Vader from Star Wars, standing on a balcony at night. He is wearing his iconic black armor and cape. His right hand is raised in a gesture. The background shows a city skyline with many lit windows, creating a bokeh effect. In the foreground, there are some cylindrical structures and a yellow light fixture.

Командное  
тестирование

Что же ты такое?



## Командное тестирование

Я, гражданин Союза Советских Социалистических Республик, вступаю в ряды Рабоче-Крестьянской Красной Армии, принимаю присягу и торжественно клянусь быть честным, храбрым, дисциплинированным, бдительным бойцом, строго хранить военную и государственную тайну, беспрекословно выполнять все воинские уставы

# Определение



## Командное тестирование

Армии, правильно расходуя и терпеливо выслушивая быть честным, храбрым, дисциплинированным, бдительным бойцом, стрелком, храбрым, волевым и государственному делу, беспрекословно выполнять все воинские уставы и приказы командиров и начальников.

Я люблю добросовестно изучать военное дело, всегда беречь оружие и оружием выносливым и до последнего дыхания быть преданным своему Народу, своей Советской Родине и Рабоче-Крестьянскому Правительству.

Я всегда готов по призыву Рабоче-Крестьянского Правительства выступить на защиту моей Родины — Союза Советских Социалистических Республик и как член Рабоче-Крестьянской Красной Армии, я обязуюсь защищать ее мужественно, умело, с достоинством и честью, не щадя своей крови и своей жизни для достижения полной победы над врагами.

Если же по злому умыслу и варвару эту мою торжественную присягу, то пусть меня постигнет суровый кара советского закона, всеобщая невеста и презрение трудящихся.

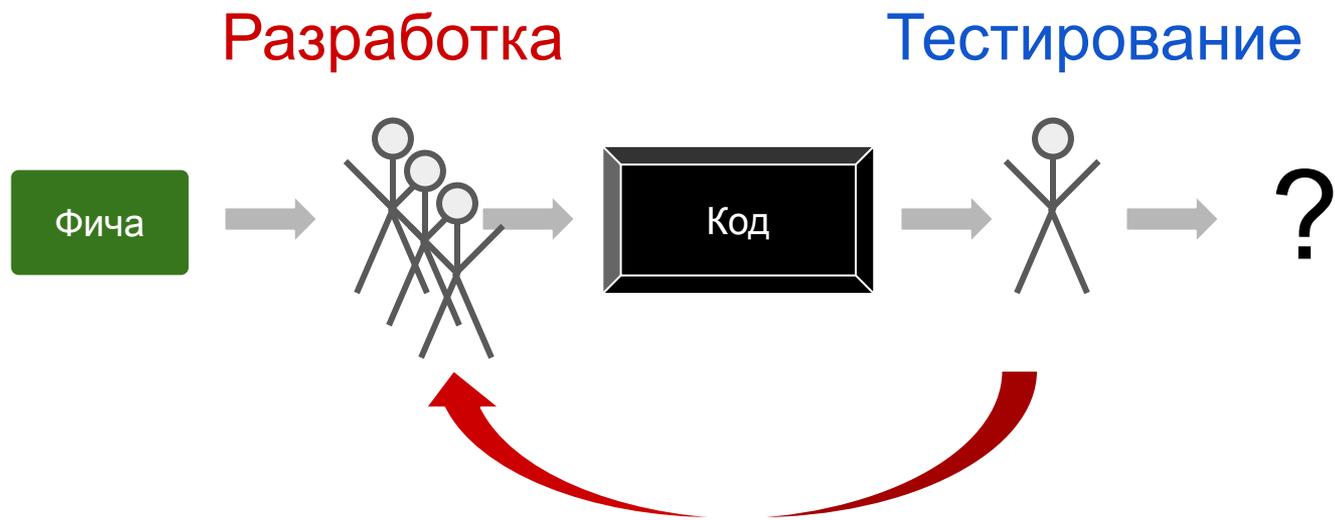
# Определение не найдено





А у вас есть QA?

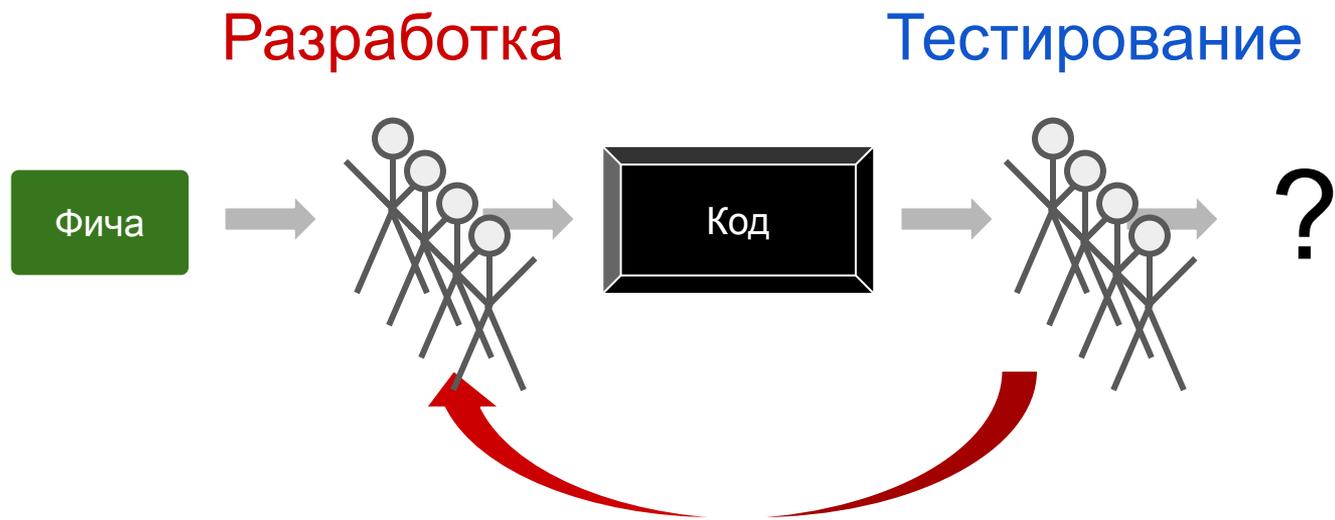
# Универсальный процесс компании



А отдел QA?



# Универсальный процесс компании



? — и к какому результату это приводит

Нет коллективного  
владения тестами

Сложные и  
«падучие» тесты

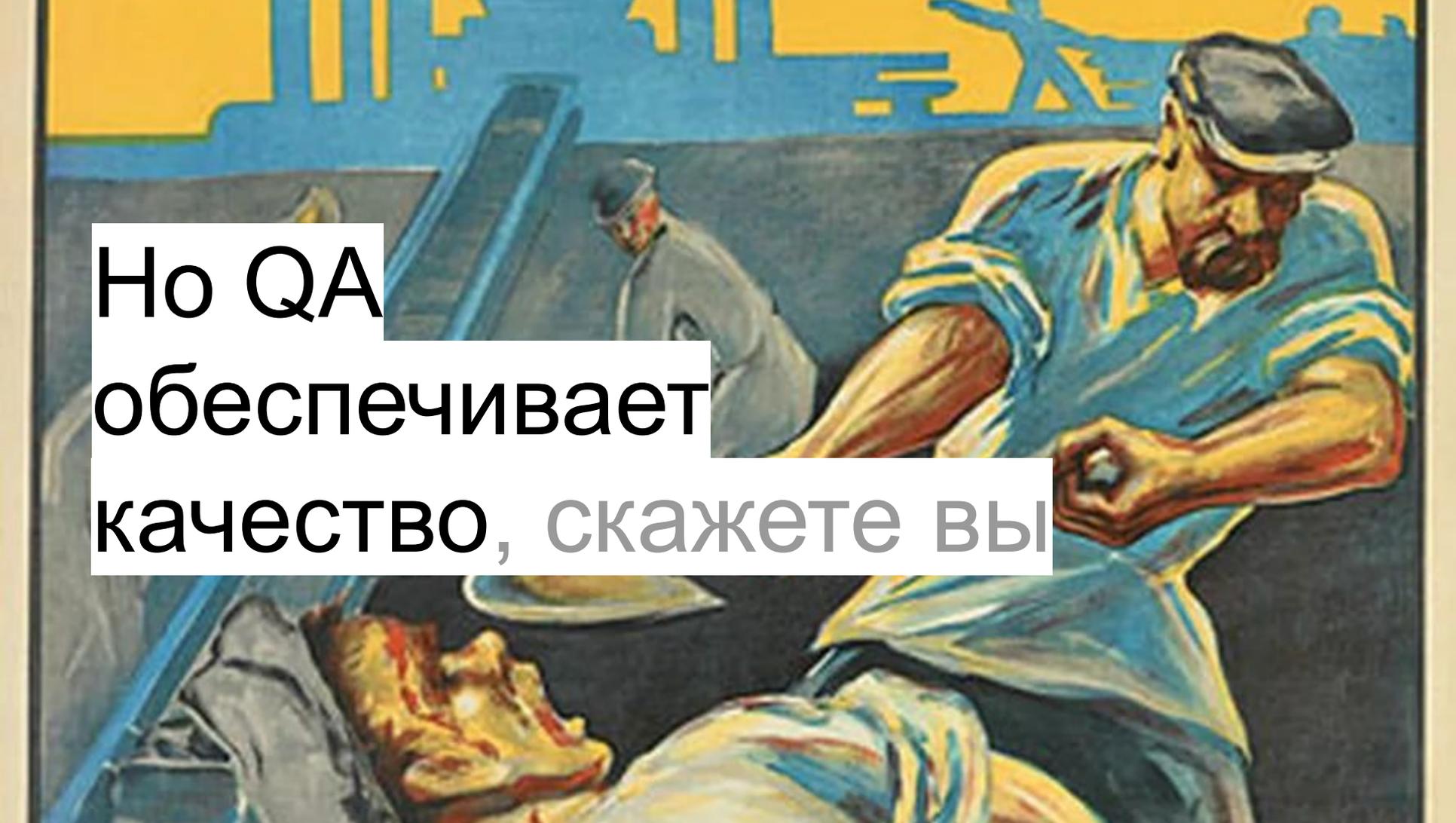
Долгие регрессы

Непрозрачная  
зависимость тестов и  
компонент системы

Частые геореп

Чувствительность  
к данным

Сложность  
настройки окружения



Но QA  
обеспечивает  
качество, скажете вы



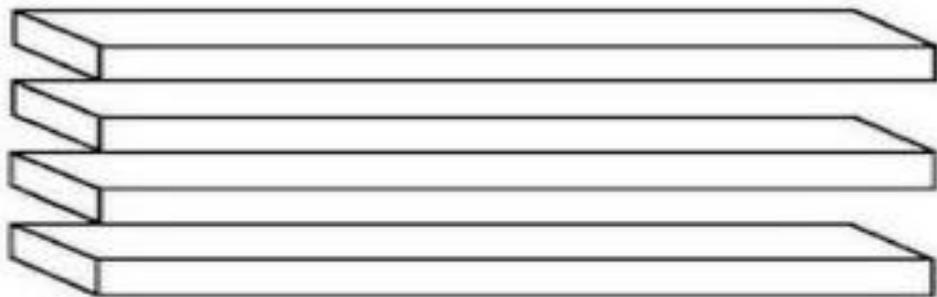




**НЕТ!**



# Священная война



Разработка

Тестирование





# А как вы

Проверяете

1. ваши тестовые сценарии на критерий достаточности?
2. покрытие бизнес сценариев тестами на разных уровнях?

Следите за

1. дублированием тестов на разных слоях?
2. проблемными местами системы целиком?





**Стыдно!**

## Как это бывает

**3 774 357 ₹** 158 586 ₹/м<sup>2</sup>

В ипотеку за NaN,undefined ₹/мес. ▼

Спасибо!



Как это бывает

**3 774 357 ₹** 158 586 ₹/м<sup>2</sup>

В ипотеку за NaN,undefined ₹/мес. ▾

Спасибо!



## Как это бывает

3 774 3!

В ипотеку за N

Спасибо!



**0 в месяц**

оплата undefined ежегодно

undefined в месяц

# Командное тестирование



Когда тестирование идет совместно с созданием продукта

И каждая роль в команде сфокусирована на обеспечении качества

Я, гражданин Союза Советских Социалистических Республик, вступая в ряды Рабоче-Крестьянской Красной Армии, принимаю присягу и торжественно клянусь быть честным, храбрым, дисциплинированным, безбоязненным бойцом, строго хранить военную и государственную тайну, беспрекословно выполнять все воинские уставы и приказы командиров и начальников.

Я клянусь добросовестно изучать военное дело, всемерно беречь оружие и народное имущество и во всяком случае быть преданным своему Наркату, своей Советской Родине и Рабоче-Крестьянскому Правительству.

Я всегда готов по призыву Рабоче-Крестьянского Правительства выступить на защиту моей Родины—Союза Советских Социалистических Республик и, как

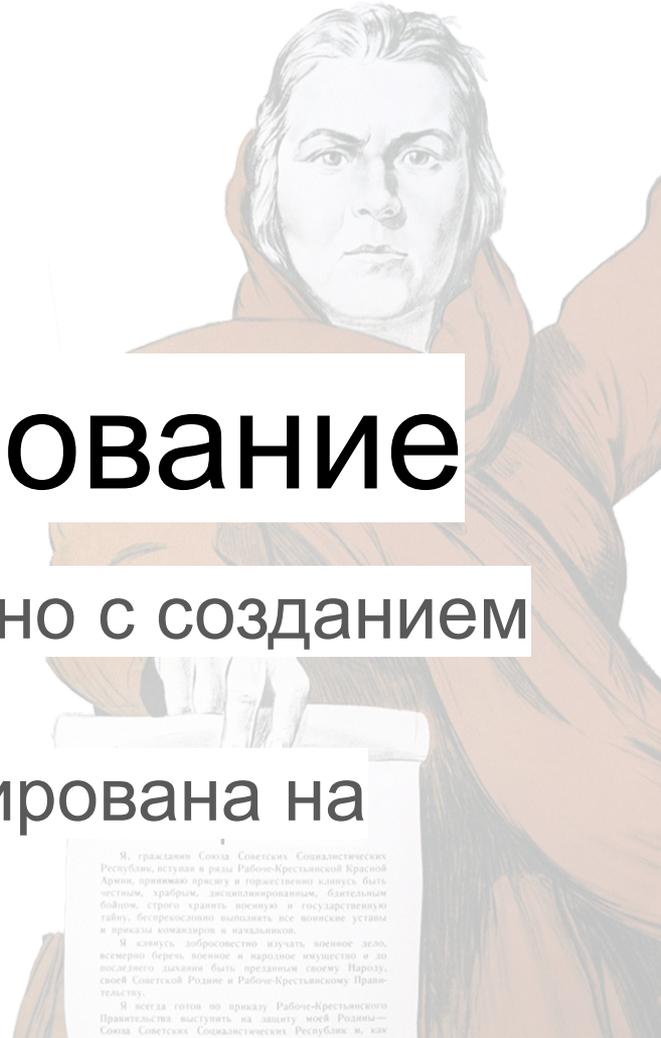
Наш тезис



# Командное тестирование

Когда тестирование идет совместно с созданием продукта

И каждая роль в команде сфокусирована на обеспечении качества



Я, гражданин Союза Советских Социалистических Республик, вступая в ряды Рабоче-Крестьянской Красной Армии, принимаю присягу и торжественно клянусь быть честным, храбрым, дисциплинированным, беззаветным бойцом, строго хранить военную и государственную тайну, беспрекословно выполнять все воинские уставы и приказы командиров и начальников.

Я клянусь добросовестно изучать военное дело, всемерно беречь оружие и народное имущество и до последнего дыхания быть преданным своему Наркду, своей Советской Родине и Рабоче-Крестьянскому Правительству.

Я всегда готов по призыву Рабоче-Крестьянского Правительства выступить на защиту моей Родины—Союза Советских Социалистических Республик и, как

# Предистория

# HEISENBUG

Тестирование. Не только для тестировщиков!

Как добраться

Программа

Схема площадки



17-18 мая 2019



Гостиница «Park Inn by Radisson Пулковская»

Санкт-Петербург, пл. Победы, 1 (вход с Варшавской ул.)

# Предистория

Как разработчик ...

# Покупает машину

habr

Публикации

Новости

Пользователи

Хабы

Компании

Песочница

 nikitos18 7 июня 2016 в 11:57

## Как программист машину покупал

Data Mining, R, Машинное обучение

Из песочницы

Недавно я озадачился поиском б.у. автомобиля, взамен только что проданного, и, как это обычно бывает, на эту роль претендовали несколько конкурентов.

Как известно, для покупки авто на территории РФ существует несколько крупных авторитетных сайтов (auto.ru, drom.ru, avito.ru), поиску на которых я и отдал предпочтение. Моим требованиям отвечали сотни, а для некоторых моделей и тысячи, автомобилей, с перечисленных выше сайтов.

Помимо того, что искать на нескольких ресурсах неудобно, так еще, прежде чем ехать смотреть

# Покупает 3D принтер

**habr**

Публикации

Новости

Пользователи

Хабы

Компании

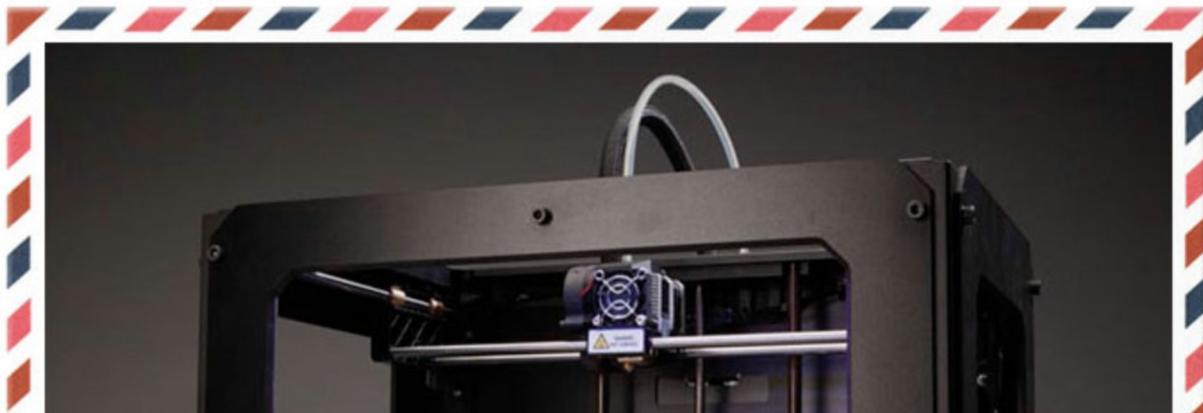
Песочница



GoldSerg 11 декабря 2013 в 10:10

## Как я покупал 3D-принтер

3D-принтеры



Итак, машина есть



Итак, машина есть, 3D принтер есть



# Покупает квартиру

**habr**

Публикации

Новости

Пользователи

Хабы

Компании

Песочница



ffriend 31 июля 2012 в 06:31

## Как я покупал квартиру

Data Mining, R

Я хотел написать статью про линейную регрессию, но потом подумал, да ну её, лучше куплю квартиру. И пошёл искать, что предлагают. А предлагают, как оказалось, много чего. В подходящий мне ценовой диапазон попало больше 500 квартир. И что, мне теперь все это просматривать? Ну неееет, программист я в конце концов или не программист. Надо это дело как-то автоматизировать.

Сбор данных



Ипотека

Кирилл и Лена  
берут Ипотеку!

КТО-КУДА

А Я

В СБЕРКАДГУ!





Кирилл и Лена  
берут Ипотеку!

КТО-КУДА

Хочешь ипотеку – сделай

А Я  
В СБЕРКАССУ!

# Контракт - user story

1. Я как пользователь хочу отправить запрос на ипотеку
2. Я как пользователь хочу узнать одобрена ли мне ипотека
3. Я как пользователь хочу увидеть детали ипотечных предложений

# Spec By Example — зафиксируем что делать

**Given:** я открыл сервис

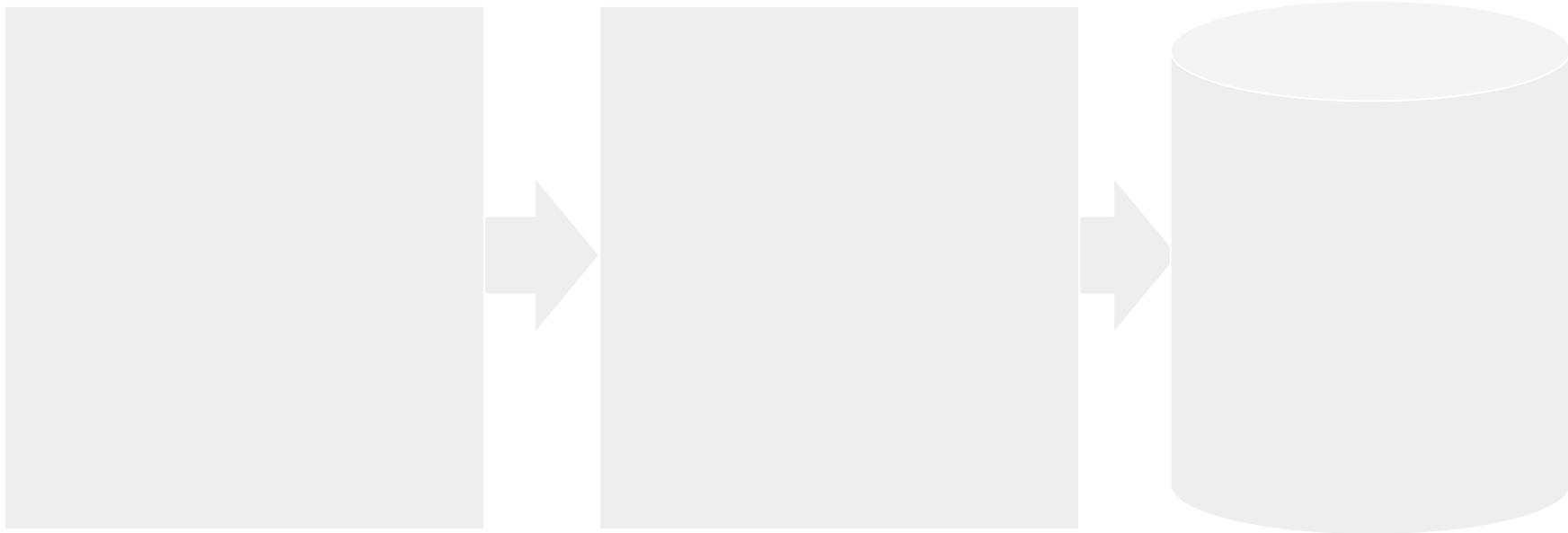
**When:** жму «Хочу ипотеку»

**Then:** вижу одобрена мне ипотека или нет

**Then:** вижу предложения от банков



# Универсальная архитектурная диаграмма

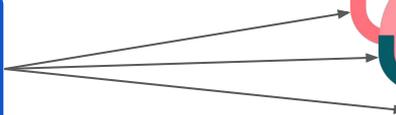


# Что делаем?

UI показа ипотечных предложений



API взаимодействия с банками



# Что делаем? Ипотека по Cookies!



# Универсальная архитектурная диаграмма



# Контракт – Backend/Frontend

```
{  
  "status": "SUCCESS",  
  "offers": [{  
    "rate": 10.99,  
    "monthlyPayment": 52000,  
    "term": 20  
  }, {  
    "rate": 14.5,  
    "monthlyPayment": 20000,  
    "term": 30  
  }]  
}
```

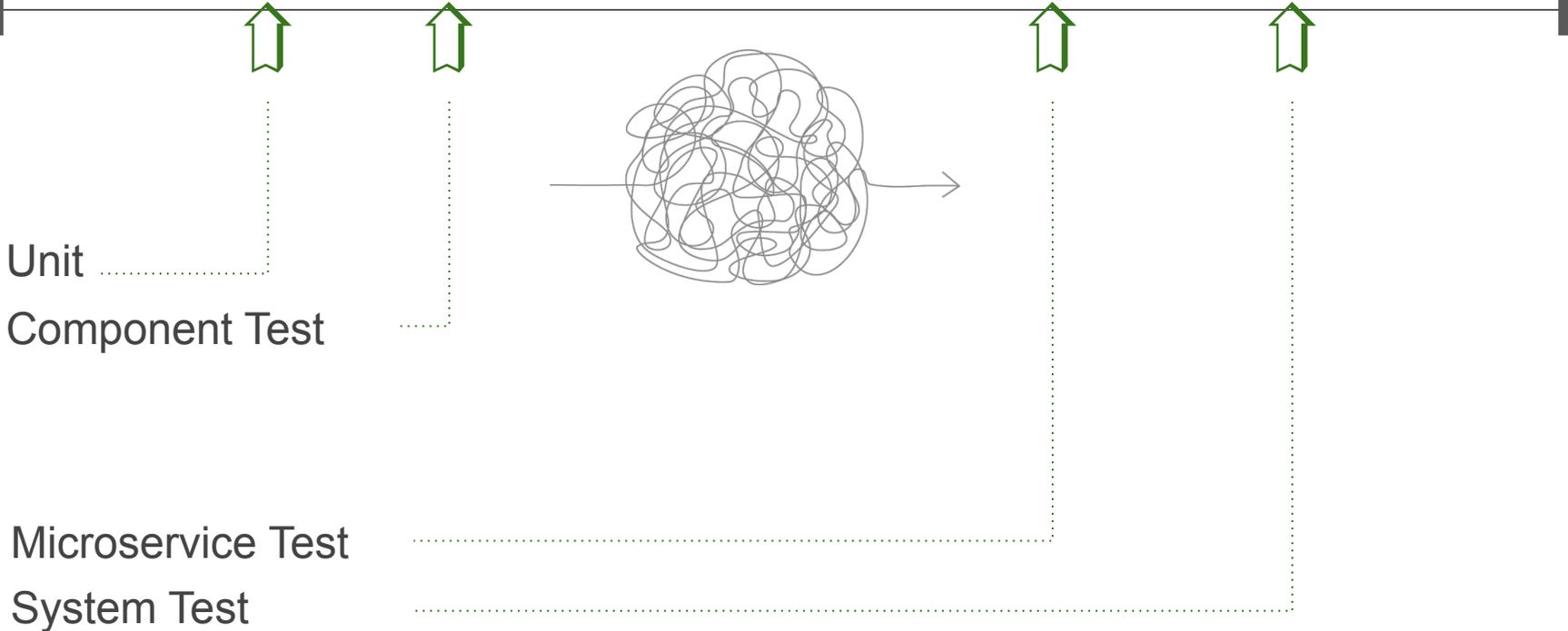


# Немного теории

О чём будем говорить



# Шкала Тестов



Unit  
Component Test

Microservice Test  
System Test

НОВЕЙШАЯ ТЕХНИКА-  
ПУТЬ К ПОВЫШЕНИЮ  
ПРОИЗВОДИТЕЛЬНОСТИ  
ТРУДА!

Начнём

Овладею!

Frontend

```
$ npx create-react-app team-testing-ui --typescript
```



```
$ npx create-react-app team-testing-ui --typescript
```





```
$ npx create-react-app team-testing-ui --typescript
```

Creating a new React app in `team-testing-ui`.

Installing packages. This might take a couple of minutes.

Installing react, react-dom, and react-scripts...

```
yarn add v1.10.1
```

```
[1/4] 🔍 Resolving packages...
```

```
[2/4] 🚚 Fetching packages...
```

```
[3/4] 🔗 Linking dependencies...
```

```
[#####-----] 19782/37337
```



```
$ npx create-react-app team-testing-ui --typescript
```

```
Creating a new React app in team-testing-ui.
```

```
Installing packages. This might take a couple of minutes.
```

```
Installing react, react-dom, and react-scripts...
```

```
yarn add v1.10.1
```

```
[1/4] 🔍 Resolving packages...
```

```
[2/4] 🚚 Fetching packages...
```

```
[3/4] 🔗 Linking dependencies...
```

```
[#####-----] 19782/37337
```

```
...
```

```
Happy hacking!
```

```
$ npm install --save-dev \  
  @types/enzyme \  
  enzyme \  
  enzyme-adapter-react-16
```





```
$ npm install --save-dev \  
  @types/enzyme \  
  enzyme \  
  enzyme-adapter-react-16
```

```
+ @types/enzyme@3.9.1  
+ enzyme-adapter-react-16@1.13.0  
+ enzyme@3.9.0
```

```
added 3 packages and audited 889529 packages in 14.531s  
found 0 vulnerabilities
```



```
$ bat src/setupTests.js
```

```
1 | import { configure } from 'enzyme';  
2 | import Adapter from 'enzyme-adapter-react-16';  
3 |  
4 | configure({ adapter: new Adapter() });
```

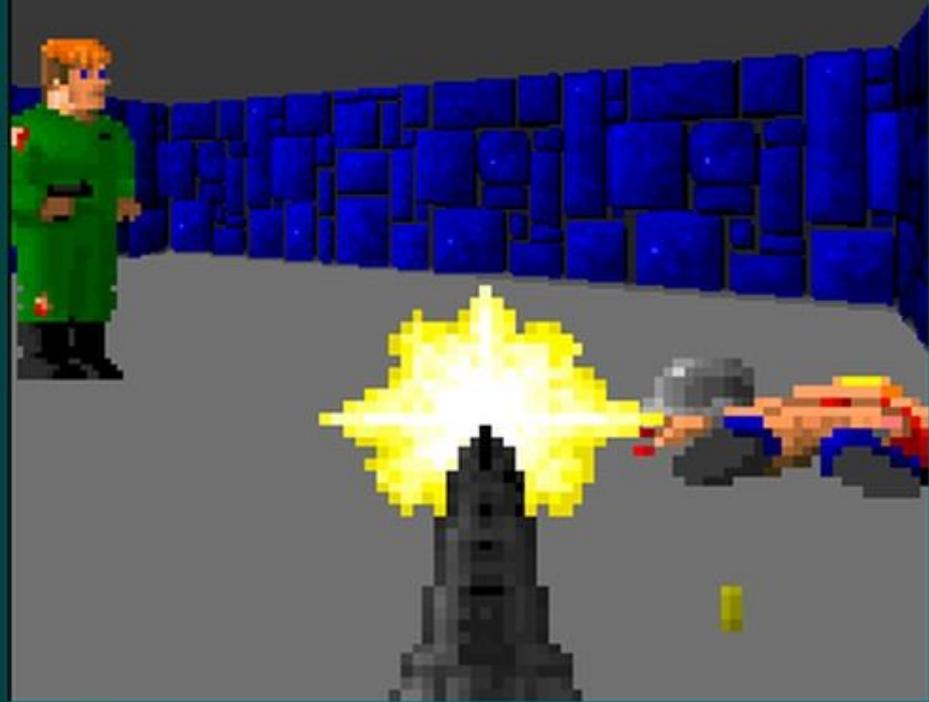
# Время кодить



Тем временем...



LEVEL	SCORE	LIVES		HEALTH	AMMO	
1	0	∞		100%	8	



LEVEL	SCORE	LIVES		HEALTH	AMMO	
1	0	∞		100%	95	

Тем временем...



Frontend

LEVEL	SCORE	LIVES		HEALTH	AMMO	
1	0	∞		100%	8	



Backend

LEVEL	SCORE	LIVES		HEALTH	AMMO	
1	0	∞		100%	95	

Backend



# Откуда берётся backend <https://start.spring.io>

---



Spring **Initializr**

Bootstrap your application

## Project

Maven Project

Gradle Project

## Language

Java

Kotlin

Groovy

## Spring Boot

2.2.0 M2

2.2.0 (SNAPSHOT)

2.1.5 (SNAPSHOT)



# Откуда берётся backend <https://start.spring.io>

**Dependencies** [See all](#)

Search dependencies to add

web

## **Reactive Web** [Web]

Reactive web applications with Spring WebFlux and Netty



## **Rest Repositories** [Web]

Exposing Spring Data repositories over REST via Spring Data REST

## **Web Services** [Web]

Contract-first SOAP services with Spring Web Services

2.1.4

1.5.20

# Откуда берётся backend <https://start.spring.io>



## Project Metadata

Group

ru.team.testing

Artifact

testing-api

More options

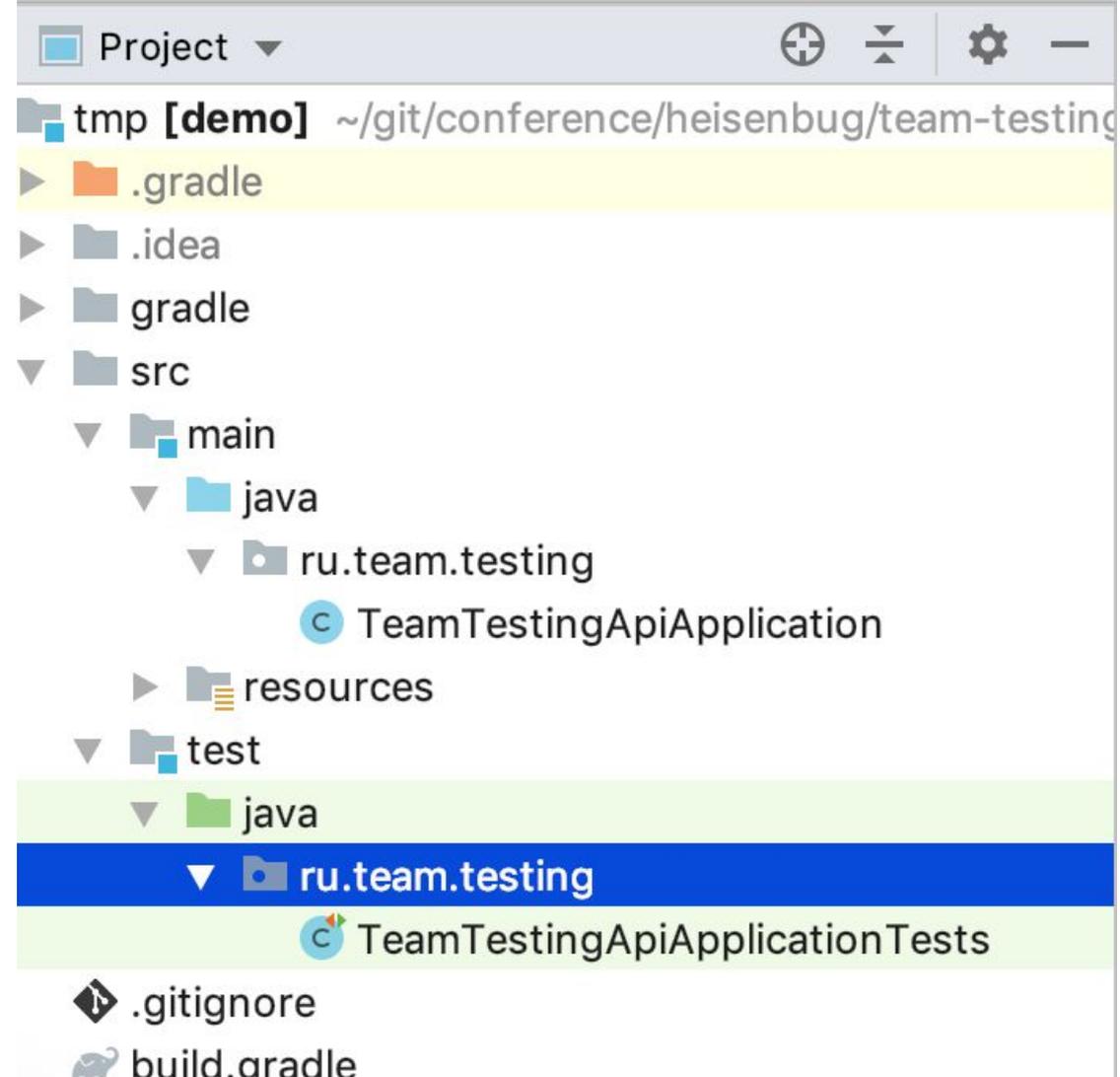
Dependencies [See all](#)

Generate Project - ☰ + ↵



# Или тоже самое в терминале

```
$ http start.spring.io/starter.zip \  
  name==team-testing-api \  
  packageName==ru.team.testing \  
  type==gradle-project \  
  dependencies==lombok,web -d  
$ unzip demo.zip  
$ idea .
```



Получили  
готовый проект

# Время кодить



Нет времени объяснять  
пора в Production



# Время кодить



# Spec By Example — вспомним что делать

**Given:** я открыл сервис

**When:** жму «Хочу ипотеку»

**Then:** вижу одобрена мне ипотека или нет

**Then:** вижу предложения от банков

# Chrome Capture Cookies

# Chrome Security



```
launch: {  
  args: [  
    '--disable-features=NetworkService,  
    '--disable-site-isolation-trials',  
  ]  
}
```



# Chrome Security

← → ↻ Chrome | chrome://flags ☆ 👤 ⋮

🔍 network servi ✕

Reset all to default

## Experiments

74.0.3729.131

Available

Unavailable

● **Enable network service**

Enables the network service, which makes network requests through a separate process. – Mac, Windows, Linux, Chrome OS, Android

[#network-service](#)

Enabled ▾

# Cookies is available now!

× Headers Preview Response Cookies Timing

**Connection:** keep-alive

**Cookie:** x-generated-userid=fcaf71a0-f336-11e8-8128-5fe81f0352e0; no\_cache=1; Idea-84d69fbc=894a970e-6a0e-49a6-9bc8-f098aaf769; userId=1f8f5d50-1d14-4346-89e2-bdd2fd185b58; jv\_visits\_count\_hMFXm0cLY0=9

**Host:** localhost:8080

**Origin:** http://localhost:3000

**Pragma:** no-cache

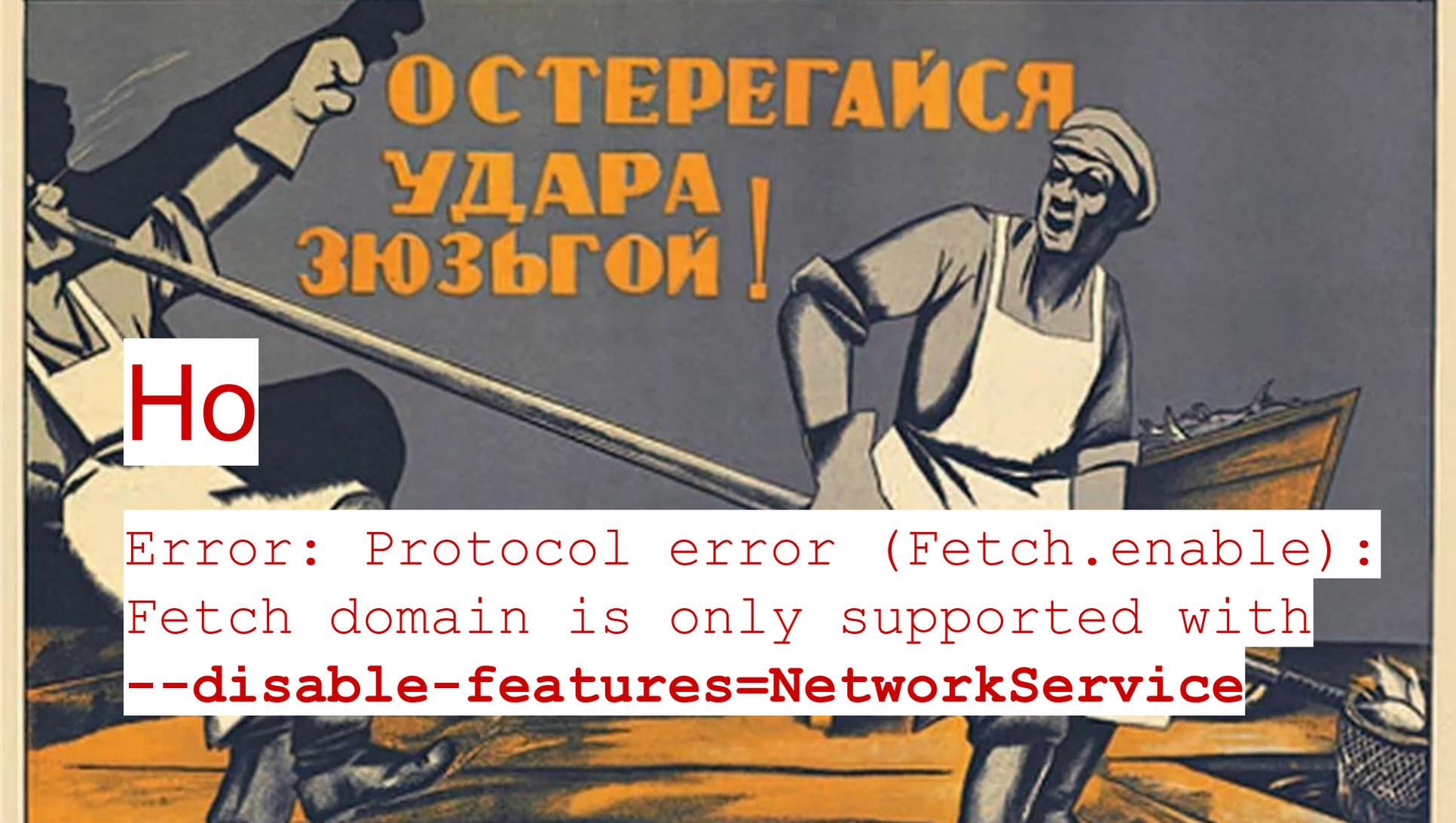
**Referer:** http://localhost:3000/test/offers

**User-Agent:** Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_13\_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.131 Safari/537.36

# Тест на отправку печенек

```
it('проверяем что печеньки отправляются', async (done) => {
  await page.goto('http://demo.ru:3000');
  await page.setRequestInterception(true);
  await page.setCookie({name: 'печенька', value: 'пяточок'});
  page.on('request', request => {
    if (interceptedRequest.url().endsWith('/offers')) {
      expect(request.headers()['cookie']).toContain('печенька');
      request.continue(); done();
    }
  });
  const button = await page.waitForXPath('//*[@text()="Хочу ипотеку"]');
  await button.click();
});
```

Ho...



ОСТЕРЕГАЙСЯ  
УДАРА  
ЗЮЗЬГОЙ!

Но

Error: Protocol error (Fetch.enable):  
Fetch domain is only supported with  
**--disable-features=NetworkService**



**RTFM!**

# Не беда, Unit тесты спешат на помощь



```
it('должен быть выставлен флаг отправки печенек', () => {
  jest.spyOn(global, 'fetch')
    .mockImplementation(() => Promise.resolve({
      json: jest.fn()
    }));
  return fetchDecisionStatus().then(value => {
    expect(fetch).toHaveBeenCalledWith(
      'http://demo.ru:8080/offers',
      { method: 'GET', credentials: 'include' }
    )
  })
});
```



# Время кодить





# Про cookie и puppeteer

```
it('проверяем что куки отправляются', async (done) => {
  await page.goto('http://demo.ru:3000');
  await page.setRequestInterception(true);
  await page.setCookie({name: 'userId', value: 'пяточок'});

  page.on('request', interceptedRequest => {
    if (interceptedRequest.url().endsWith('/offers')) {
      expect(interceptedRequest.headers()['cookie']).toContain('userId');
      interceptedRequest.continue();
      done();
    }
  });
  const button = await page.waitForXPath('//*[@text()="Хочу ипотеку"]');
  await button.click();
});
```

# Про cookie и puppeteer vs selenium





# Про cookie и puppeteer vs selenium

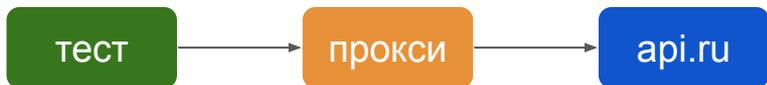
Через предустроенный проху





# Про cookie и puppeteer vs selenium

Через предустроенный проху



Через browser extension





# End to End JS

```
it('Кирилл и Лена хотят ипотеку (e2e)', async () => {  
  await page.goto('http://localhost:3000');  
  await (await page.waitForXPath(  
    '//*[text()="Хочу ипотеку"]'  
  )).click();  
  await page.waitForXPath('//*[text()="Ждёмс..."]');  
  await page.waitForXPath('//*[text()="SUCCESS"]');  
  
  await page.waitForXPath('//*[contains(text(),"Срок  
кредита:")]');  
});
```



# End to End Groovy

```
@Test
void 'Кирилл и Лена хотят ипотеку (e2e)' () {
    given:
        open 'http://localhost:3000'

    expect:
        $ byText("Хочу ипотеку") should(exist) click()
        $ byText("Ждёмс...") should(exist)
        $ byText("SUCCESS") should(exist)
}
```



# Вывод #2

Контракт **соблюдай**, инструменты бери **удобные**





Жизнь после выхода в Production





Прод не лежит, он отдыхает

# Поддержка:

Да вы@#\$34 опять 2%;№”. E#\$@u23Tryu !!



Прод не лежит, он отдыхает

Поддержка:

Да вы@#\$34 опять 2%;№”. E#\$@u23Tryu !!

Разработчик:

Какого ?%:@#/ Вы@!#!. Я так ”№?К%”№%” немогу!!



Прод не лежит, он отдыхает

Поддержка:

Да вы@#\$34 опять 2%;№”. E#\$@u23Tryu !!



Прод не лежит, он отдыхает

Разработчик:

Какого ?%:@#/ Вы@!#!. Я так ”№?К%”№%” немогу!!

Поддержка:

У нас @\$34у. Завтра у#\$@ будет \$\$\$@% ±!!!! E#\$@u23Tryu не объясняете!

Разработчик:

Ds к?%АП. Как всегда \$#%\$@, на своём \$\$\$%\$#@. Не формулируете!

## SUCCESS

Ставка: 10.96 %

Срок кредита: 10 лет

Ежемесячный платёж: 65140 руб

---

Ставка: 9.5 %

Срок кредита: 20 лет

Ежемесячный платёж: 91673 руб

---

## SUCCESS

Ставка: 12.00 %

Срок кредита: 5 лет

Ежемесячный платёж: 99999 руб

---

Ставка: 10.99 %

Срок кредита: 15 лет

Ежемесячный платёж: 102000 руб

---

# В чём причина?

Бэк тормозит!





В чём причина?

Фронт много запросов шлёт!

# Время кодить

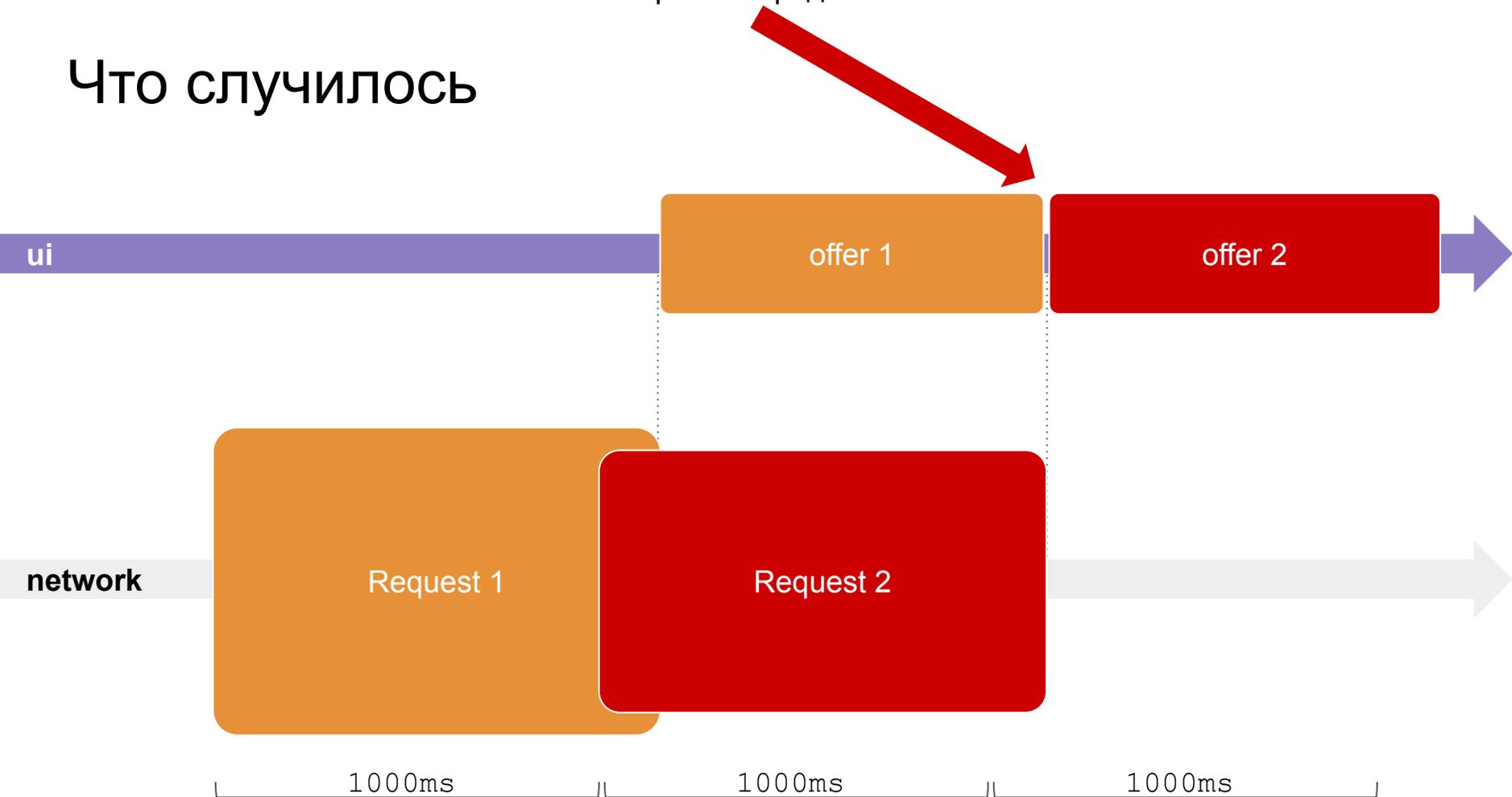


# Что случилось

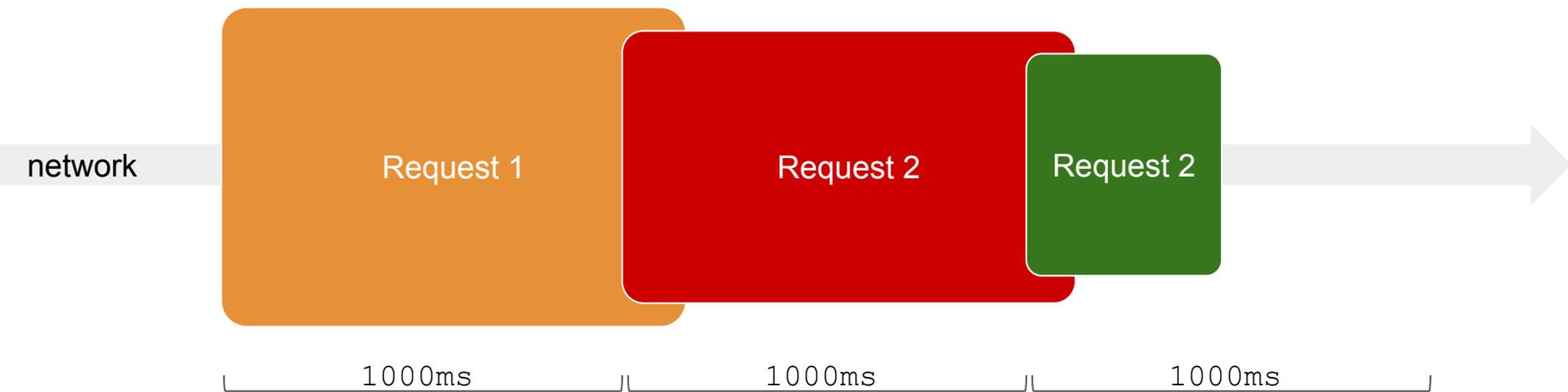


# Что случилось

Моргание предложений



# setInterval



# setTimeout



# Вывод

Хорошо знать что ты делаешь и разбираться вокруг



# Локално запускатъ тесты

Не грешновато ли?

# Локально запускать тесты

Не грешновато ли?



Грешновато!

Всё ещё

стыдно!

Как понять что не так?



**ОСТОРОЖНЕЕ С ЛОПАТОЙ**

# Отчет Gradle

## Package ru.team.testing.testtestingapi

all > ru.team.testing.testtestingapi

5

tests

0

failures

0

ignored

3.262s

duration

100%

successful

### Classes

Class	Tests	Failures	Ignored
<a href="#">OfferControllerTest</a>	3	0	0
<a href="#">OffersRepositoryTest</a>	1	0	0
<a href="#">TestTestingApiApplicationTests</a>	1	0	0

# Отчёт Maven

<b>Project Documentation</b> Maven Surefire Report						
built by: 						
<b>Summary</b>						
[Summary][Package List][Test Cases]						
Tests	Errors	Failures	Success Rate	Time		
24	0	0	100%	2.234		
Note: failures are anticipated and checked for with assertions while errors are unanticipated.						
<b>Package List</b>						
[Summary][Package List][Test Cases]						
Package	Tests	Errors	Failures	Success Rate	Time	
org.apache.maven.plugins.surefire.report	24	0	0	100%	2.234	
Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.						
<b>org.apache.maven.plugins.surefire.report</b>						
	Class	Tests	Errors	Failures	Success Rate	Time
	ReportTestCaseTest	4	0	0	100%	0.016
	ReportTestSuiteTest	7	0	0	100%	0.047
	SurefireReportMojoTest	4	0	0	100%	2.078
	SurefireReportParserTest	9	0	0	100%	0.093
<b>Test Cases</b>						
[Summary][Package List][Test Cases]						
<b>ReportTestCaseTest</b>						
	testSetName	0				
	testSetTestCases	0				

# Отчет Jest

PASS src/decision.test.tsx

...

PASS src/offer.test.tsx

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	79.31	93.33	75	79.31	
App.tsx	90	100	75	90	31
api-decision.ts	0	100	0	0	27,28
decision.tsx	100	75	100	100	21
index.tsx	0	100	100	0	6
offer.tsx	100	100	100	100	
poller.tsx	80	100	83.33	80	16,17

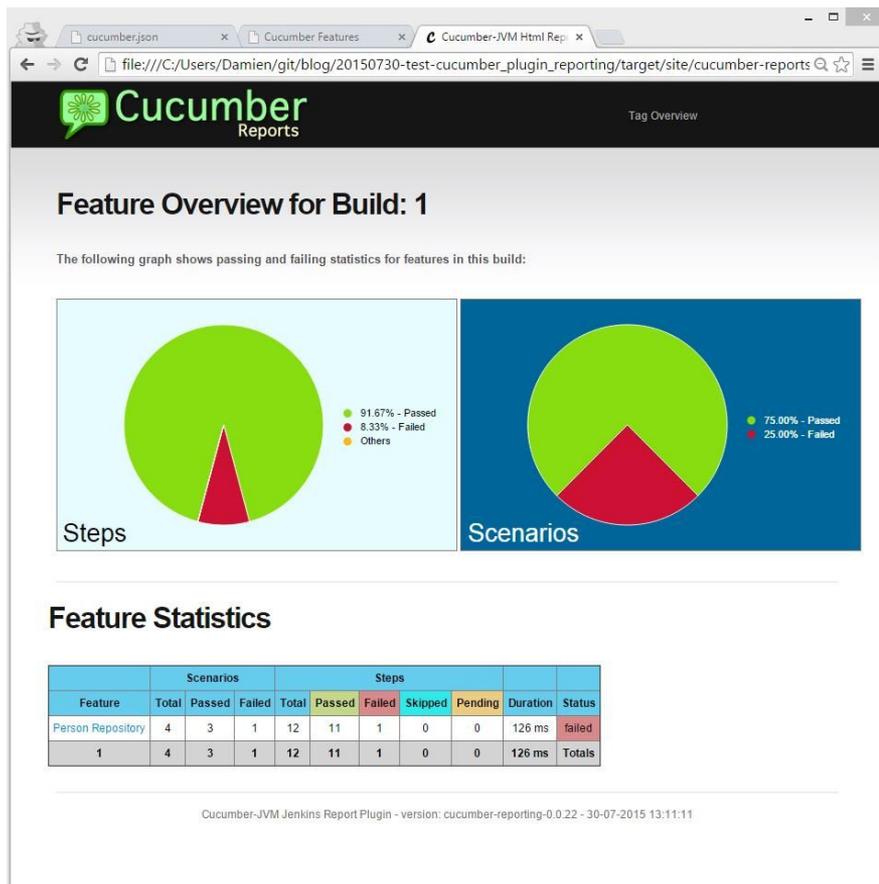
Test Suites: **4 passed**, 4 total

Tests: **11 passed**, 11 total

Time: 3.282s

Ran all test suites.

# Отчёт Cucumber



# Как понять что не так?

Репортим вместе с Allure



**ОСТОРОЖНЕЕ С ЛОПАТОЙ**

# Suites



order  name  duration  status 

Status: 0 0 5 0 0

Marks:  

▼ ru.team.testing.testtestingapi.OfferControllerTest	3
✓ #3 IN_PROGRESS пока банки обрабатывают запрос	4ms
✓ #2 SUCCESS когда пришел ответ от банков для пользователя	254ms
✓ #1 Метим пользователя если он анонимный	233ms
▼ ru.team.testing.testtestingapi.OffersRepositoryTest	1
✓ #1 Предложения банков должны быть сроком больше 1	2s 600ms
▼ ru.team.testing.testtestingapi.TestTestingApiApplicationTests	1
✓ #1 contextLoads	1ms

# Suites



order name duration status

Status: 2 0 25 0 0

Marks:  

▼ Интеграционные тесты ui				1
✓ #1 Эмулируем баг с продакшена 666				3s 399ms
> Компонен Decision				6
▼ Компонен Offer			1 1	
✗ #2 Отображаем информацию об оффере				5ms
✓ #1 Отображаем ифнормацию об оффере				16ms
> компонент оффера				1
▼ Хочу ипотеку				4
✓ #1 renders without crashing				16ms
✓ #2 жмя - и ждун предложений; кнопка хочу пропадает				6ms

# Контракт Allure

```
allure-results/
```

```
|- {id1}-testsuite.xml
```

```
|- {id2}-testsuite.xml
```

or

```
allure-results/
```

```
|- {id1}-result.json
```

```
|- {id2}-result.json
```

# Отчёты allure

```
$ allure report # -> generate html
```

```
#or
```

```
$ allure serve # -> generate html and open in browser
```

Нужно мёрзнуть  
репорты!



# Время кодить



# Контракт по которому allure мержит тесты

1. Скопировать содержимое всех директорий `allure-results/` в одну папку
2. Запустить команду `$ allure serve ALL_RESULTS_DIR`



# Отчёты allure

```
$ mkdir -p results
```

```
$ cp -rf team-testing-ui/allure-results/* results/
```

```
$ cp -rf team-testing-api/allure-results/* results/
```

```
$ cp -rf team-testing-e2e/allure-results/* results/
```

```
$ allure serve results/
```

# Suites

order name duration status

Marks:  

- > ru.integ.MorgageGetEndToEndTest
- > ru.team.testing.testtestingapi.OfferControllerTest
- > ru.team.testing.testtestingapi.OffersRepositoryTest
- > ru.team.testing.testtestingapi.TestTestingApiApplication
- > Интеграционные тесты ui
- > Компонен Decision
- > компонент оффера
- > получение ипотеки
- > Тесты на поллер решения от банка
- > Хочу ипотеку

КИРПИЧ  
УКЛАДЫВАЙ



# Вспомним про User Story

Контракт

У нас же был контракт





## Behaviors

order  name  duration  status  Status: 3 0 18 0 0Marks:  **Контракт**✓ Получение решения от банков 3 15✓ Обновление информации о решении 6

- ✓ #6 IN\_PROGRESS пока банки обрабатывают запрос 4ms
- ✓ #5 SUCCESS когда пришел ответ от банков для пользователя 254ms
- ✓ #1 каждую секунду мы спрашиваем апишку, готово ли решение? 8ms
- ✓ #4 Метим пользователя если он анонимный 233ms
- ✓ #2 мы прекращаем полить в тот момент, когда возвращается какое-то решение 1ms
- ✓ #3 уведомляем о решении 1ms

✓ Отображение предложения от банка 1

- ✓ #1 должен отображать параметры оффера 14ms

✓ Отображение решения банка 4

- ✓ #1 should render 9ms
- ✓ #2 должен отобразить статус решения 2ms
- ✓ #3 Если статус success мы показываем список офферов 5ms
- ✓ #4 Предложения банков должны быть сроком больше 1 2s 600ms

✓ Отправка запроса на ипотеку 4

- ✓ #1 renders without crashing 16ms



# Вывод

Контракт **соблюдай**, инструменты бери **удобные**



Выводы

# Выводы

1. **Хорошо знать что ты делаешь и разбираться вокруг**



# Переход к Зому выводу

Если процесс тестирования и разработки разорван – то получается тестирование чёрного ящика

Это сложнее и не оптимально. Привести в пример кейс с e2e тестом

Нарисовать картиночку кто что получил на вход

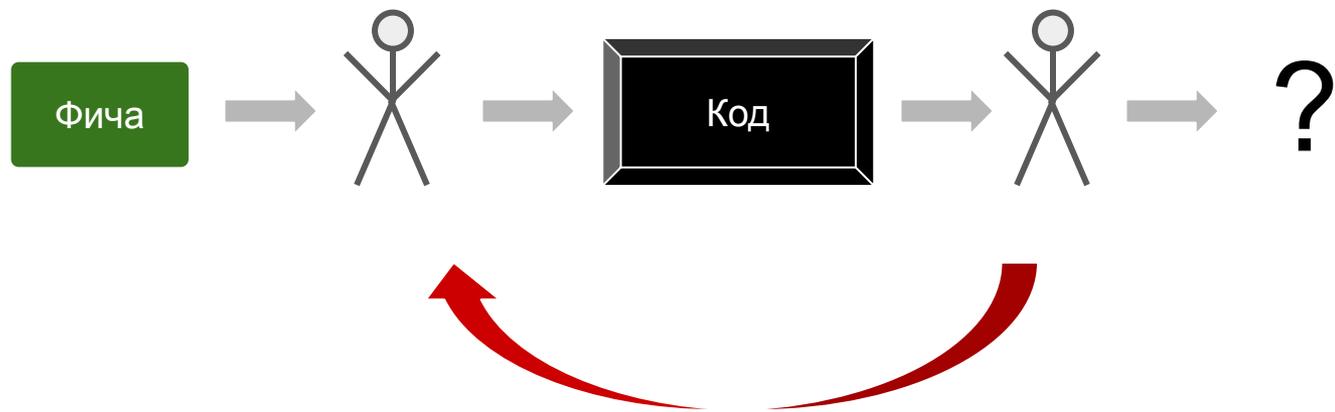
User story -> разработ -> черный ящик(код) -> тестировщик -> месиво e2e тестов

# Выводы

1. **Хорошо знать** что ты делаешь и **разбираться** вокруг
2. Контракт **соблюдай**, инструменты бери **удобные**



# Универсальный процесс компании



# Выводы

1. **Хорошо знать** что ты делаешь и **разбираться** вокруг
2. Контракт **соблюдай**, инструменты бери **удобные**
3. Тестирование **непрерывный** процесс  
и оно **неотделимо** от процесса реализации



# Выводы

1. Хорошо знать что ты делаешь и **разбираться** вокруг
2. Контракт **соблюдай**, инструменты бери **удобные**
3. Тестирование **непрерывный** процесс и оно **неотделимо** от процесса реализации



Не плачь – херачь

# QA



# Ссылки с материалами

1. [Spec By Example](#)
2. code [team-testing-api](#)
3. code [team-testing-ui](#)
4. code [team-testing-e2e](#)
5. [TDD By Example](#) Kent Beck