LevelUp Lunch presents: TDD vs BDD

The what, the how, the why and when…

And then let’s make them fight for fun and profit!

By: David - davidre@zillowgroup.com

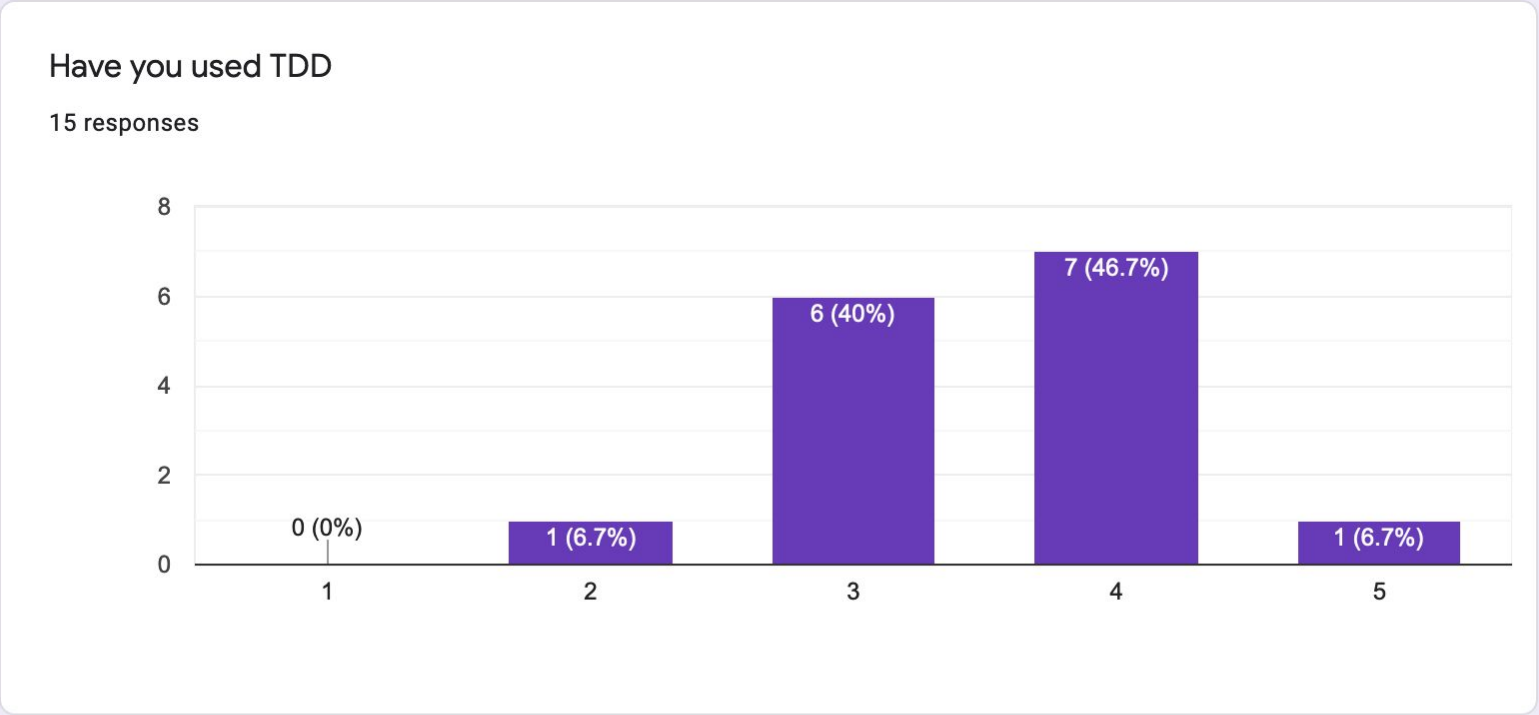
But first, let’s add some spice.

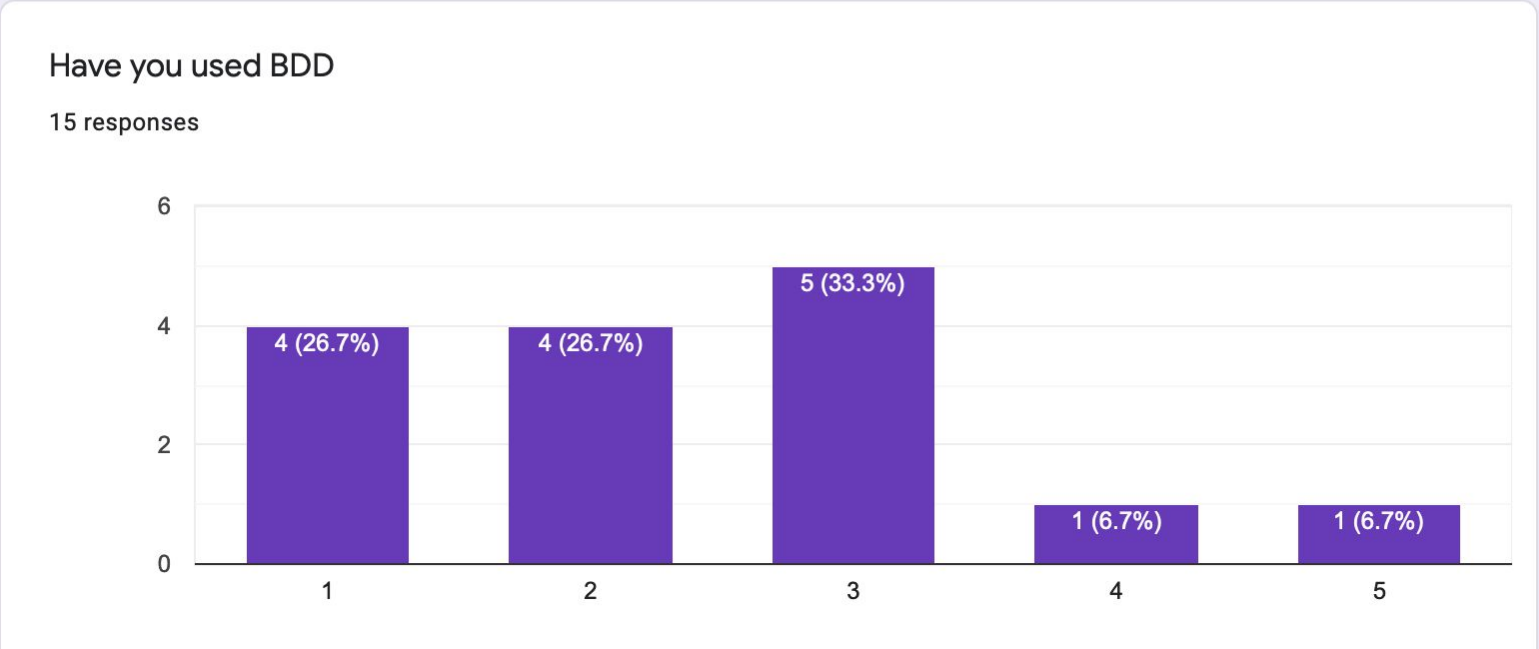
- In the blue-jeans chat, a link to an anonymous google survey has been sent, please fill it out in the next 30 secs.
- Please leave your questions in the chat, we will address them at the end!
- Agenda:
  - TTP
  - TDD
  - BDD
  - TDD vs BDD
  - TL;DR on TDD & BDD
  - Q&A about TDD/BDD/ATDD/ETC

Let’s check the results!

https://docs.google.com/forms/d/10Ag0euBeONUglO52Kfp276ryuRZ6vNyCjxn8j_slsuk/edit#responses

I don’t understand why it is so “hard” to embed a web-view in a presentation…

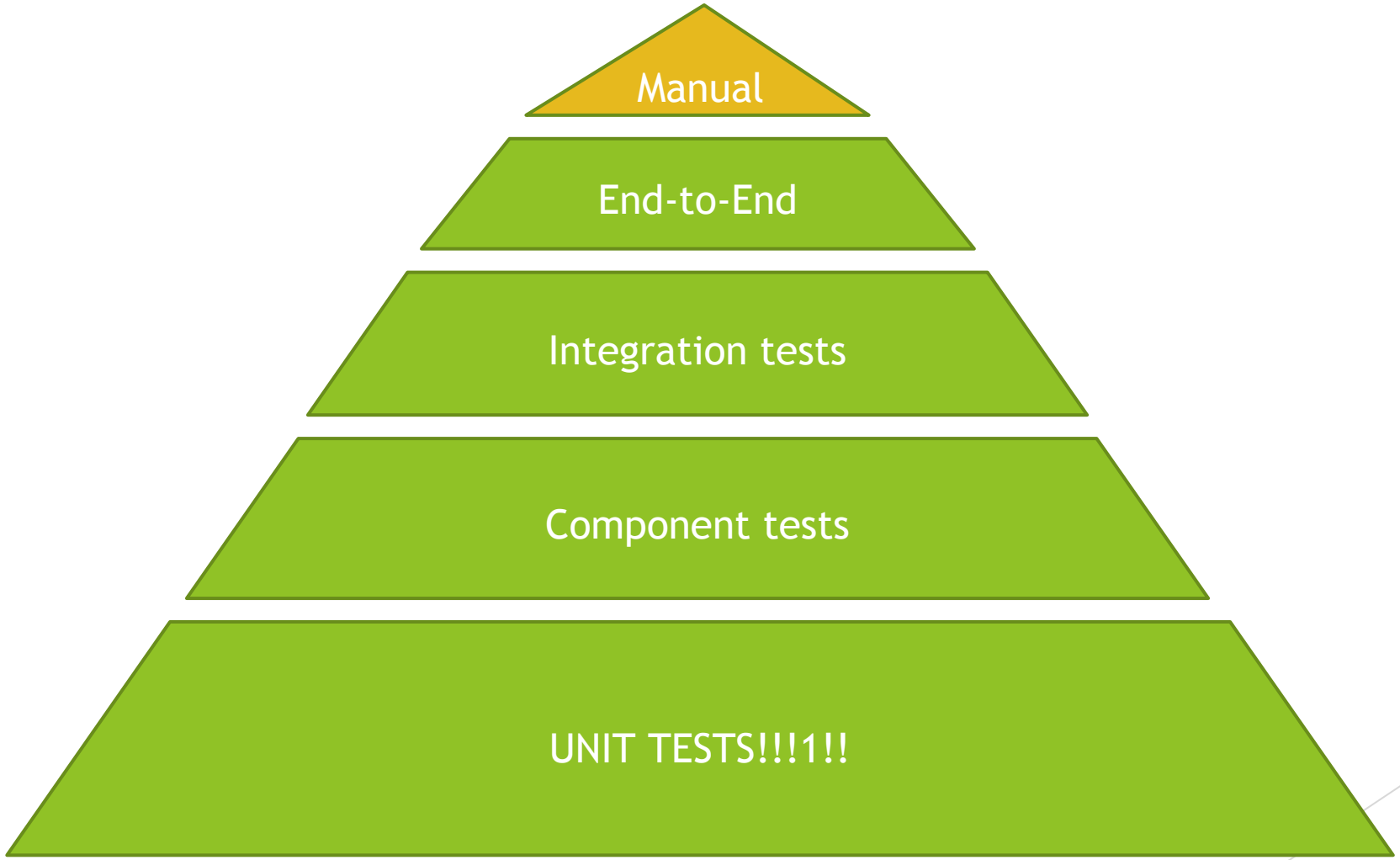

Have you used TDD
15 responses
8
6
4
2
0
0 (0%)
1 (6.7%)
6 (40%)
7 (46.7%)
1 (6.7%)
1
2
3
4
5

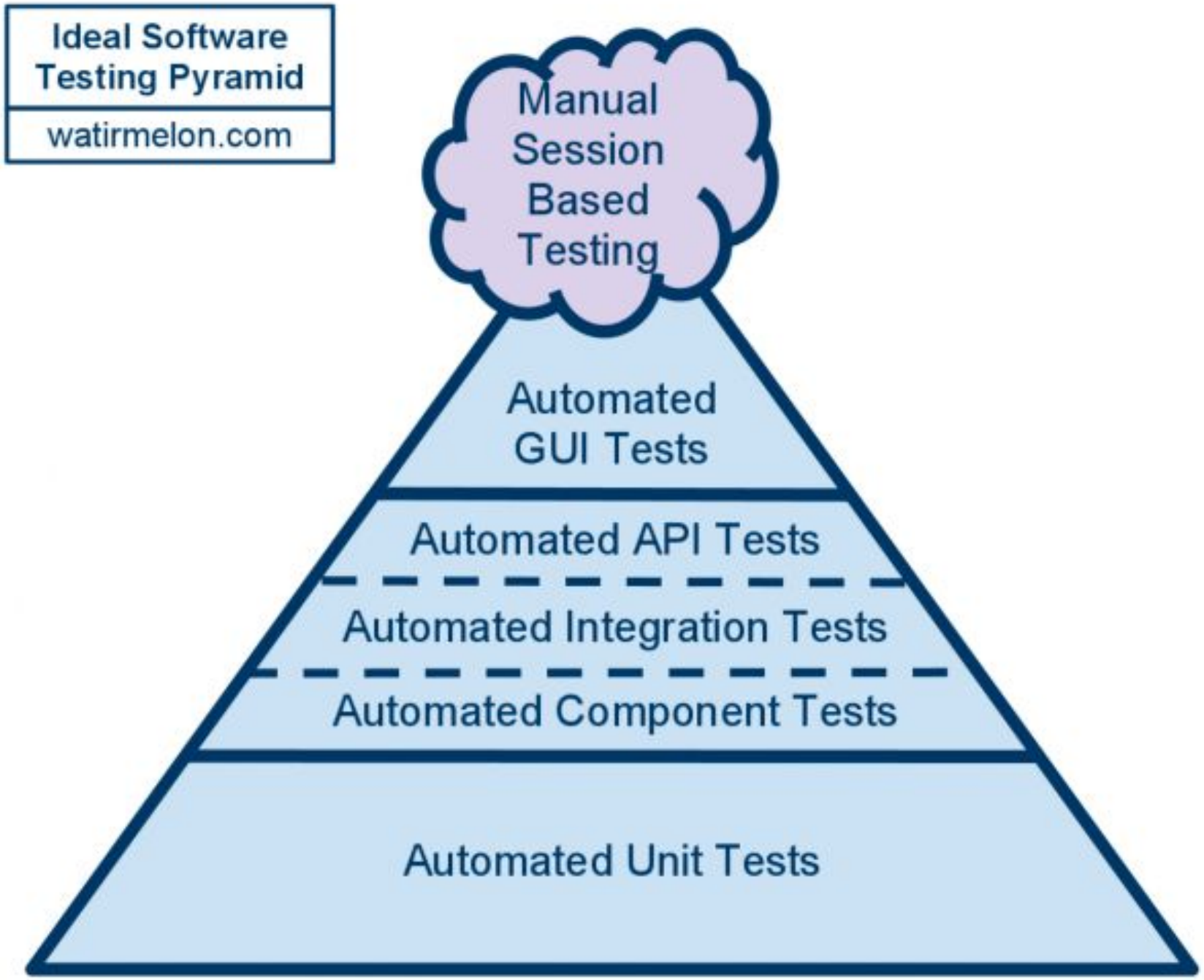

Have you used BDD
15 responses
6
4
2
0
4 (26.7%)
4 (26.7%)
5 (33.3%)
1 (6.7%)
1 (6.7%)
1
2
3
4
5

TTP

The Test Pyramid

The Testing Pyramid (my take)




Ideal Software Testing Pyramid
watirmelon.com
Manual Session Based Testing
Automated GUI Tests
Automated API Tests
Automated Integration Tests
Automated Component Tests
Automated Unit Tests

TDD

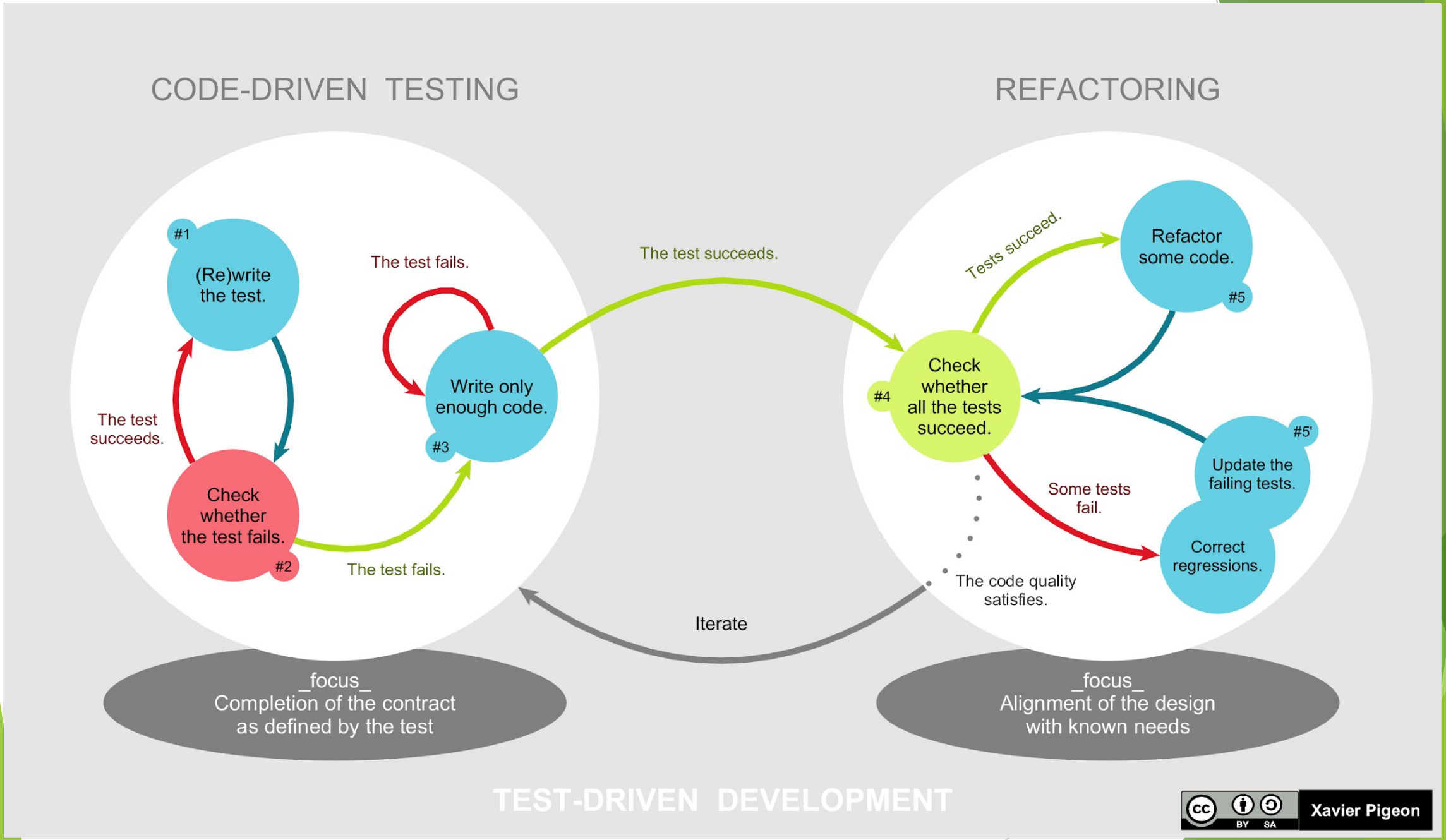
Test Driven Development

What is TDD
(Test Driven Development)

- An agile methodology to write code by focusing on unit tests first
- A testimonial from an internet rando:

> About 10 years ago, I was a TDD skeptic. I wasn't a *unit testing* skeptic, mind you – I'd accepted that as a helpful practice pretty much from the get-go.
>
> But TDD? I wasn't so sure.
>
> I decided I'd write a blog post about why TDD wasn't all that great.
>
> But I didn't want to just write a flimsy, dime-a-dozen opinion piece on the matter. So instead I decided to do a small client project (fixed price, BTW) rigidly following TDD so that I could write a post with the premise "I spent a couple of weeks doing pure TDD and it's not great."
>
> But fate had other plans.

- Let's take a look at the theory!



TDD in action!

Let’s see some examples! Onward to my IDE!

(Not really, that would be best left for a workshop)

Only write tests that bring value

```java
import ...

@Service
public class KeyHandler {

    private final KeyProvider keyProvider;

    @Autowired
    public KeyHandler(final KeyProvider keyProvider) { this.keyProvider = keyProvider; }

    /**
     * Return all the public keys that ZOT is using/aware of.
     *
     * @return PublicKeys containing the public key and corresponding keyId.
     *
     * @throws InternalServerErrorException If an internal error occurs in the service.
     */
    public PublicKeys getPublicKey() {
        return keyProvider.getPublicKeys();
    }

}
```

- rules 100% classes, 98% lines covered
  - basic 100% classes, 100% lines covered
    - AssignmentLogicHelper 100% methods, 100% lines covered
    - BasicRuleEngine 100% methods, 100% lines covered
  - hopscotch 100% classes, 100% lines covered
    - core 100% classes, 100% lines covered
      - AttributeEvaluatorHelper 100% methods, 100% lines covered
      - HopscotchDataFetcher 100% methods, 100% lines covered
    - model 100% classes, 100% lines covered
      - HopscotchAttributes 100% methods, 100% lines covered
      - HopscotchData 100% methods, 100% lines covered
    - HopscotchRule 100% methods, 100% lines covered
  - market 100% classes, 50% lines covered
    - MarketRuleEngine 0% methods, 50% lines covered
  - RuleSet 66% methods, 80% lines covered
  - RulesService 100% methods, 100% lines covered

```java
@Component
public class MarketRuleEngine {

    public String calculateAssignment(final String orid,

        // find Market for ORID

        // check saved attributes to match

        // roll the dice if necessary

        return "ON";
    }
}
```

Let your IDE help!

```java
public class AttributeEvaluatorHelperTest {

    private static final Map<String, String> VALID_ATTRIBUTE_MAP = ImmutableMap.of(
            VINTAGE, VINTAGE_YEAR,
            PURCHASED, PURCHASED_YEAR
    );

    private final AttributeEvaluatorHelper subject = new AttributeEvaluatorHelper();

    @Test
    public void testIsResultEligibleForHopscotch_returnsTrue_whenExpected() {
        final HopscotchData data = buildValidData();
        final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP);

        Assert.assertTrue(result);
    }

    @Test
    public void testIsResultEligibleForHopscotch_returnsFalse_whenDiscrepanciesAreFound() {
        final HopscotchData data = buildValidData()
                .setHasScrubbingDiscrepancies(Boolean.TRUE);
        final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP);

        Assert.assertFalse(result);
    }

    @Test
    public void testIsResultEligibleForHopscotch_returnsFalse_whenDiscrepanciesDataIsNotPresent() {
        final HopscotchData data = buildValidData()
                .setHasScrubbingDiscrepancies(null);
        final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP);

        Assert.assertFalse(result);
    }

    @Test
    public void testIsResultEligibleForHopscotch_returnsFalse_whenTimeFrameIsManyMonths() {
        final HopscotchData data = buildValidData()
                .setClosingTimeFrame("4-to-6-months");
        final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP);
```

```
private final AttributeEvaluatorHelper subject = new AttributeEvaluatorHelper();

@Test
public void testIsResultEligibleForHopscotch_returnsTrue_whenExpected() {
    final HopscotchData data = buildValidData();
    final boolean result = subject.isCaseE

    Assert.assertTrue(result);
}

@Test
public void testIsResultEligibleForHopscot
    final HopscotchData data = buildValidD
            .setHasScrubbingDiscrepancies(
    final boolean result = subject.isCaseE

    Assert.assertFalse(result);
}

@Test
public void testIsResultEligibleForHopscot
    final HopscotchData data = buildValidD
            .setHasScrubbingDiscrepancies(
    final boolean result = subject.isCaseE

    Assert.assertFalse(result);
}

@Test
public void testIsResultEligibleForHopscot
    final HopscotchData data = buildValidD
            .setClosingTimeFrame("4-to-6-m
```

Copy Reference	⌥⇧⌘C
Paste	⌘V
Paste from History...	⇧⌘V
Paste without Formatting	⌥⇧⌘V
Column Selection Mode	⇧⌘8
Find Usages	⌥F7
Refactor	▸
Folding	▸
Analyze	▸
Go To	▸
Generate...	^N
Recompile '...atorHelperTest.java'	⇧⌘F9
Run 'testIsResultEligib...()'	^⇧F10
Debug 'testIsResultEligib...()'	^⇧F9
Run 'testIsResultEligib...()' with Coverage	
Create 'testIsResultEligib...()'...	
Reveal in Finder	

Rename...	⇧F6
Change Signature...	⌘F6
Type Migration...	⇧⌘F6
Make Static...	
Convert To Instance Method...	
Move...	F6
Copy...	F5
Safe Delete...	⌘⌦
Extract	▸
Inline...	⌥⌘N
Find and Replace Code Duplicates...	
Invert Boolean...	

```java
public class AttributeEvaluatorHelperTest {

    private static final Map<String, String> VALID_ATTRIBUTE_MAP = ImmutableMap.of(
            VINTAGE, VINTAGE_YEAR,
            PURCHASED, PURCHASED_YEAR
    );

    private final AttributeEvaluatorHelper subject

    @Test
    public void testIsResultEligibleForHopscotch_r
        final HopscotchData data = buildValidData(
        final boolean result = subject.isCaseEligi

        Assert.assertTrue(result);
    }

    @Test
    public void testIsResultEligibleForHopscotch_r
        final HopscotchData data = buildValidData(
                .setHasScrubbingDiscrepancies(Bool
        final boolean result = subject.isCaseEligi

        Assert.assertFalse(result);
    }

    @Test
    public void testIsResultEligibleForHopscotch_r
        final HopscotchData data = buildValidData(
                .setHasScrubbingDiscrepancies(null
        final boolean result = subject.isCaseEligi

        Assert.assertFalse(result);
    }

    @Test
    public void testIsResultEligibleForHopscotch_r
        final HopscotchData data = buildValidData(
                .setClosingTimeFrame("4-to-6-months");
        final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP);
```

Change Signature

Visibility: public

Return type: boolean

Name: isCaseEligibleForHopscotch

Parameters | Exceptions

```
HopscotchData       data
Map<String, String> ruleAttributes
```

Type: ShortTrial

Name: overrides

Default value:

Use Any Var

Method calls: Modify | Delegate via overloading method

Signature Preview

```java
public boolean isCaseEligibleForHopscotch(HopscotchData data,
                                          Map<String, String> ruleAttributes,
                                          ShortTrial overrides)
```

Cancel | Preview | Refactor

```
rHopscotch_returnsTrue_whenExpected() {
ldValidData();
.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP, validOverrides);
```

Create local variable 'validOverrides'
Create field 'validOverrides' in 'AttributeEvaluatorHe
Create parameter 'validOverrides'
Rename reference
Expand boolean declaration to 'if else'
Split into declaration and assignment

```
rHopscotch_returnsFalse_whenDiscrepanciesAreFound() {
ldValidData()
pancies(Boolean.TRUE);
.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP, validOverrides);
```

```
rHopscotch_returnsFalse_whenDiscrepanciesDataIsNotPresent() {
ldValidData()
pancies(null);
.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP, validOverrides);
```

Cove

```java
private ShortTrial validOverrides = new ShortTrial().setRuleSet(RuleSet.HOPSCOTCH);

private final AttributeEvaluatorHelper subject = new AttributeEvaluatorHelper();

@Test
public void testIsResultEligibleForHopscotch_returnsTrue_whenExpected() {
    final HopscotchData data = buildValidData();
    final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP, validOverrides);

    Assert.assertTrue(result);
}

@Test
public void testIsResultEligibleForHopscotch_returnsFalse_whenDiscrepanciesAreFound() {
    final HopscotchData data = buildValidData()
            .setHasScrubbingDiscrepancies(Boolean.TRUE);
    final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP, validOverrides);

    Assert.assertFalse(result);
}

@Test
public void testIsResultEligibleForHopscotch_returnsFalse_whenDiscrepanciesDataIsNotPresent() {
    final HopscotchData data = buildValidData()
            .setHasScrubbingDiscrepancies(null);
    final boolean result = subject.isCaseEligibleForHopscotch(data, VALID_ATTRIBUTE_MAP, validOverrides);
```

Feel empowered to bend the rules a bit…

…It’s the results that matter!

- Testable code
- High coverage
- Confidence in your tests

With TDD your unit tests will…

BE EASY TO CHANGE/GROW

SERVE AS DOCUMENTATION

GUARD FUTURE DEVS AGAINST REGRESSIONS.

BDD

Behavior Driven Development

What is BDD
(Behavior Driven Development)

- An agile methodology to write the specification of a system in a shared, easy to understand language.
- The use of a set of tools/languages/practices to write that specification and/or automate the verification of the specification against a system.
- BDD only cares about results, not how you achieve them, so it can help make sure time is focused on building the right thing.
- ATDD (Acceptance Tests Driven Development) can be considered a form of BDD
- The core of BDD is specifying three things:
  - Given, When, Then

Some common tools for BDD

- JBehave:

  https://jbehave.org/
- Cucumber

  https://cucumber.io/
- PyTest

  https://testautomationu.applitools.com/behavior-driven-python-with-pytest-bdd/

But don’t over do it…

From the Agile Alliance on BDD:

“The use of BDD requires no particular tools or programming languages, and is primarily a conceptual approach; to make it a purely technical practice or one that hinges on specific tooling would be to miss the point altogether”

Let the fight begin!

Let’s evaluate the two approaches over a set of dimensions!

Speed:
Which one is faster to use?

TDD

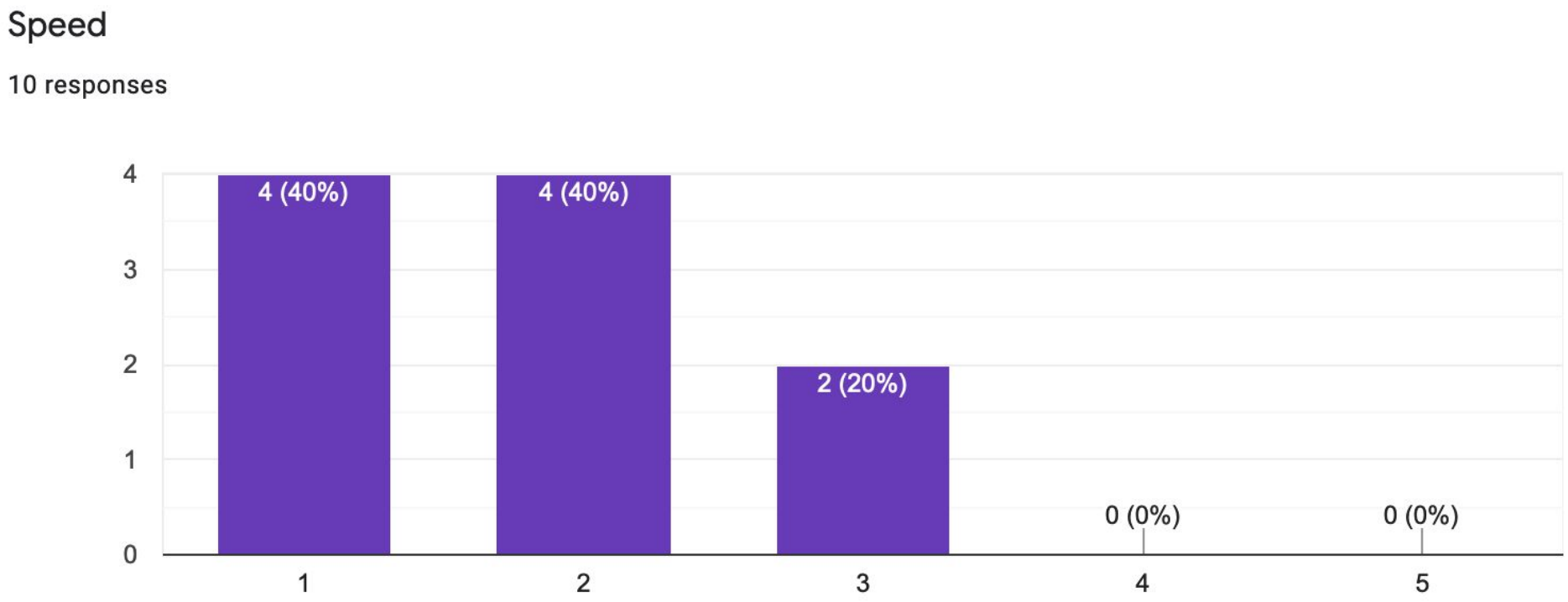
- Gets developed with the code
- Runs blazingly fast

BDD

- Written and negotiated in a group
- Is slow to execute

Cast your vote in the latest link!

Speed:
Which one is faster to use?



Debugging:
Which one helps root-cause problems better

TDD

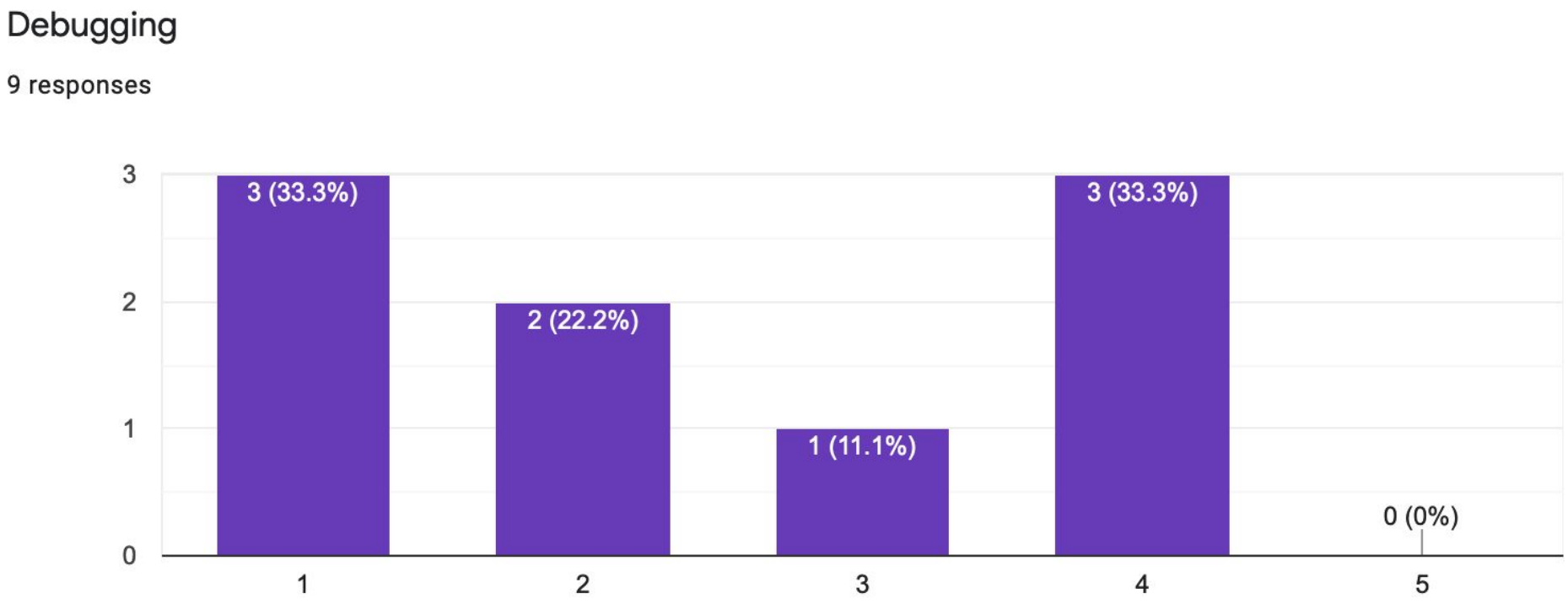
- Pin-points broken assumptions in code
- Runs within the IDE for ease of debugging

BDD

- Only tells about a broken contract, few hints as to why.
- Requires code investigation to understand failure or a connection to an open debugging port in a running server.

Cast your vote in the latest link!

Debugging:
Which one helps root-cause problems better



Coverage:
Which one gives you more code coverage

TDD

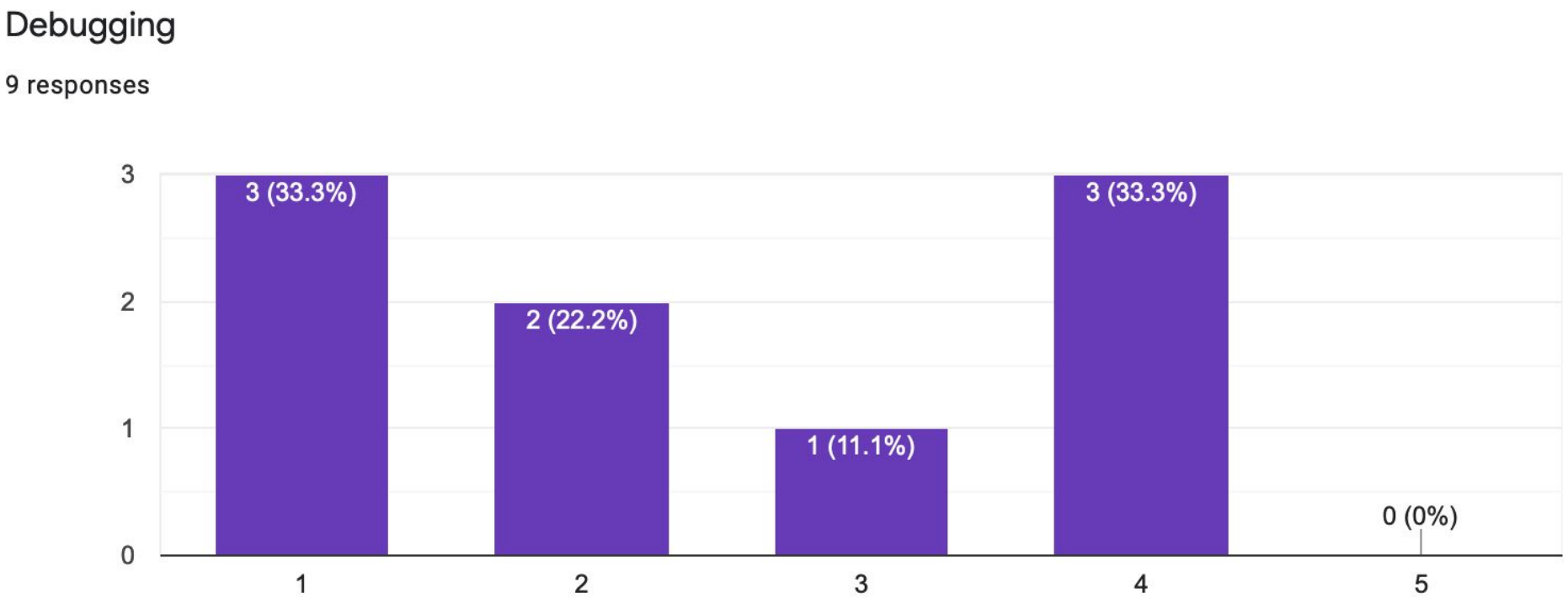
- 100% Coverage is possible but often absurd
- To reach high coverage levels you really need to be faithful to TDD and write tons of tests.

BDD

- Coverage may be harder to measure, but very likely it will be higher as many more lines are hit with a single test.

Cast your vote in the latest link!

Coverage:
Which one gives you more code coverage



Approach:
Which one brings a better approach to tests?

TDD

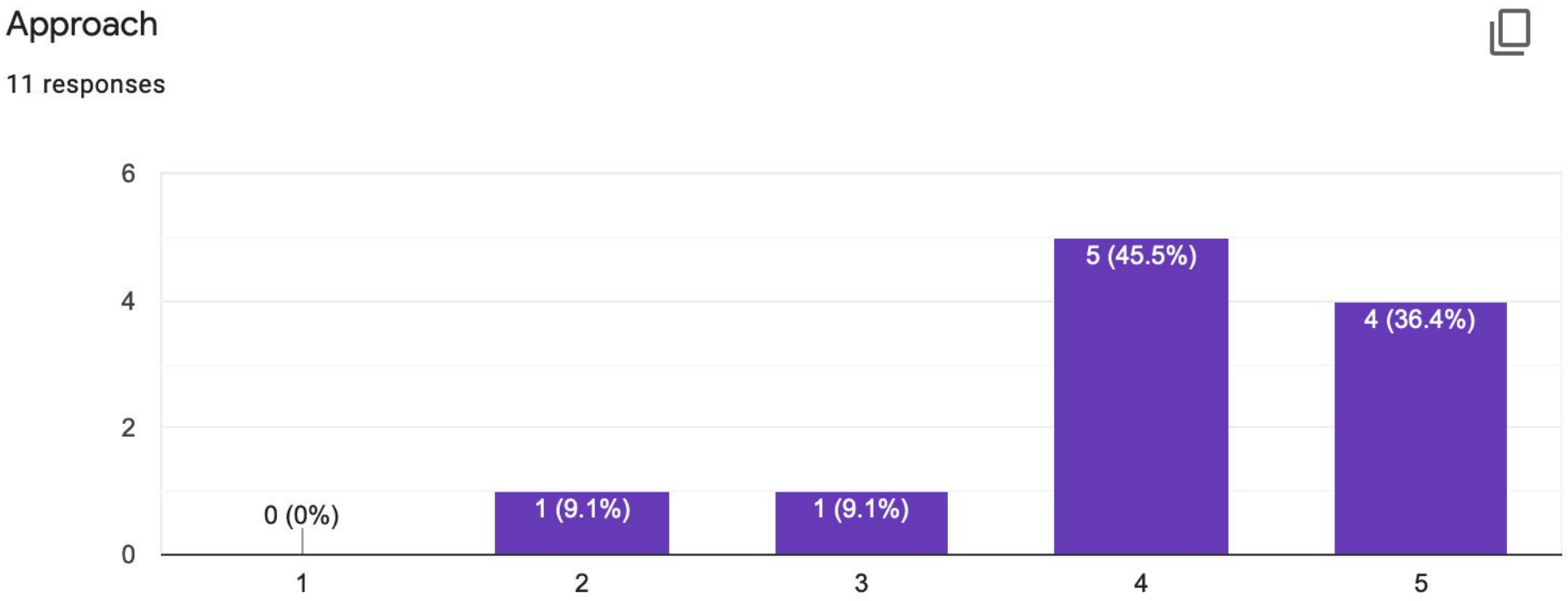
- is focused on code
- concentrates on building things the right way
- is very specific to the technology and hard for non-devs to follow

BDD

- is focused on a readable specification to represent the end user needs.
- concentrates on delivering value for your users

Cast your vote in the latest link!

Approach:
Which one brings a better approach to tests?



Maintainability:
Which one is easier to change and grow

TDD

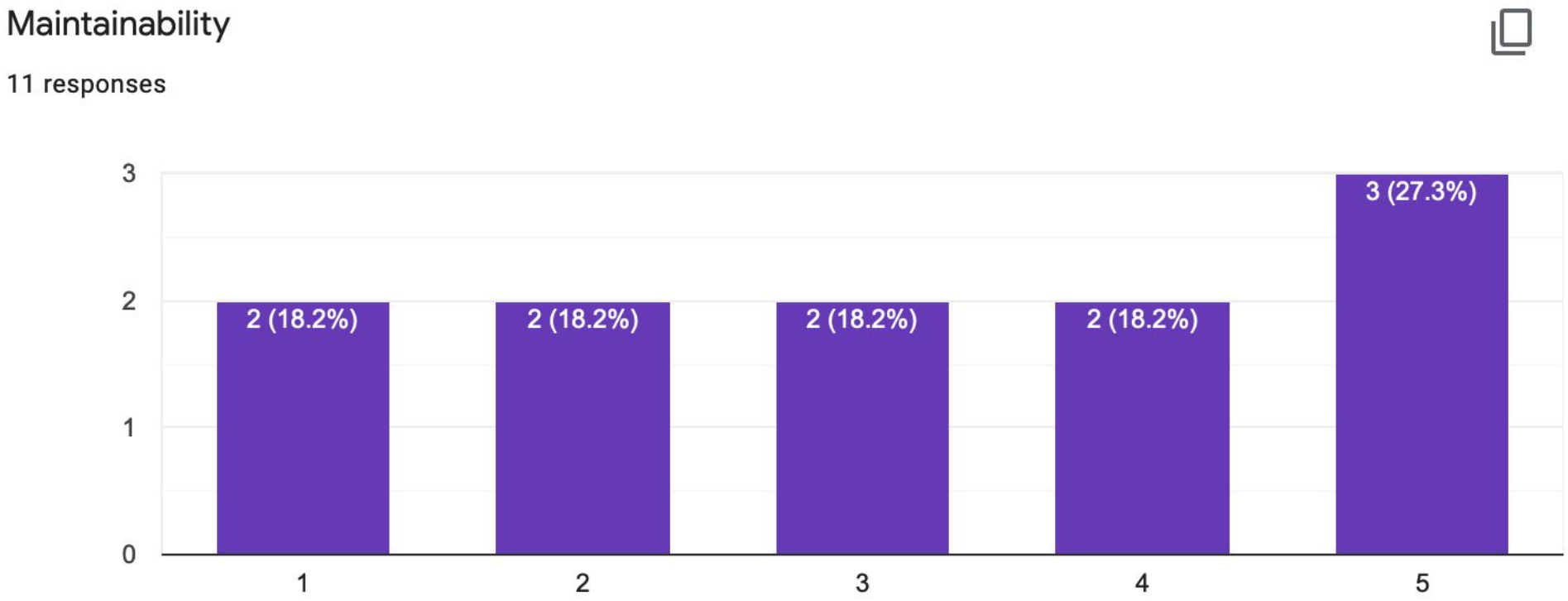
- Need to leverage the IDE to be effective at evolving tests.
- Code and tests are closely tied together, so big changes to the code will likely mean big changes to the tests.

BDD

- Tends to be independent of the underlying code
- Requires maintaining tooling around the tests.
- The system under test can be fully re-written and the contract remain unchanged and valid

Cast your vote in the latest link!

Maintainability:
Which one is easier to change and grow



And the winner is…

Oh-oh, I sense a plot twist coming.

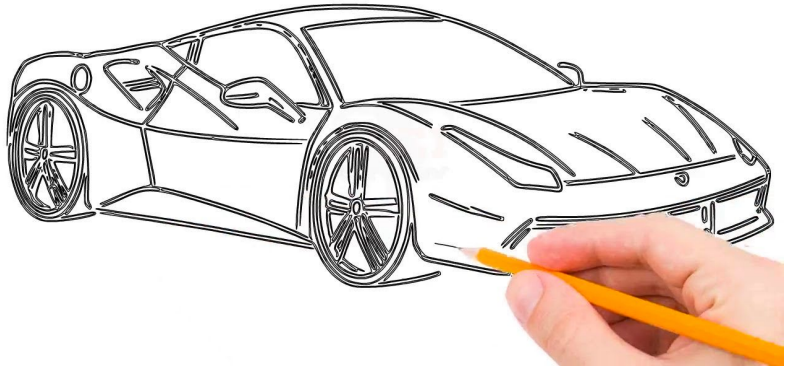
Use both!

Get the benefits of both by leveraging both techniques. Apply TDD to your code (and services), with a hint of BDD for expressivity, and use BDD to define your system's behavior and build (or at least inform) your automated integration tests (and end-to-end tests) based on that!

Why you need both:

TDD

- Helps you build the system the right way



BDD

- Helps you build the right system

Q&A

Some links for reference, in no particular order:

- https://martinfowler.com/articles/practical-test-pyramid.html
- https://www.jrebel.com/blog/when-to-use-test-driven-development
- https://daedtech.com/5-things-ive-learned-in-20-years-of-programming/
- https://www.agilealliance.org/glossary/bdd
- https://www.solutionsiq.com/resource/blog-post/behavior-driven-development-simplifying-the-complex-problem-space/
- https://cucumber.io/blog/bdd/where_should_you_use_bdd/

For interactive presentations:

- https://www.mentimeter.com/

Bonus: Discussion.

You are given ownership of a legacy service that has little to no testing whatsoever. Where would you start?

Some scenarios:

- You are asked to fix a bug with the utmost priority
- You are tasked with adding a business-critical feature
- You are hired to lead a small team of coders, as this service has been causing high attrition due to constant problems and alarms.