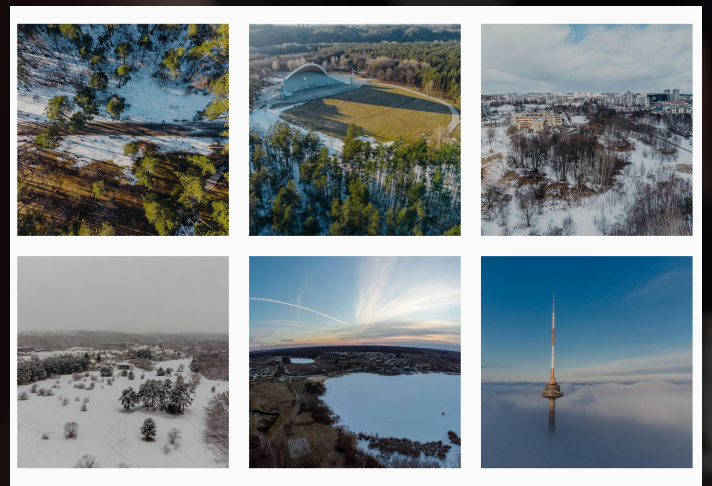
metasite

Testcontainers

Mindaugas Urbontaitis

2022-06-30



Hello 👋

- Java developer
  - Team 8
- Outdoor activities
  - My biggest 🚴 achievement Paris - Brest - Paris 2019
  - My 📸 365 days challenge

The problem

Video rental system which needs to support multiple databases

- MySQL
- MongoDB

Source Video rental system example

The project specification

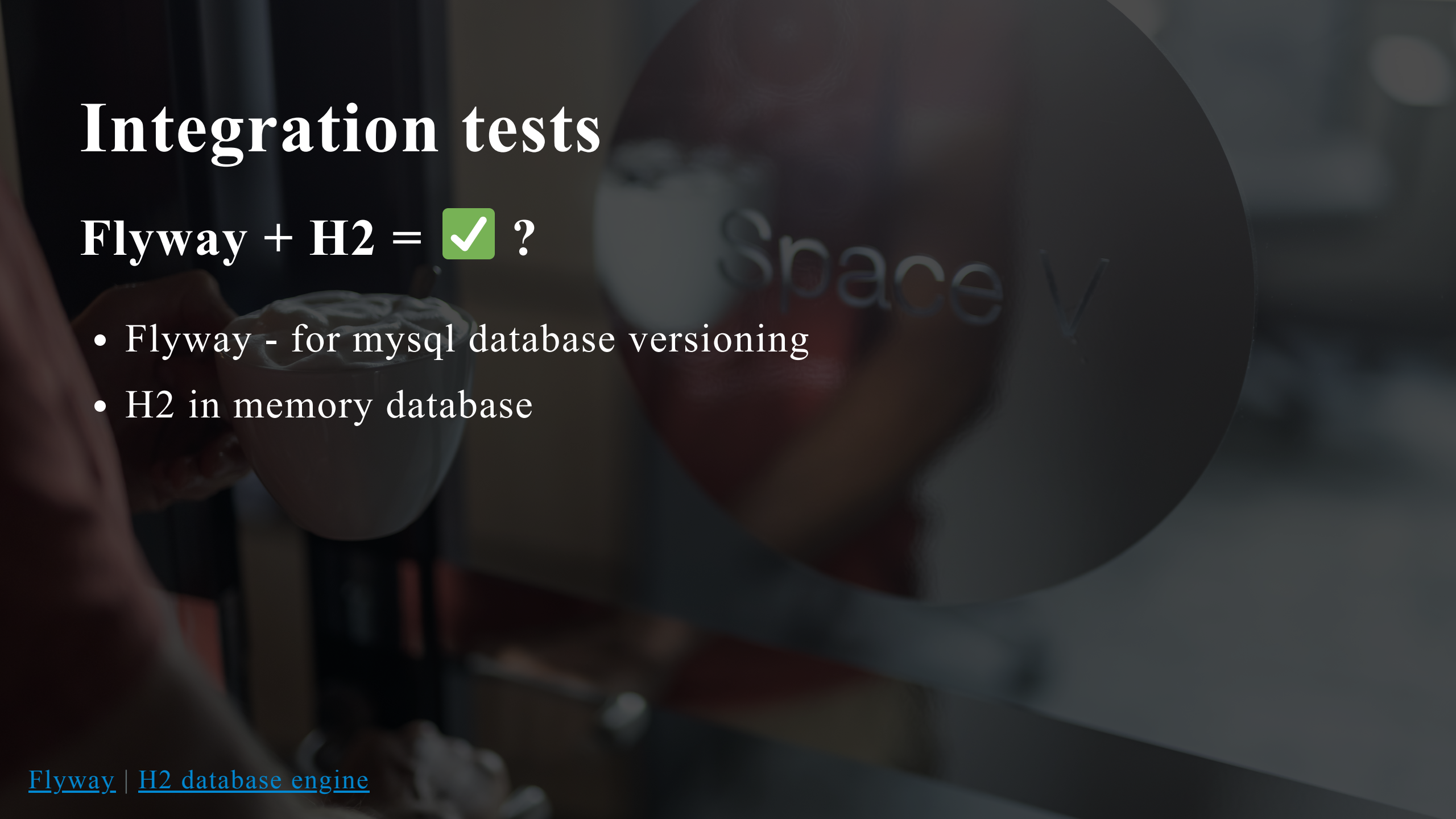
```
def "should get film details"() {
    given: "module has film Rambo"
    when: "client asks for details of Rambo"
    then: "module returns details of Rambo"
  }

  def "when no film found"() {
    when: "client asks for details of Rambo"
    then: "module does not return rambo"
  }

  def "should get films"() {
    given: "module has two films"
    when: "client asks for films"
    then: "module returns both films"
  }
```

Unit testing

- Fast, but no:
  - Database schema check
  - Check if code works as expected when runs in the production environment
  - Multiple database verification



Integration tests

Flyway + H2 = ✅ ?

- Flyway - for mysql database versioning
- H2 in memory database

Flyway | H2 database engine

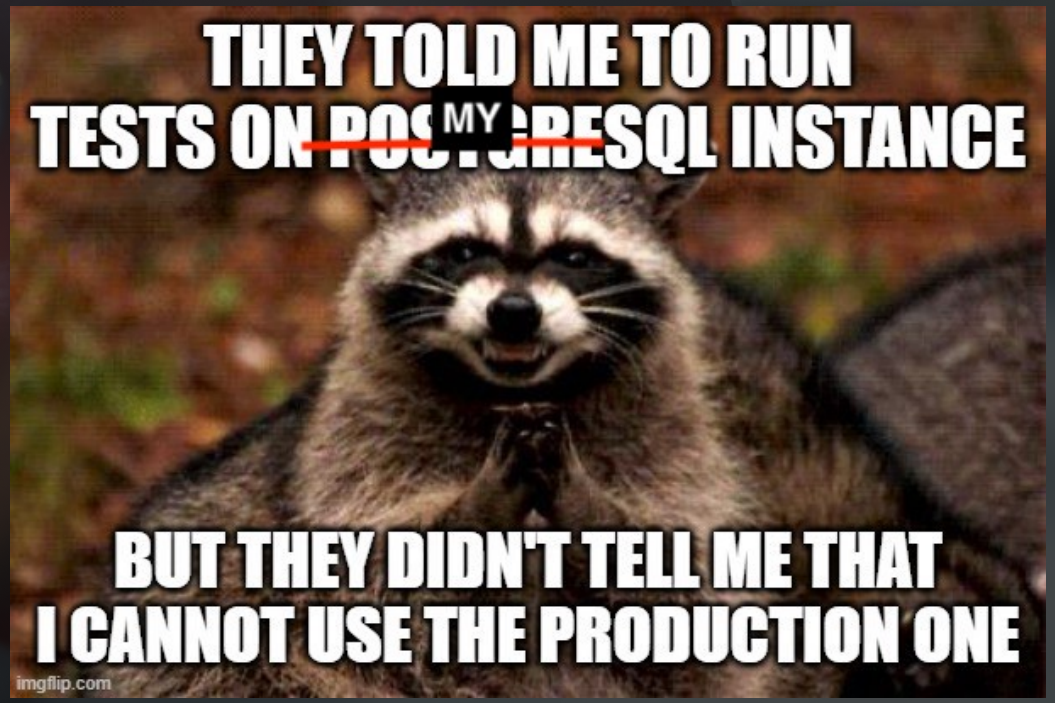
H2 database demo #1

```
dependencies {
  implementation 'org.flywaydb:flyway-core'
  implementation 'org.flywaydb:flyway-mysql'

  testImplementation 'com.h2database:h2'
}
```

Production DB

- Expensive
- High chance of failing tests
  - when tests are executed by
    - CI
    - developers



Source Image

What it is Testcontainers?

Testcontainers

Source Testcontainers

Testcontainers

- Data access layer integration tests (e.g. @DataJpaTest)
- Application integration tests
- UI/Acceptance tests
- Allows to create your own custom container classes

Source Testcontainers

MySQL database demo #2

```
dependencyManagement {
    imports {
        mavenBom "org.testcontainers:testcontainers-bom:1.17.3"
    }
}

dependencies {
    testImplementation 'org.testcontainers:spock'
    testImplementation 'org.testcontainers:mysql'
}
```

MySQL database demo #3

Embedded MySQL:

```
dependencies {
  testImplementation "org.springframework.cloud:spring-cloud-starter-bootstrap:3.1.3"
  testImplementation "com.playtika.testcontainers:embedded-mysql:2.2.0"
}
```

Configuration:
https://github.com/Playtika/testcontainers-spring-boot/tree/develop/embedded-mysql

MongoDB database demo #4

Embedded MongoDB:

```
dependencies {
    testImplementation "org.springframework.cloud:spring-cloud-starter-bootstrap:3.1.3"
    testImplementation "com.playtika.testcontainers:embedded-mongodb:2.2.0"
}
```

Configuration:
https://github.com/Playtika/testcontainers-spring-boot/tree/develop/embedded-mongodb

Flyway demo #5

Source image

Conclusion

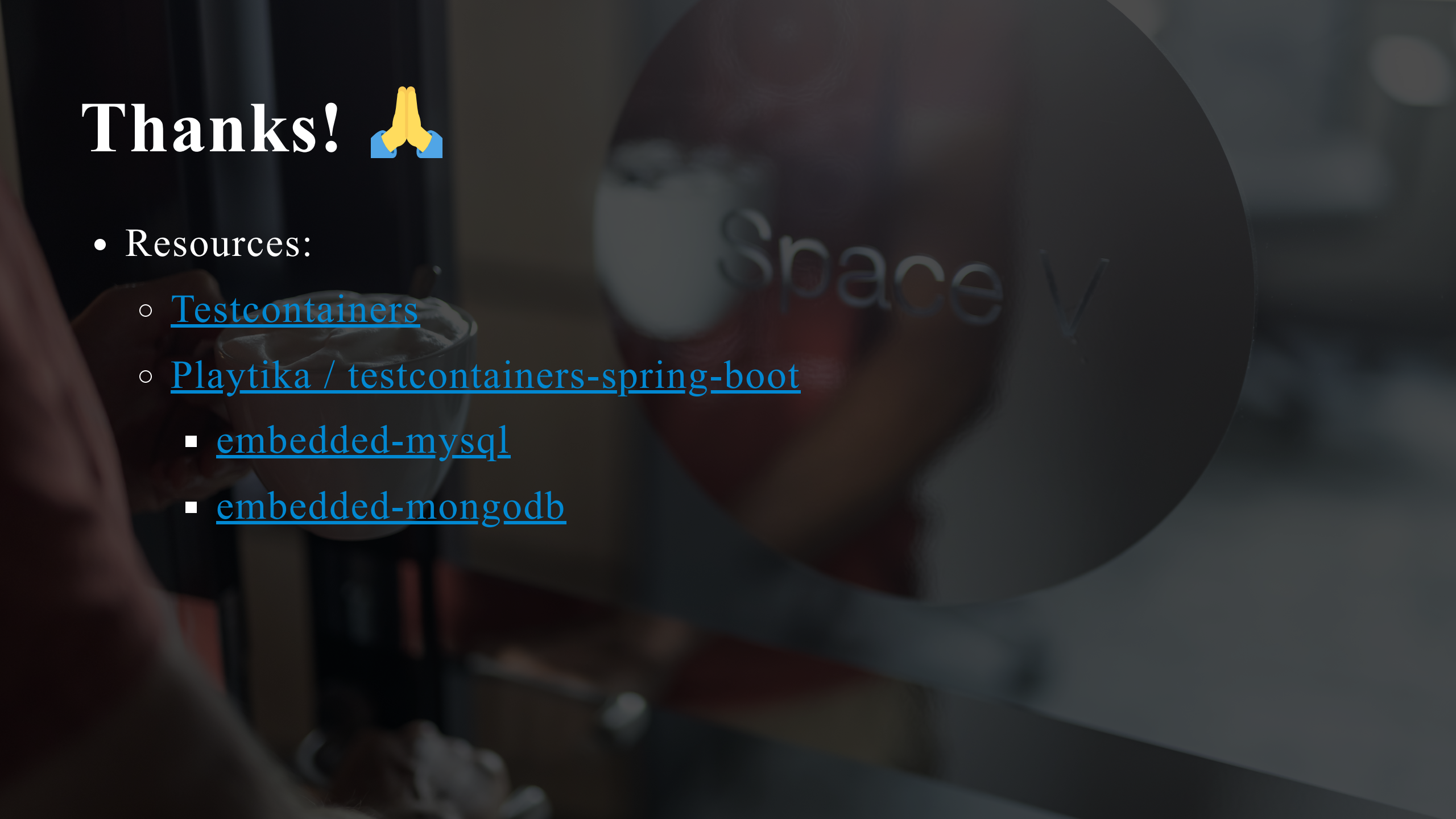
"Production" environment

Testcontainers:

```
- Easy to setup
- Configure test container image via connection URL
- host-less URI (///)
- Requires writing custom image substitutor or image name prefixed to be able to use Arm images
```

Playtika testcontainers wrapper:

```
- Docker image defined via bootstrap.properties
- Provides created container connection details like host URI, username and password
- Requires Spring Cloud Test dependency
```



Thanks! 🙏

- Resources:
  - Testcontainers
  - Playtika / testcontainers-spring-boot
    - embedded-mysql
    - embedded-mongodb