TDD with Spring Boot

Practical Session

Aleksandr Barmin

Aleksandr Barmin

Chief Software Engineer I

- More than 10 years in software engineering
- PhD in computer science
- AWS Solution Architect Associate
- AWS Developer Associate
- @AABarmin - Twitter, GitHub

Some theory about TDD

TDD in any book

Is not the same as TDD in the real life

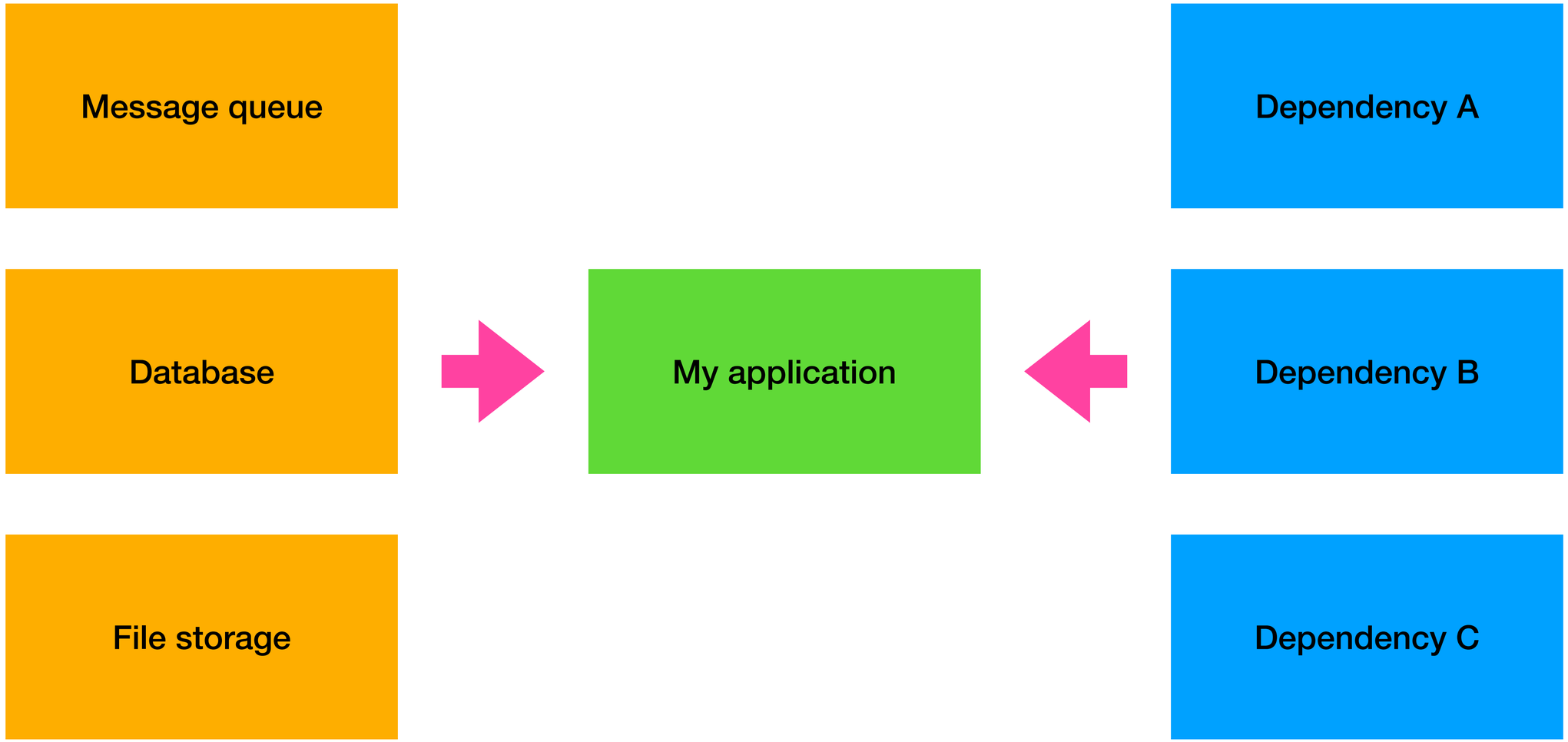
```
class MyClass {
  int sum(int a, int b) {
    return a + b;
  }
}
```

```
class MyClassTest {
  MyClass uut = new MyClass();

  @Test
  void superTest() {
    int result = uut.sum(1, 2);
    assertEquals(result, 3);
  }
}
```

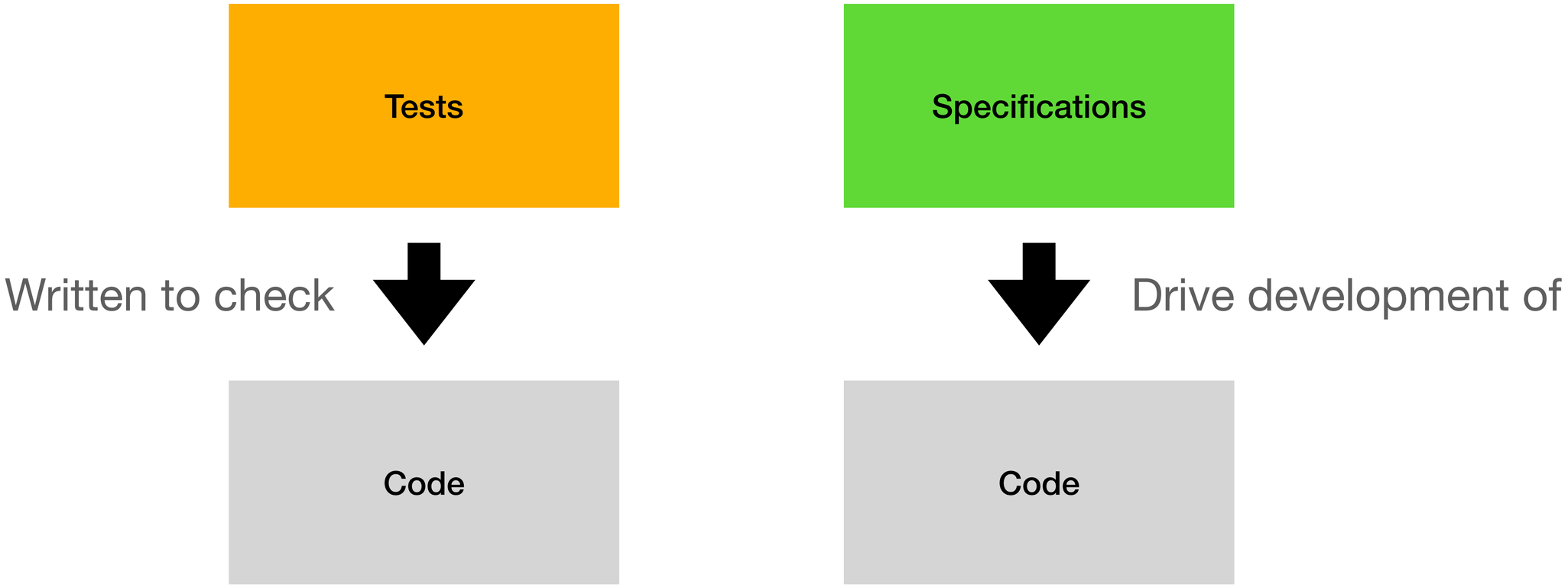
TDD in any book

Is not the same as TDD in the real life



TDD in any book

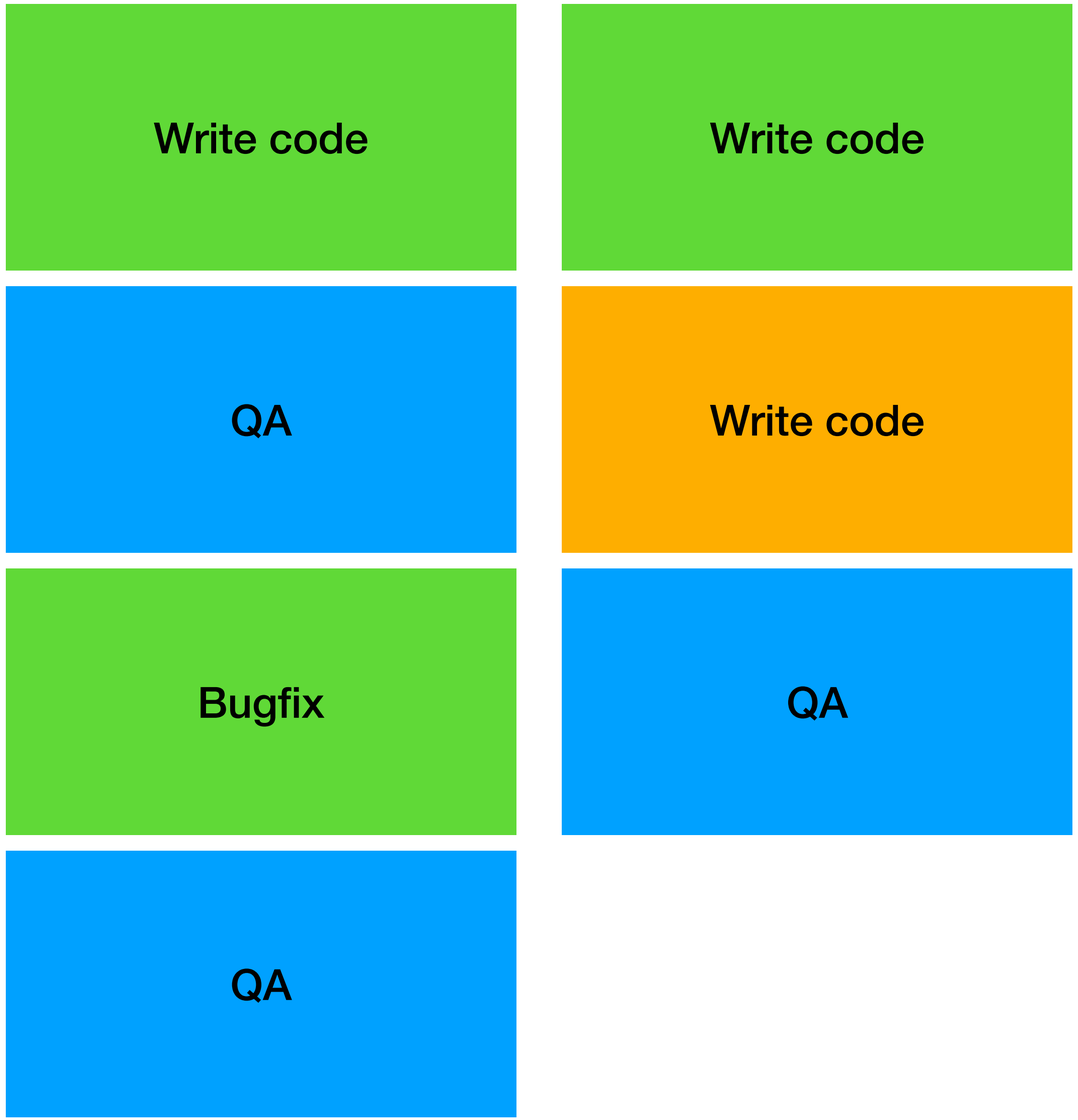
Is not the same as TDD in the real life



Why tests?

It takes twice longer

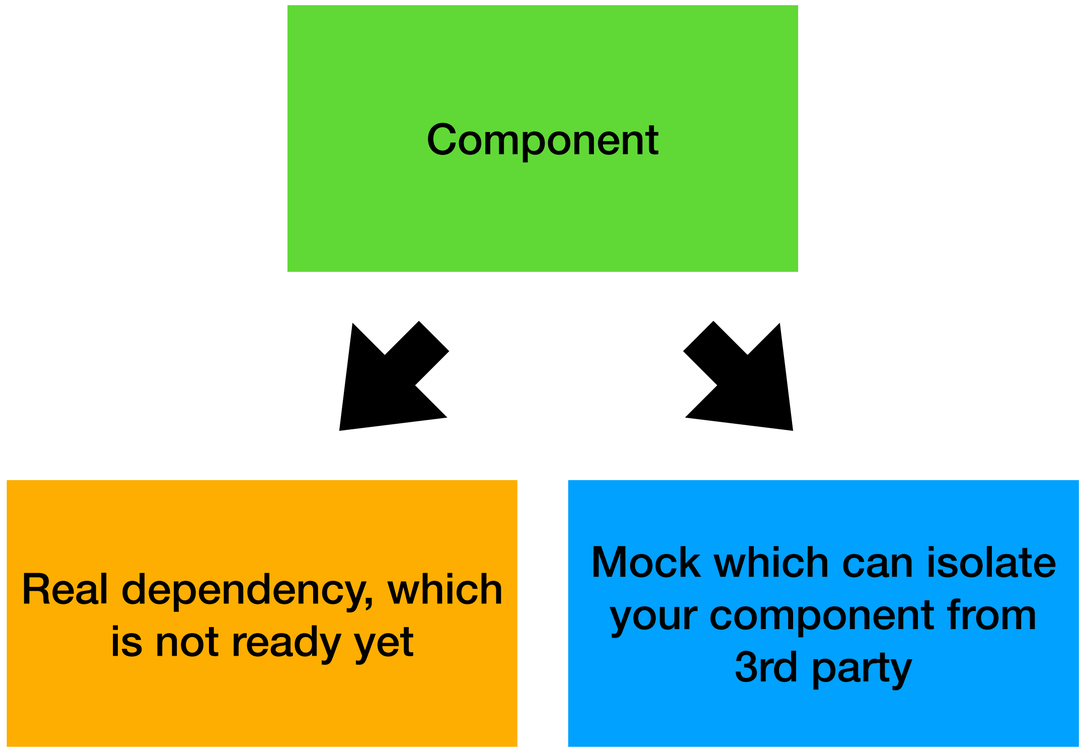
- Get quicker feedback
- Add features quicker
- Clean, readable and stable code
- Easy to refactor
- Reliability and confidence



Why before?

Why not feature-first?

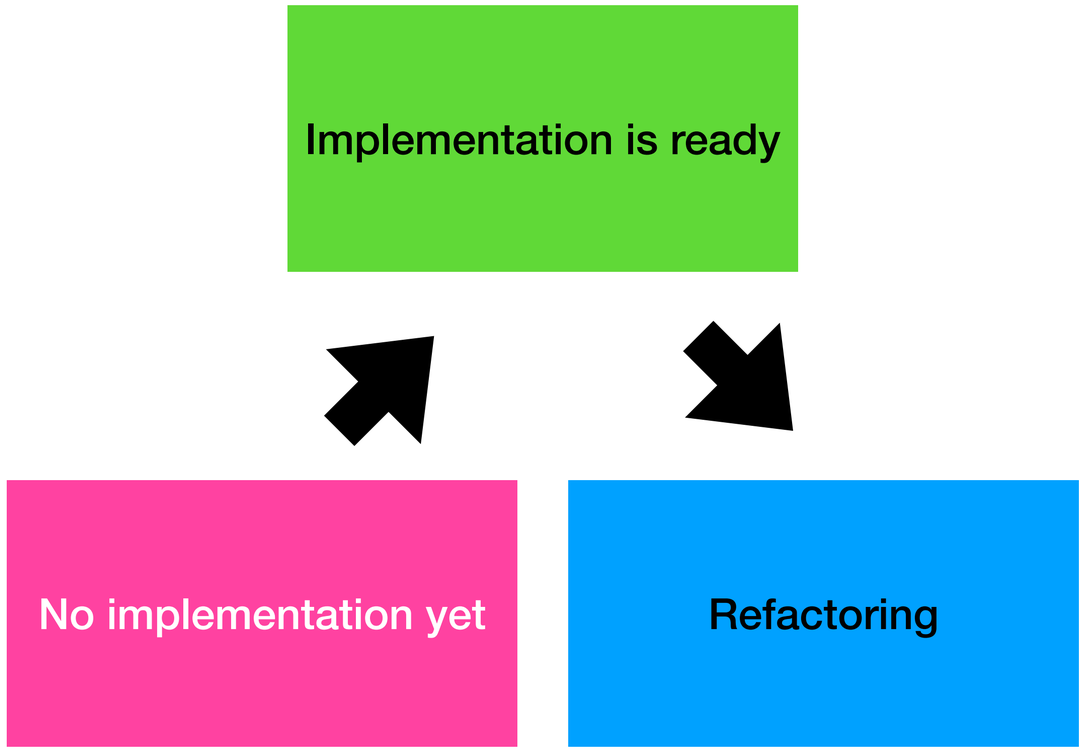
- Is it checking what it should?
- Is test fails first?
- Need to break the code to write a test
- Decomposition tool
- Architecture tool



TDD

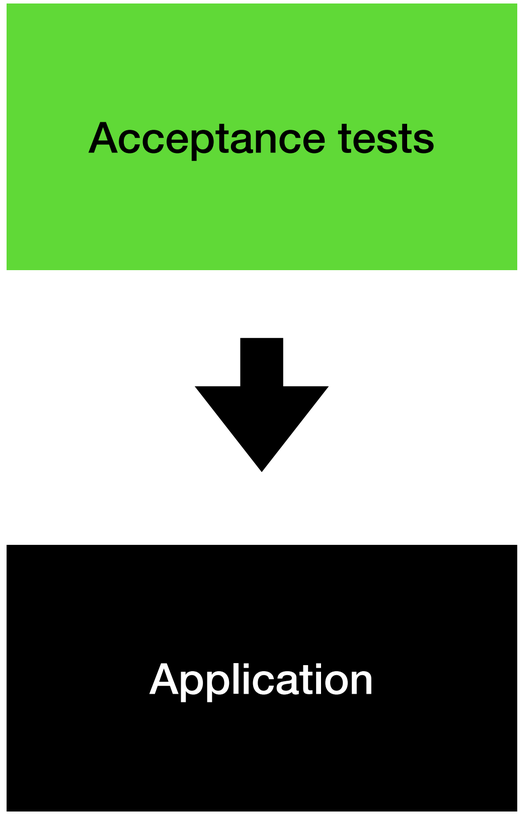
Test Driven Development

- Helps to split big task into subtasks
- Define structure of code and components
- Establish boundaries of responsibilites of components



Acceptance tests

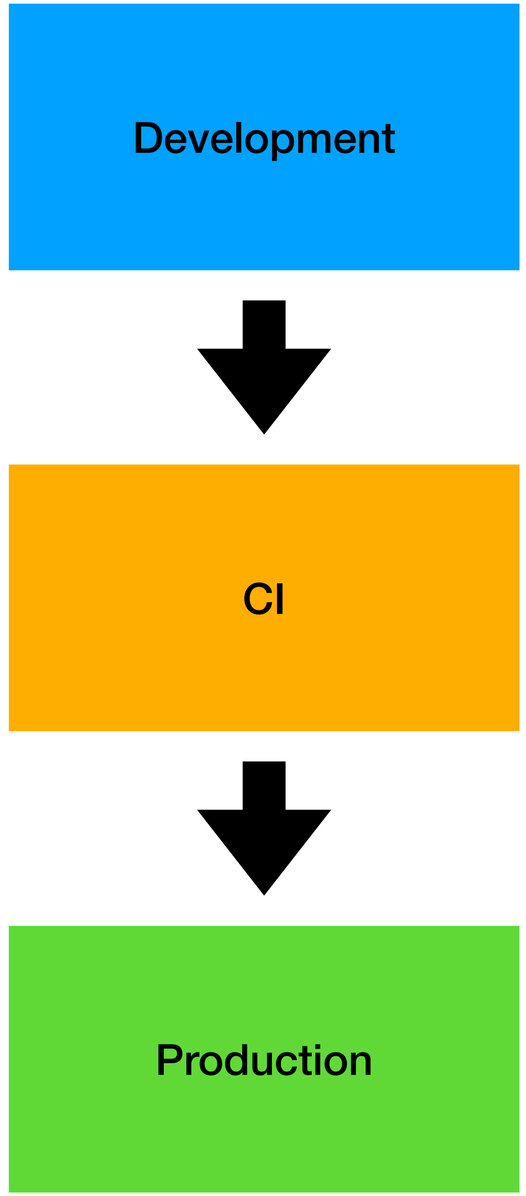
- Written from user-perspective (BDD with Gherkin as an ex.)
- Use system as a black box
- Expensive, slow and may not provide 100% coverage
- Is my test correct - can it be used as a smoke test?



CI

TDD allows to improve the coverage

- Can I deploy to prod if my CI pipeline is green?
  - Yes - have to have good coverage
  - No - why are you writing tests?



Fancy practice

Example application

Create a new bank account (happy path)

- Given an **applicant** with email test@test.com is new to bank
- When an **applicant** provides test@test.com email
- And an **applicant** provides a valid last name
- And an **applicant** creates an account
- Then the **app** should return **ID** of a created record

Example application

Create a new bank account (unhappy path)

- Given an **applicant** with email test@test.com known to bank
- When an **applicant** provides test@test.com email
- And an **applicant** provides a valid last name
- And an **applicant** creates an account
- Then the **app** should return 409 response code

Example application

Get a record by email (happy path)

- Given an **applicant** with email test@test.com known to bank
- When an **applicant** provides test@test.com email
- And an **applicant** wants to get information about account
- Then the **app** should return the record
- And response should contain email test@test.com

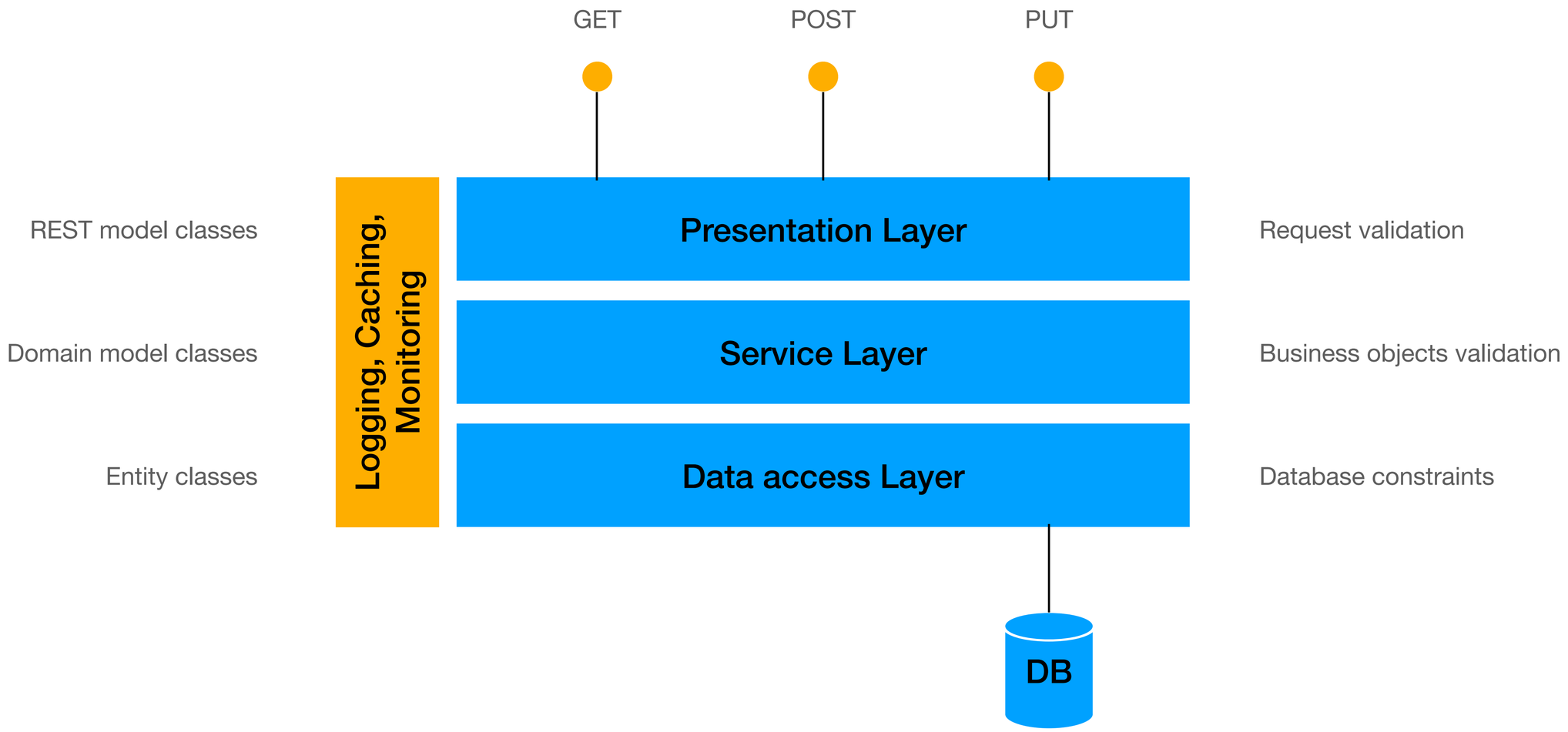
Example application

Get a record by ID (unhappy path)

- Given an **applicant** with email test@test.com is new to bank
- When an **applicant** provides test@test.com email
- And an **applicant** wants to get information about account
- Then the **app** should return 404 response code

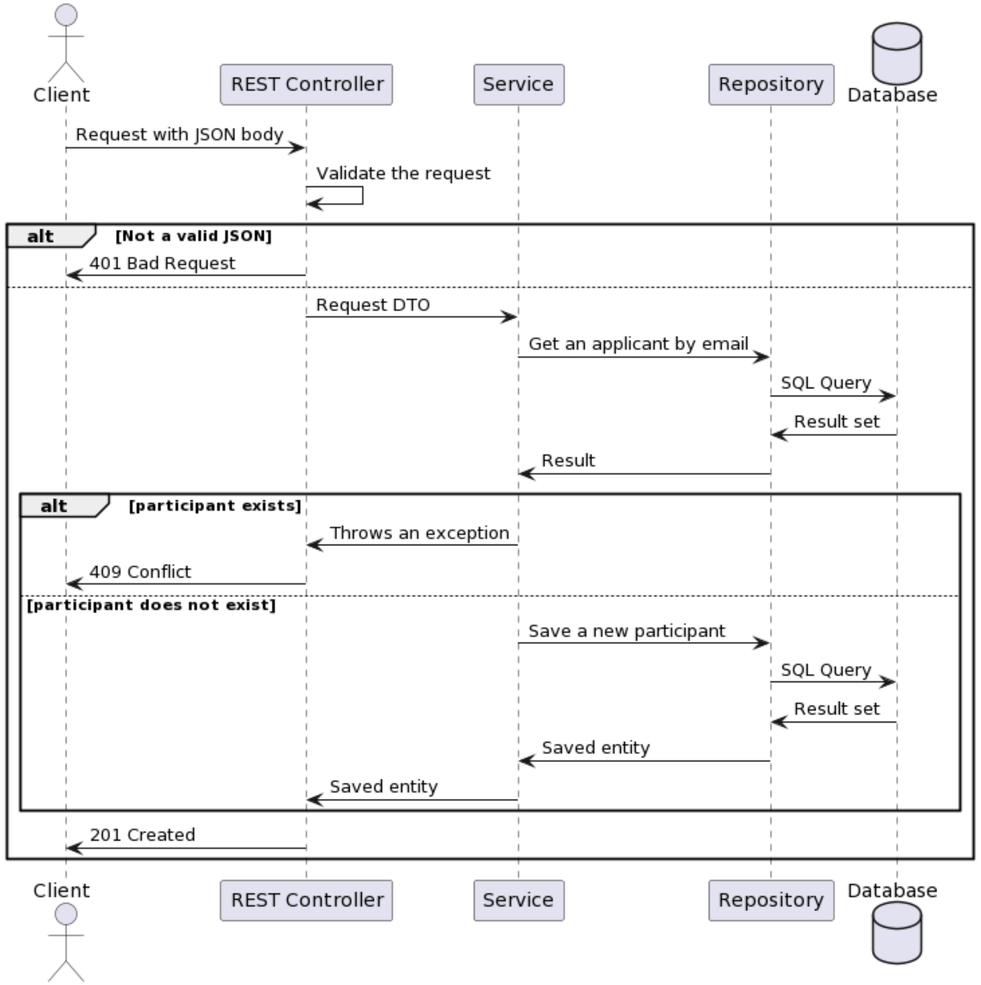
Application’s architecture

High level



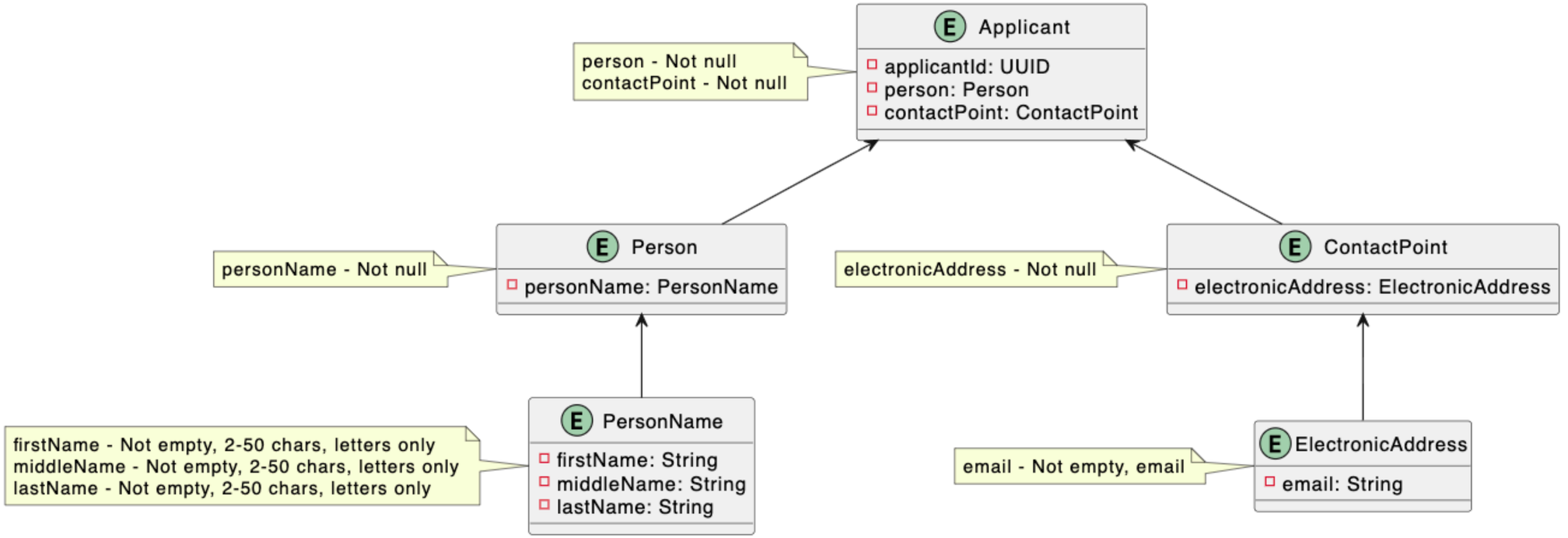
Implementation

Create an applicant sequence diagram



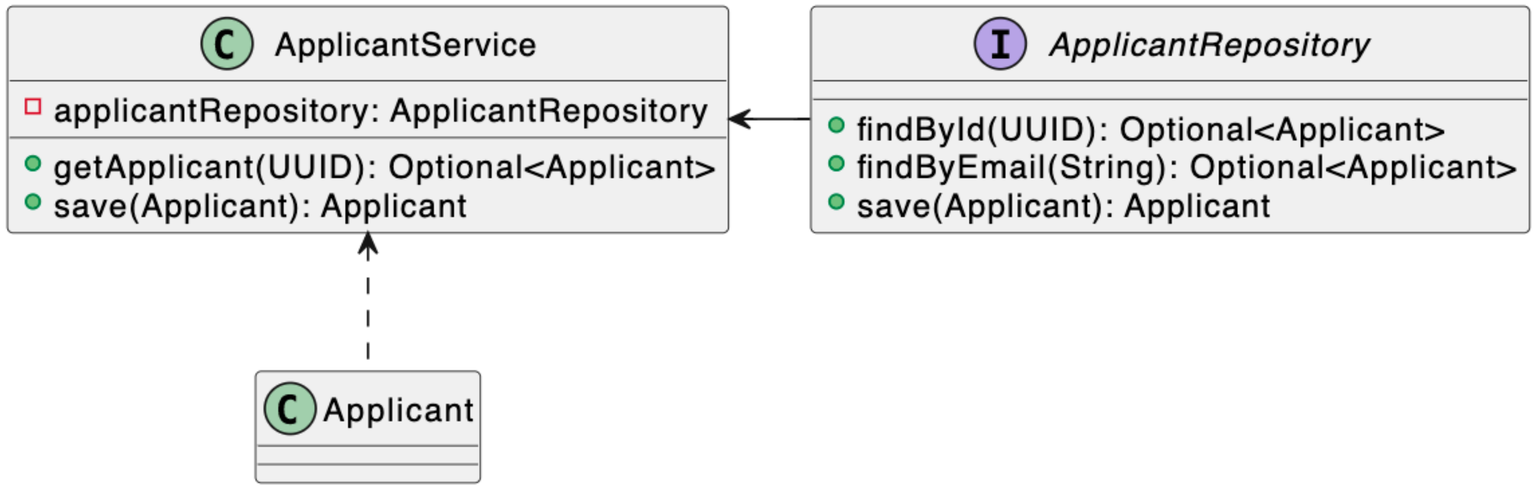
Implementation

Domain layer



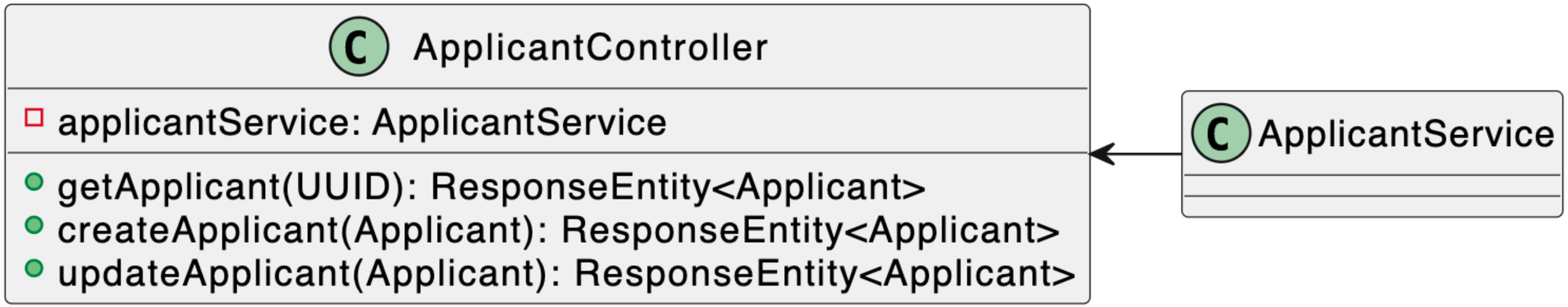
Implementation

Service layer



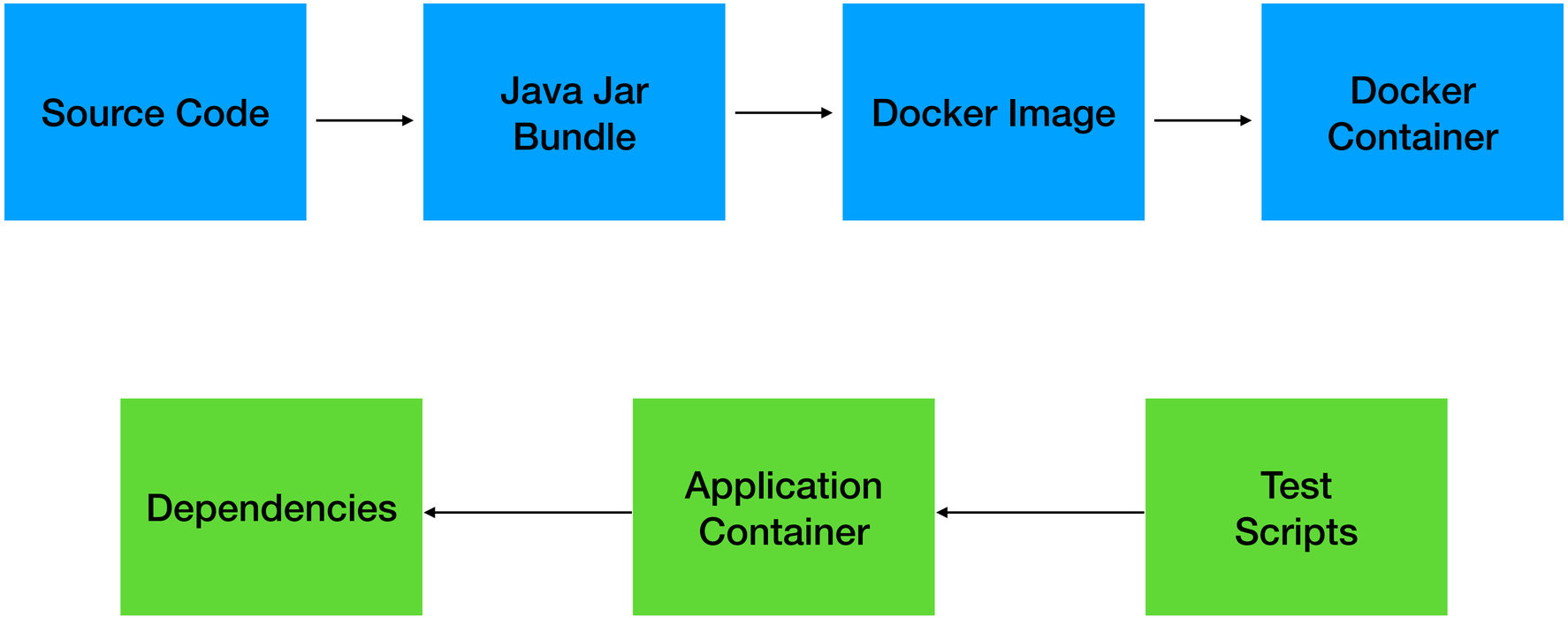
Implementation

Presentation layer - REST endpoints



End-to-end testing

With Docker and Cucumber



Summary

What we learned

- Tests are live documentation and specification for the code
- Be familiar with tools available for your stack - there are many tools which make your life easier
- Spring Framework is not only good framework for development but also for testing

Thank you

- https://github.com/aabarmin
- https://twitter.com/AABarmin
- https://github.com/aabarmin/epam-spring-boot-tdd-2022