สยาม ชำนาญกิจ
SIAM CHAMNAN KIT

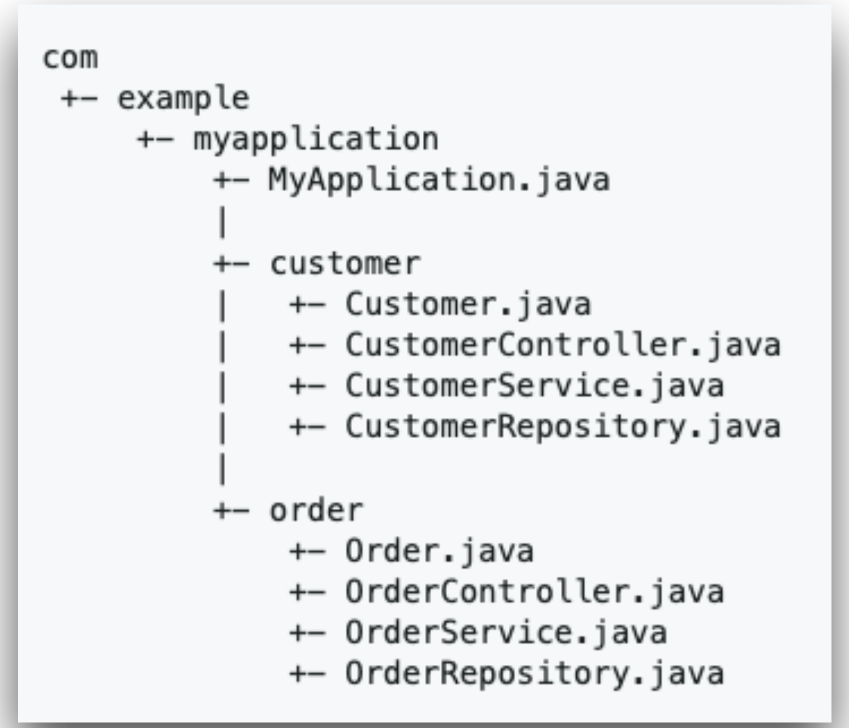
Week 04

Topics

Working with Database
Scope of components in Spring
Use cases of Mockito
Response vs Entity object
Q/A

Structure of project

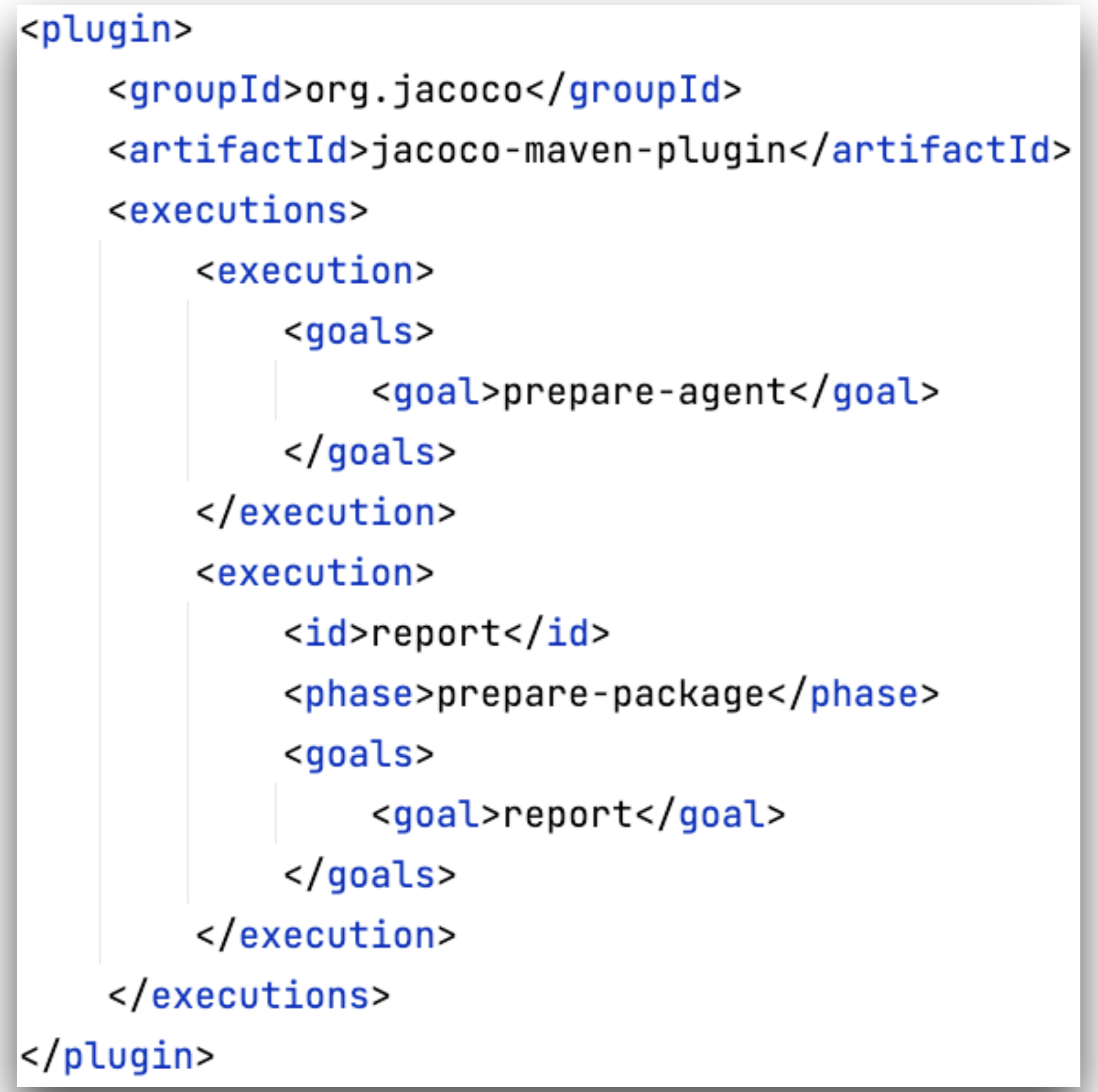
Structure of project



```
com
 +- example
     +- myapplication
         +- MyApplication.java
         |
         +- customer
         |   +- Customer.java
         |   +- CustomerController.java
         |   +- CustomerService.java
         |   +- CustomerRepository.java
         |
         +- order
             +- Order.java
             +- OrderController.java
             +- OrderService.java
             +- OrderRepository.java
```

https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#using.structuring-your-code

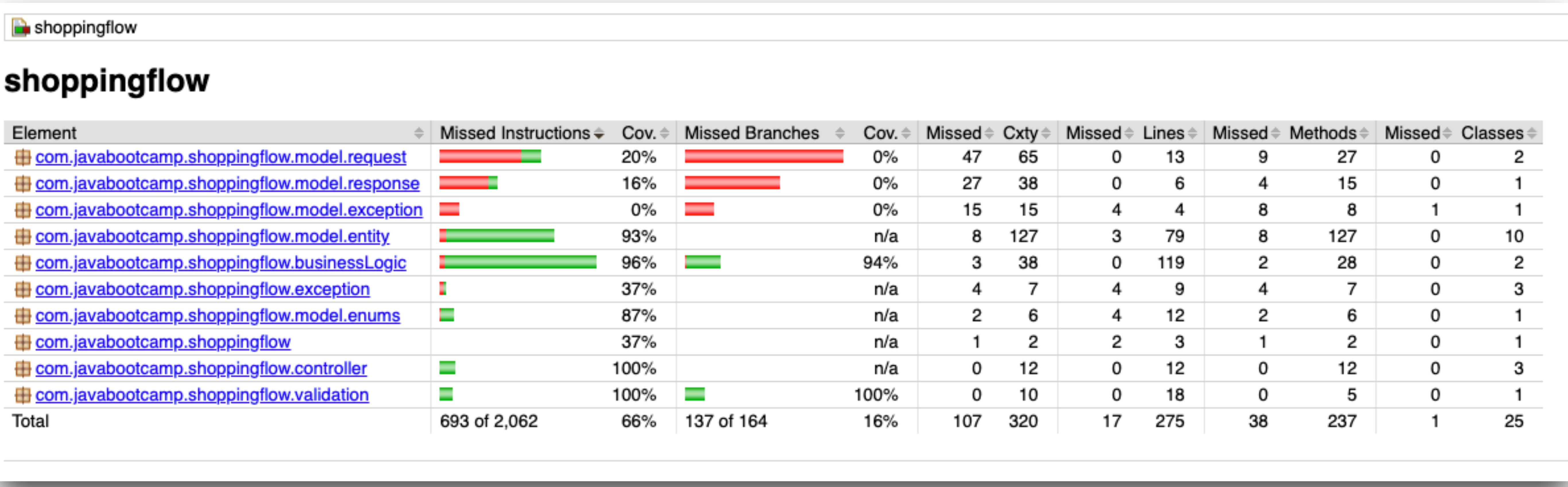
Code Coverage



```
<plugin>
    <groupId>org.jacoco</groupId>
    <artifactId>jacoco-maven-plugin</artifactId>
    <executions>
        <execution>
            <goals>
                <goal>prepare-agent</goal>
            </goals>
        </execution>
        <execution>
            <id>report</id>
            <phase>prepare-package</phase>
            <goals>
                <goal>report</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

https://www.jacoco.org/

Code Coverage



shoppingflow

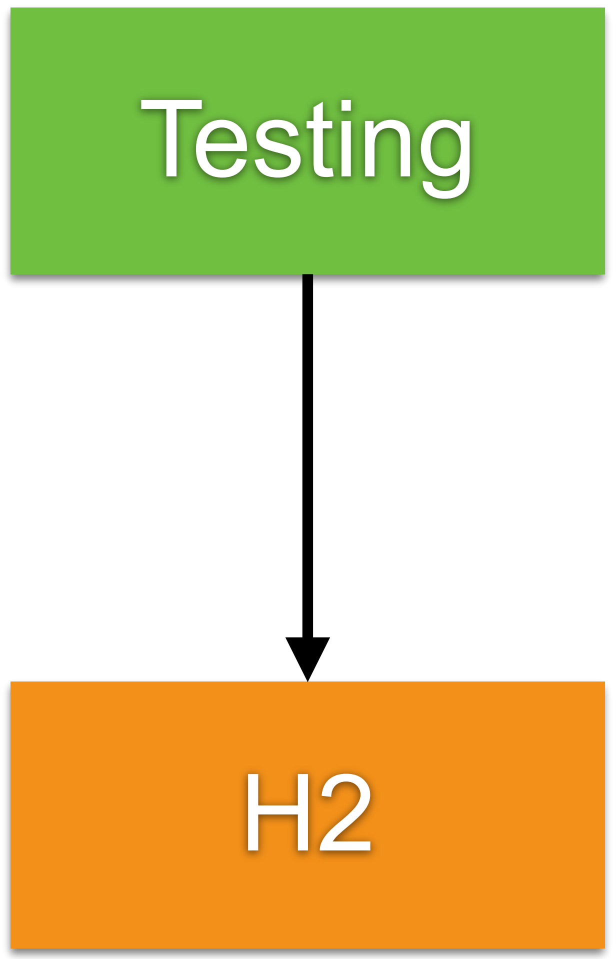
shoppingflow

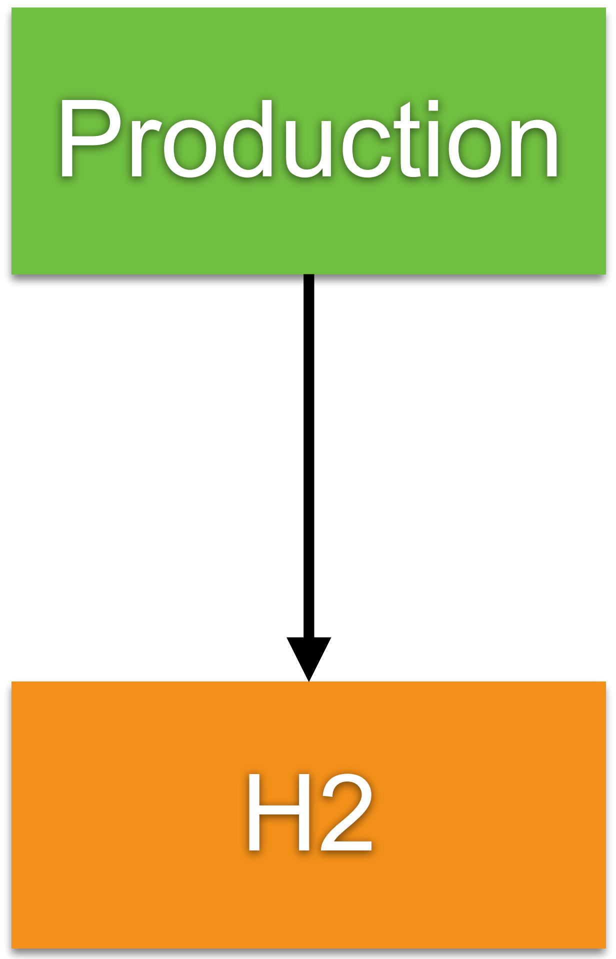
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.javabootcamp.shoppingflow.model.request		20%		0%	47	65	0	13	9	27	0	2
com.javabootcamp.shoppingflow.model.response		16%		0%	27	38	0	6	4	15	0	1
com.javabootcamp.shoppingflow.model.exception		0%		0%	15	15	4	4	8	8	1	1
com.javabootcamp.shoppingflow.model.entity		93%		n/a	8	127	3	79	8	127	0	10
com.javabootcamp.shoppingflow.businessLogic		96%		94%	3	38	0	119	2	28	0	2
com.javabootcamp.shoppingflow.exception		37%		n/a	4	7	4	9	4	7	0	3
com.javabootcamp.shoppingflow.model.enums		87%		n/a	2	6	4	12	2	6	0	1
com.javabootcamp.shoppingflow		37%		n/a	1	2	2	3	1	2	0	1
com.javabootcamp.shoppingflow.controller		100%		n/a	0	12	0	12	0	12	0	3
com.javabootcamp.shoppingflow.validation		100%		100%	0	10	0	18	0	5	0	1
Total	693 of 2,062	66%	137 of 164	16%	107	320	17	275	38	237	1	25

https://www.jacoco.org/

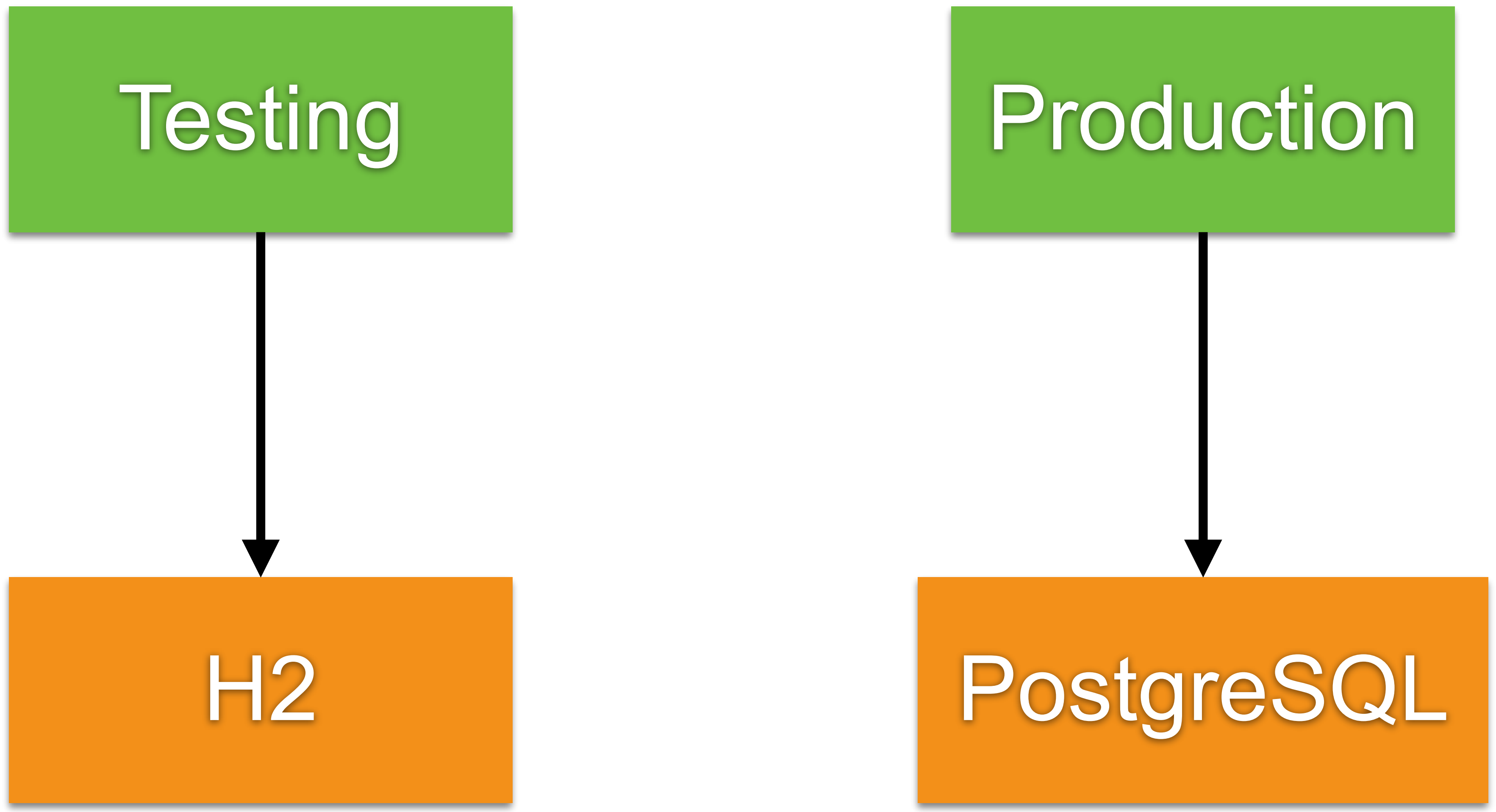
Working with database

Working with database





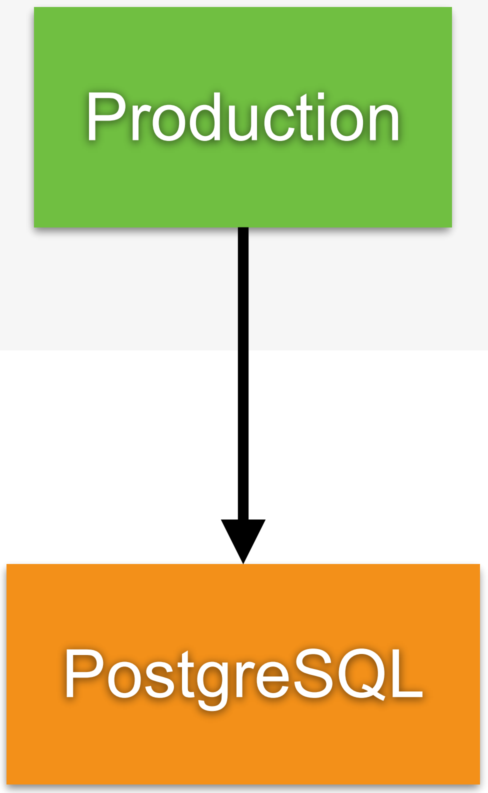
Working with database



Database Config for production

File src/main/resources/application.properties

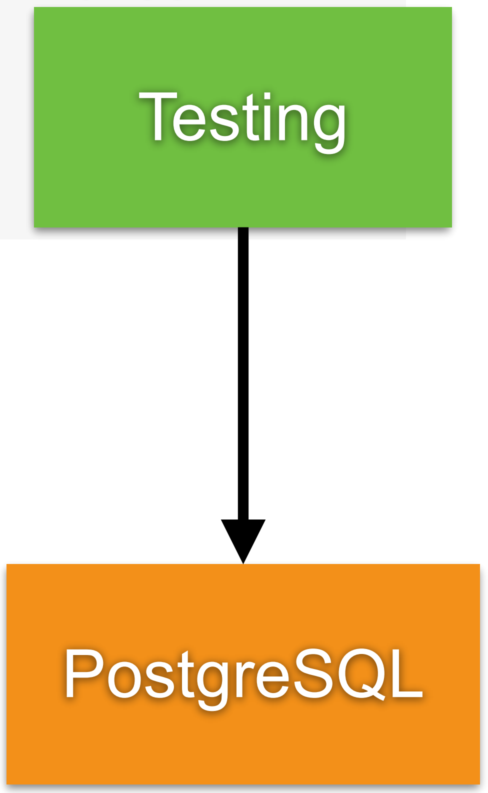
```
spring.datasource.url=${POSTGRES_URL:jdbc:postgresql://localhost:5432/demo}
spring.datasource.username=${POSTGRES_USER:postgres}
spring.datasource.password=${POSTGRES_PASS:password}

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.PostgreSQLDialect

# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto = update

spring.jpa.show-sql=true
```



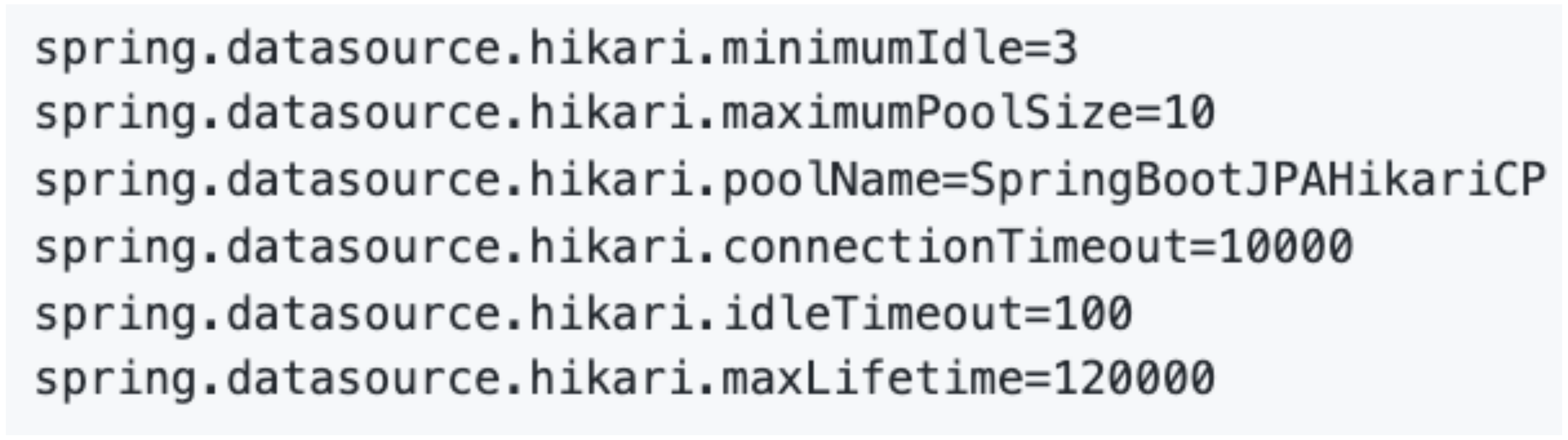
Database Config for test

File src/test/resources/application.properties

```
## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.username=sa
spring.datasource.password=

# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.H2Dialect

# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto=update
```



Tuning database configuration



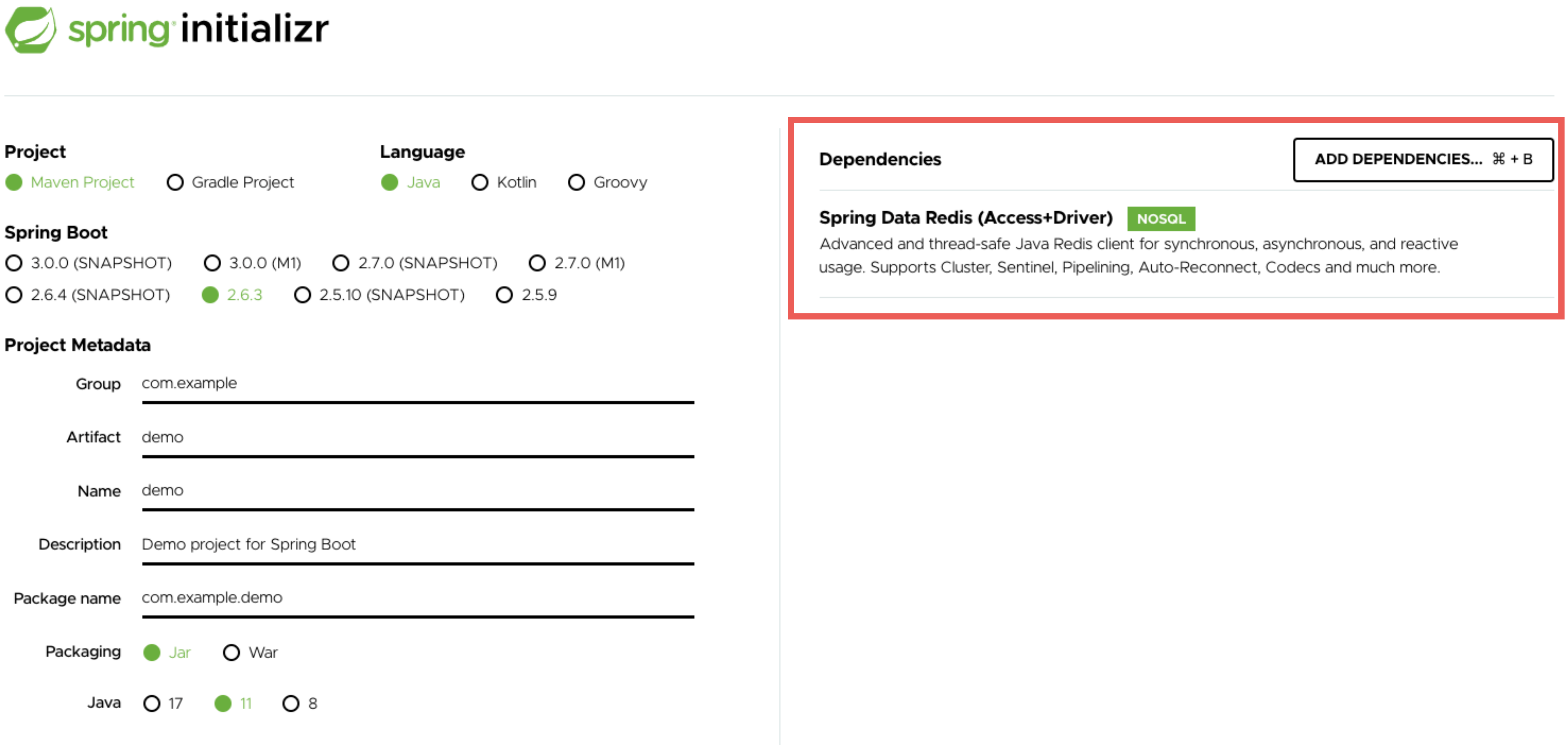
```
spring.datasource.hikari.minimumIdle=3
spring.datasource.hikari.maximumPoolSize=10
spring.datasource.hikari.poolName=SpringBootJPAHikariCP
spring.datasource.hikari.connectionTimeout=10000
spring.datasource.hikari.idleTimeout=100
spring.datasource.hikari.maxLifetime=120000
```

https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
#common-application-properties-data

Working with Redis

https://spring.io/projects/spring-data-redis

Add Spring Data Redis



spring initializr

Project
[✓ Maven Project] Gradle Project

Language
[✓ Java] Kotlin Groovy

Spring Boot
3.0.0 (SNAPSHOT) 3.0.0 (M1) 2.7.0 (SNAPSHOT) 2.7.0 (M1)
2.6.4 (SNAPSHOT) [✓ 2.6.3] 2.5.10 (SNAPSHOT) 2.5.9

Project Metadata

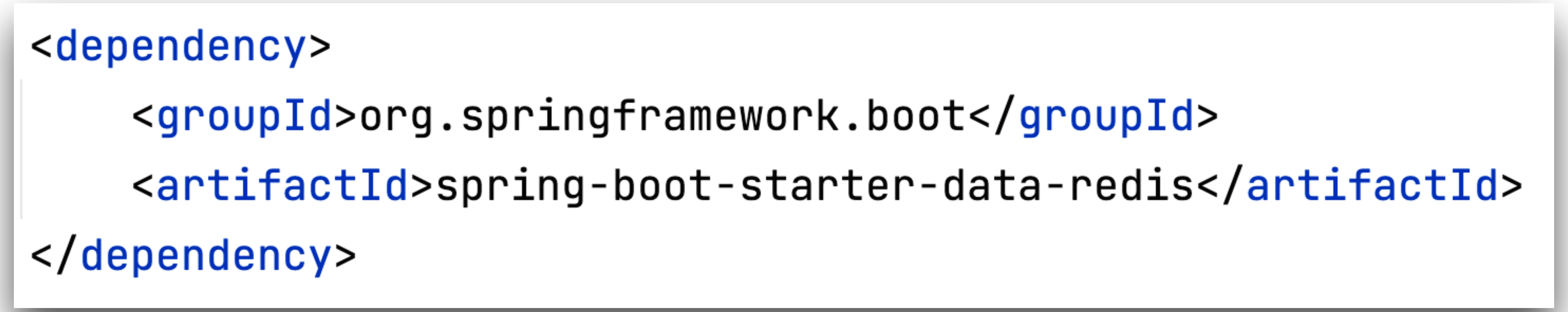
Group	com.example
Artifact	demo
Name	demo
Description	Demo project for Spring Boot
Package name	com.example.demo
Packaging	[✓ Jar] War
Java	17 [✓ 11] 8

Dependencies ADD DEPENDENCIES... ⌘ + B

Spring Data Redis (Access+Driver) NOSQL
Advanced and thread-safe Java Redis client for synchronous, asynchronous, and reactive usage. Supports Cluster, Sentinel, Pipelining, Auto-Reconnect, Codecs and much more.

https://start.spring.io/

Add dependency



```xml
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-redis</artifactId>
</dependency>
```

Redis connector

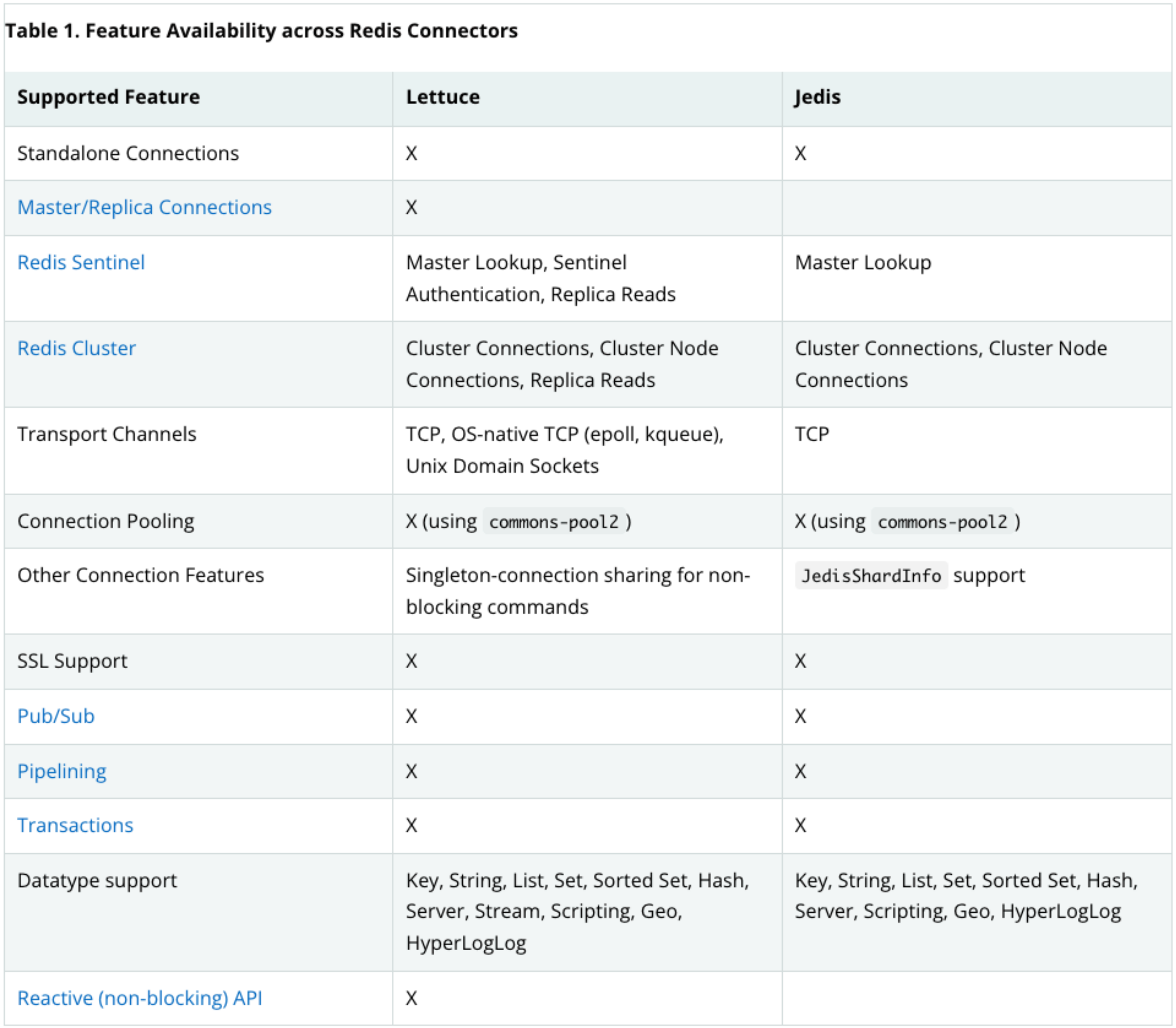


Table 1. Feature Availability across Redis Connectors

Supported Feature	Lettuce	Jedis
Standalone Connections	X	X
Master/Replica Connections	X	
Redis Sentinel	Master Lookup, Sentinel Authentication, Replica Reads	Master Lookup
Redis Cluster	Cluster Connections, Cluster Node Connections, Replica Reads	Cluster Connections, Cluster Node Connections
Transport Channels	TCP, OS-native TCP (epoll, kqueue), Unix Domain Sockets	TCP
Connection Pooling	X (using `commons-pool2`)	X (using `commons-pool2`)
Other Connection Features	Singleton-connection sharing for non-blocking commands	`JedisShardInfo` support
SSL Support	X	X
Pub/Sub	X	X
Pipelining	X	X
Transactions	X	X
Datatype support	Key, String, List, Set, Sorted Set, Hash, Server, Stream, Scripting, Geo, HyperLogLog	Key, String, List, Set, Sorted Set, Hash, Server, Scripting, Geo, HyperLogLog
Reactive (non-blocking) API	X	

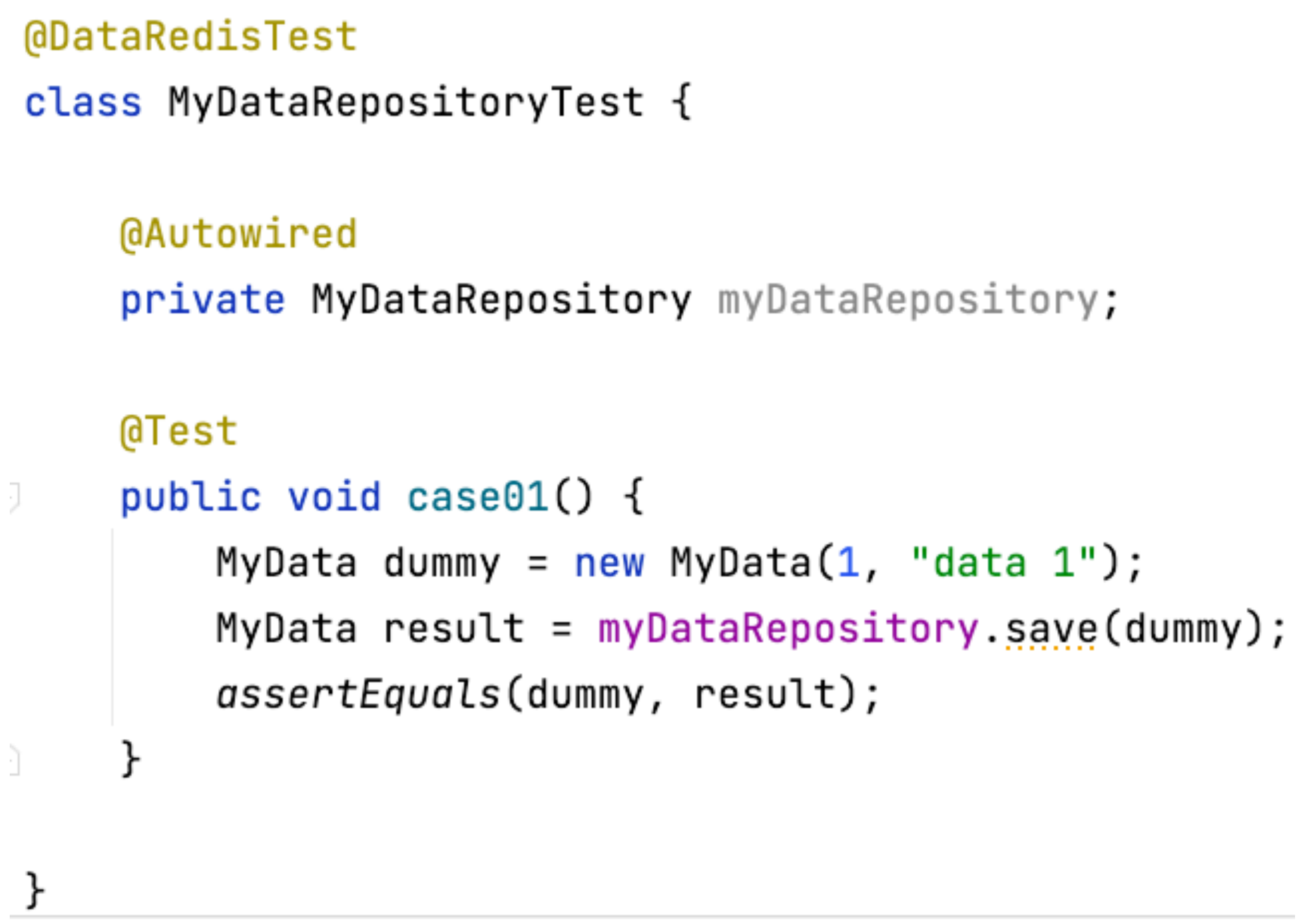
https://docs.spring.io/spring-data/data-redis/docs/current/reference/html/#redis:connectors:connection

Redis in Spring Boot

Configuration in Spring Boot

```java
@Configuration
public class RedisConfig {

    @Bean
    JedisConnectionFactory jedisConnectionFactory() {
        RedisStandaloneConfiguration configuration
                = new RedisStandaloneConfiguration("localhost", 6379);
        configuration.setPassword(RedisPassword.of("password"));
        return new JedisConnectionFactory(configuration);
    }

    @Bean
    public RedisTemplate<String, Object> redisTemplate() {
        RedisTemplate<String, Object> template = new RedisTemplate<>();
        template.setConnectionFactory(jedisConnectionFactory());
        return template;
    }
}
```

Spring Data Redis Test



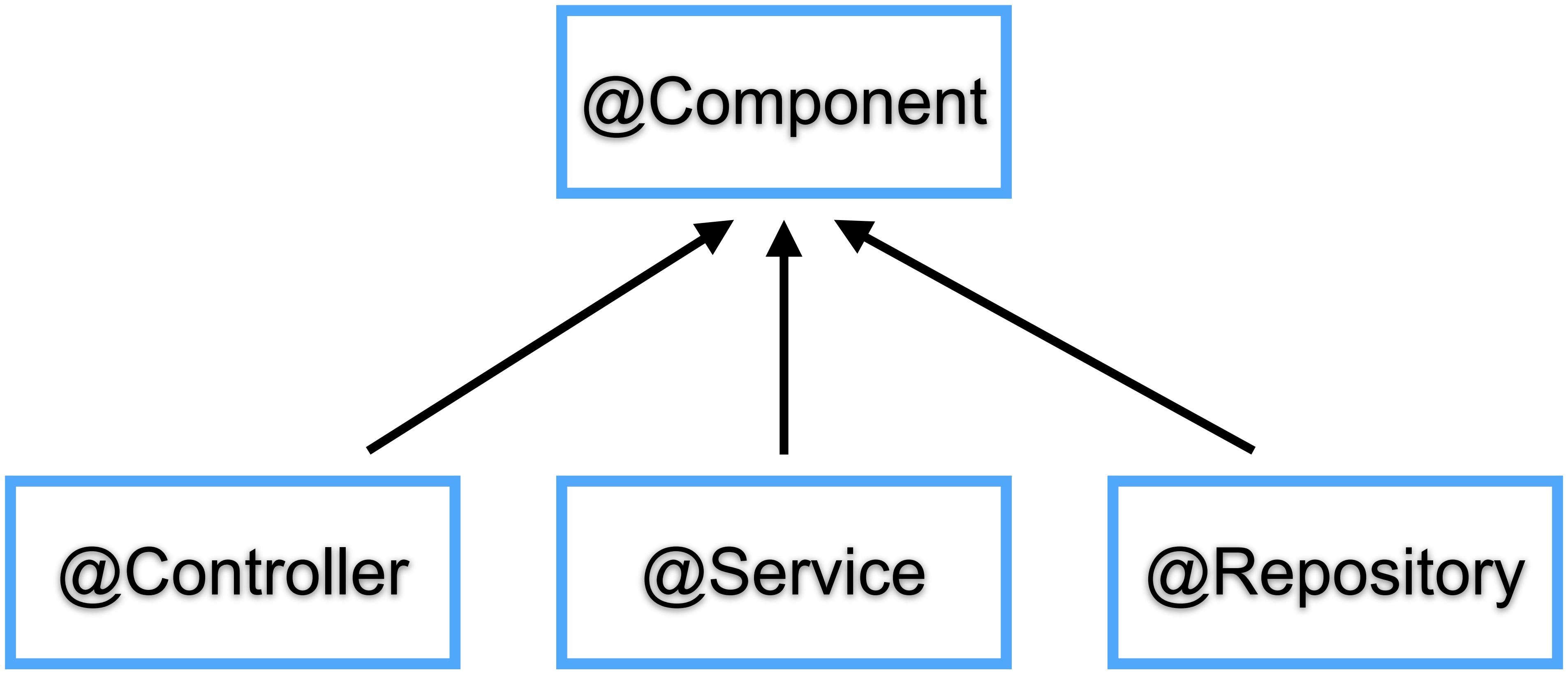
```java
@DataRedisTest
class MyDataRepositoryTest {

    @Autowired
    private MyDataRepository myDataRepository;

    @Test
    public void case01() {
        MyData dummy = new MyData(1, "data 1");
        MyData result = myDataRepository.save(dummy);
        assertEquals(dummy, result);
    }

}
```

Working with Testcontainers

https://www.testcontainers.org/

Scope of components in Spring

What are difference of …



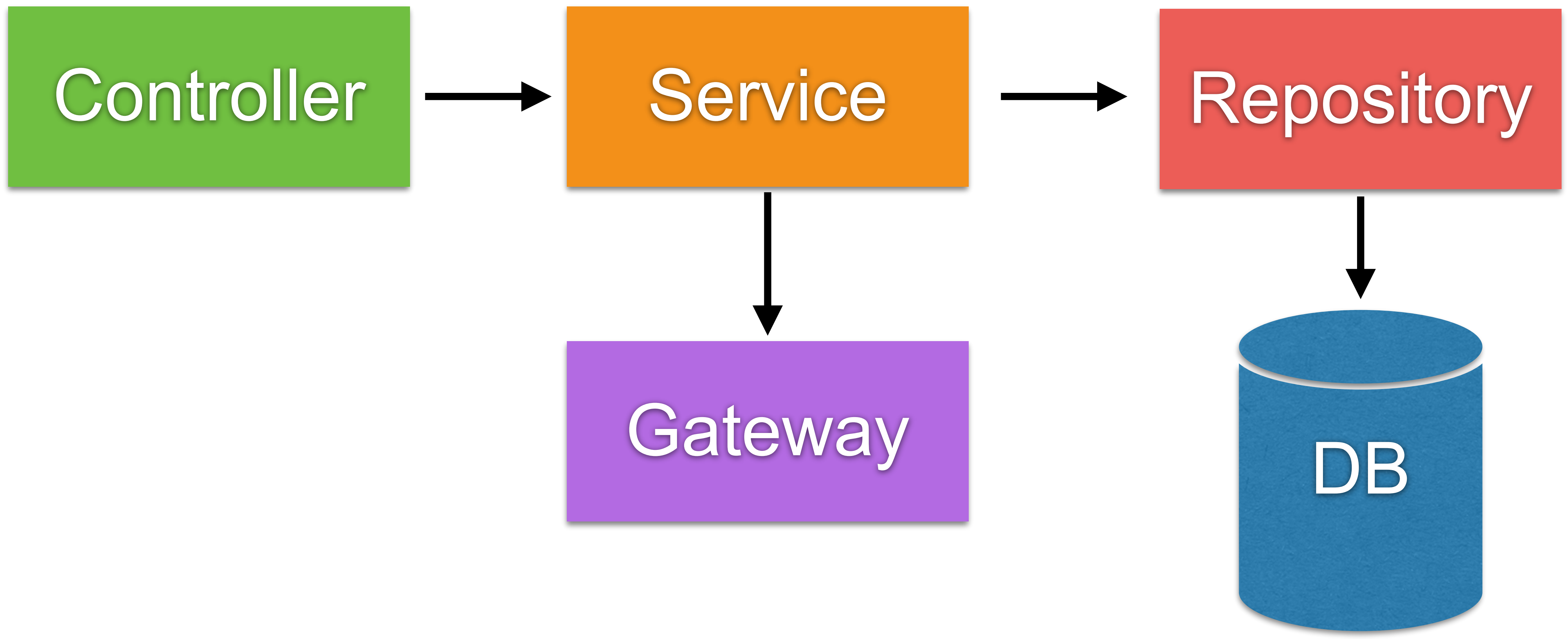
Bean Scopes

Scope	Description
singleton	Single instance for each container
prototype	Single bean definition to any number of object instances.
request	HTTP request
session	HTTP session
application	ServletContext

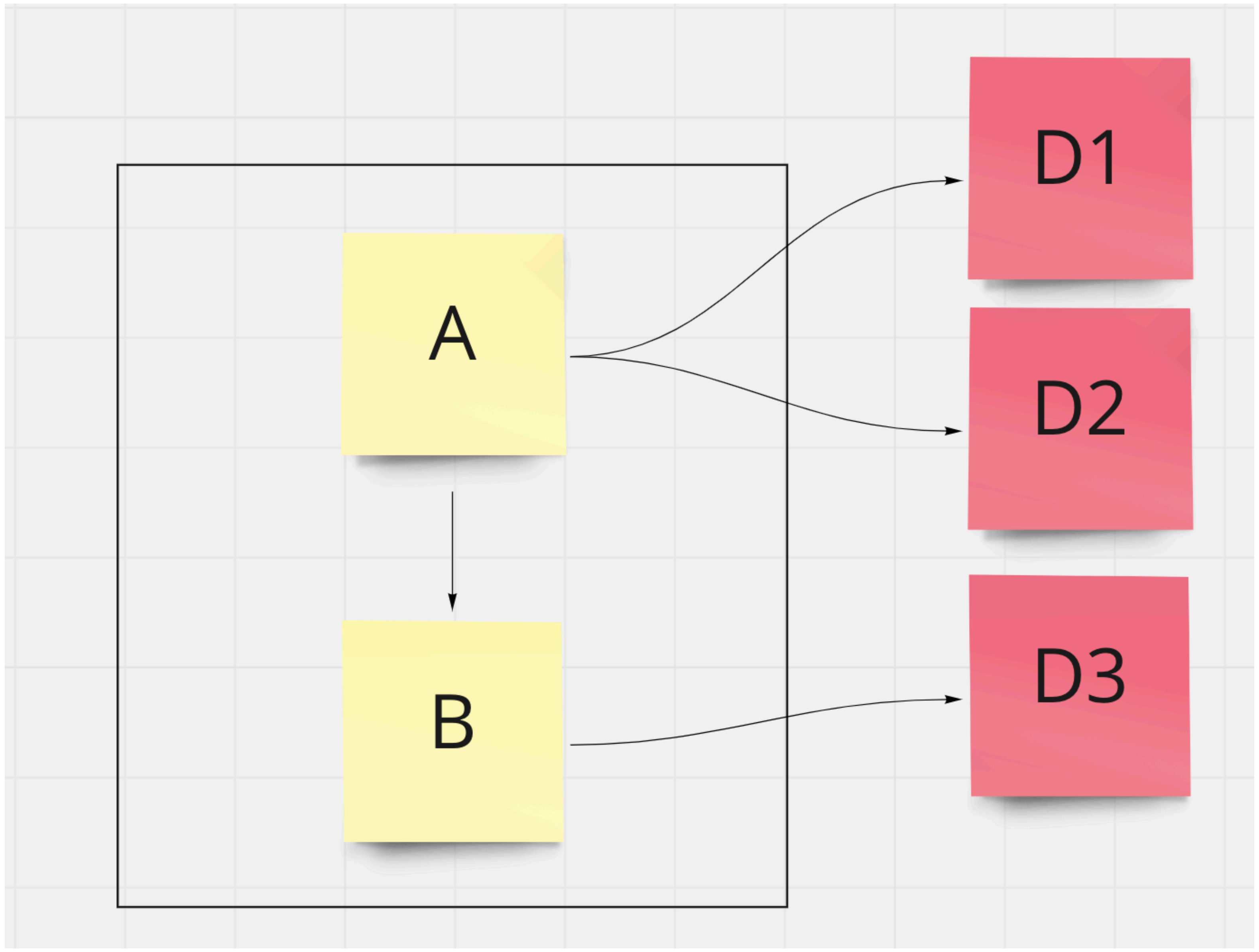
https://docs.spring.io/spring/docs/current/spring-framework-reference/core.html#beans-factory-scopes

Use cases of Mockito

How to test ?



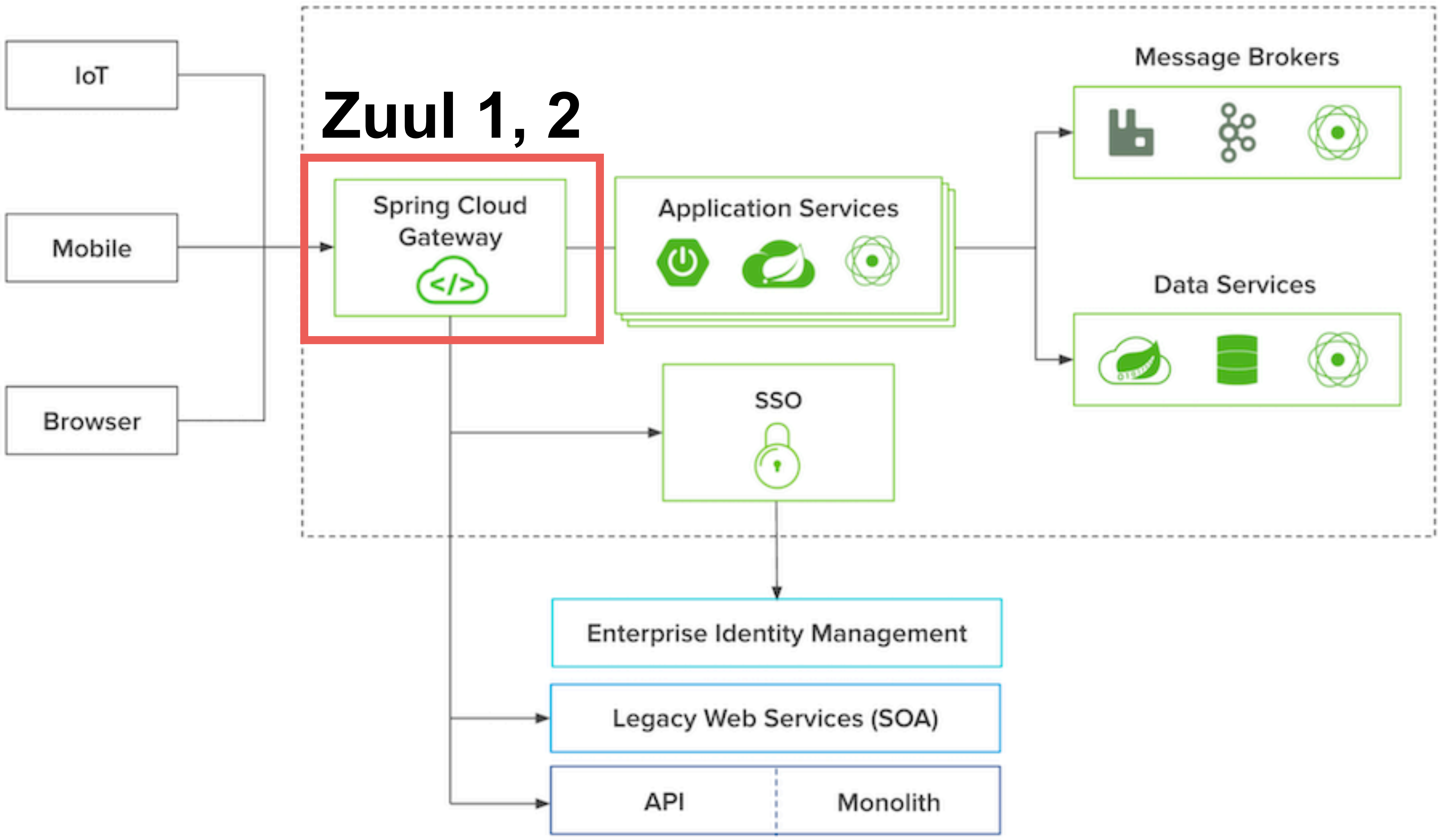
How to test ?



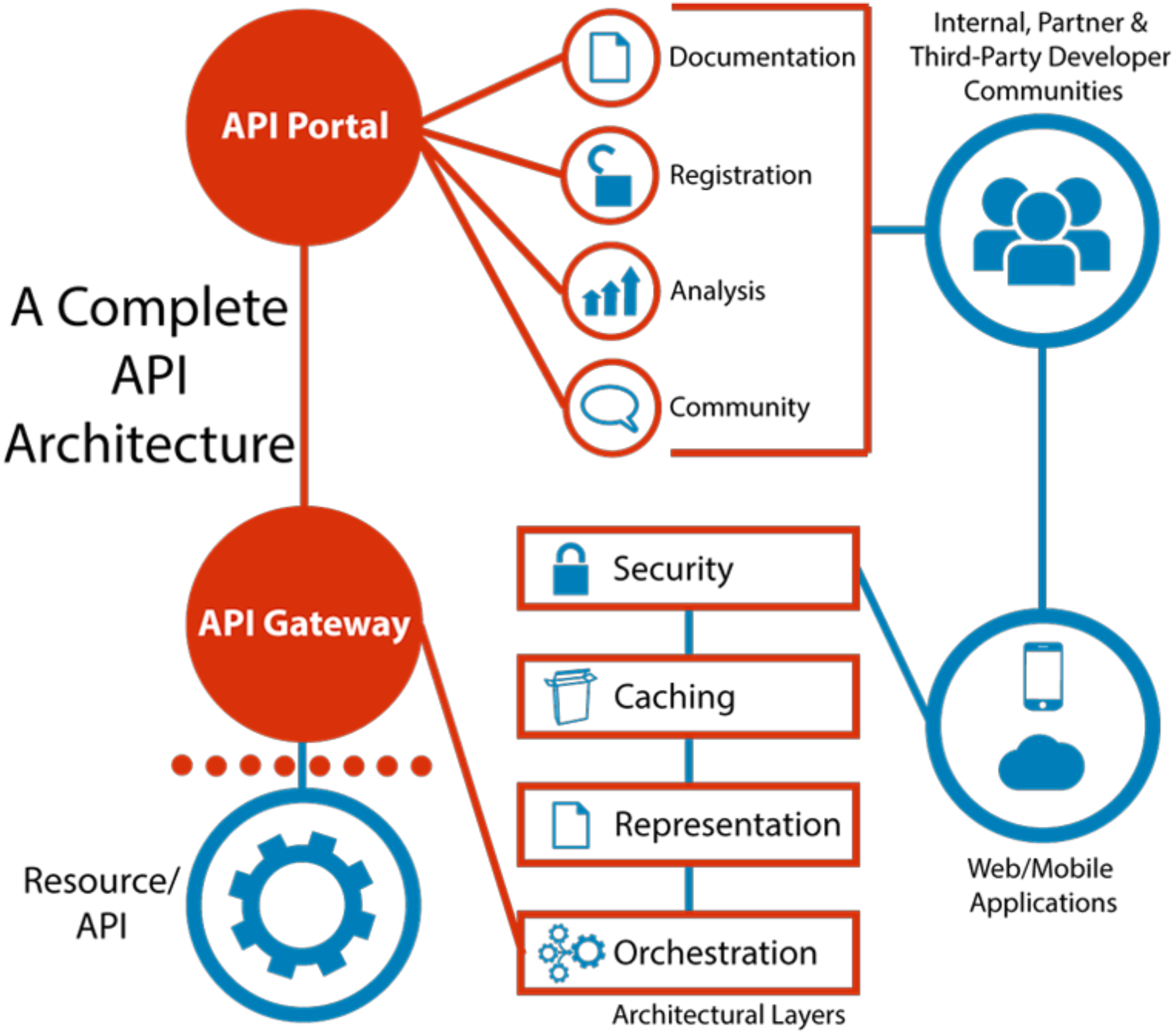
Response vs Entity object

Working with Docker

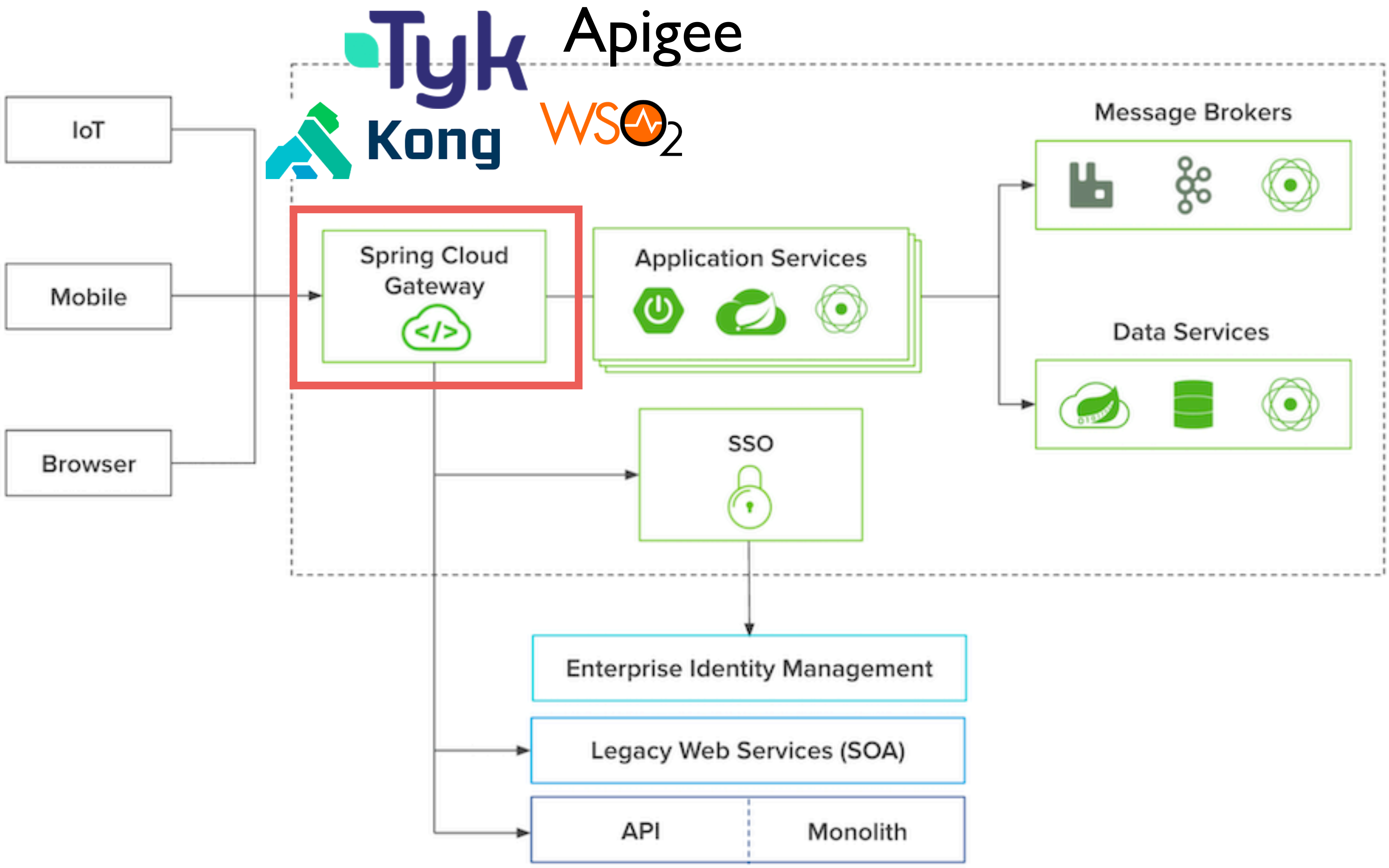
Spring Cloud Gateway



https://spring.io/projects/spring-cloud-gateway

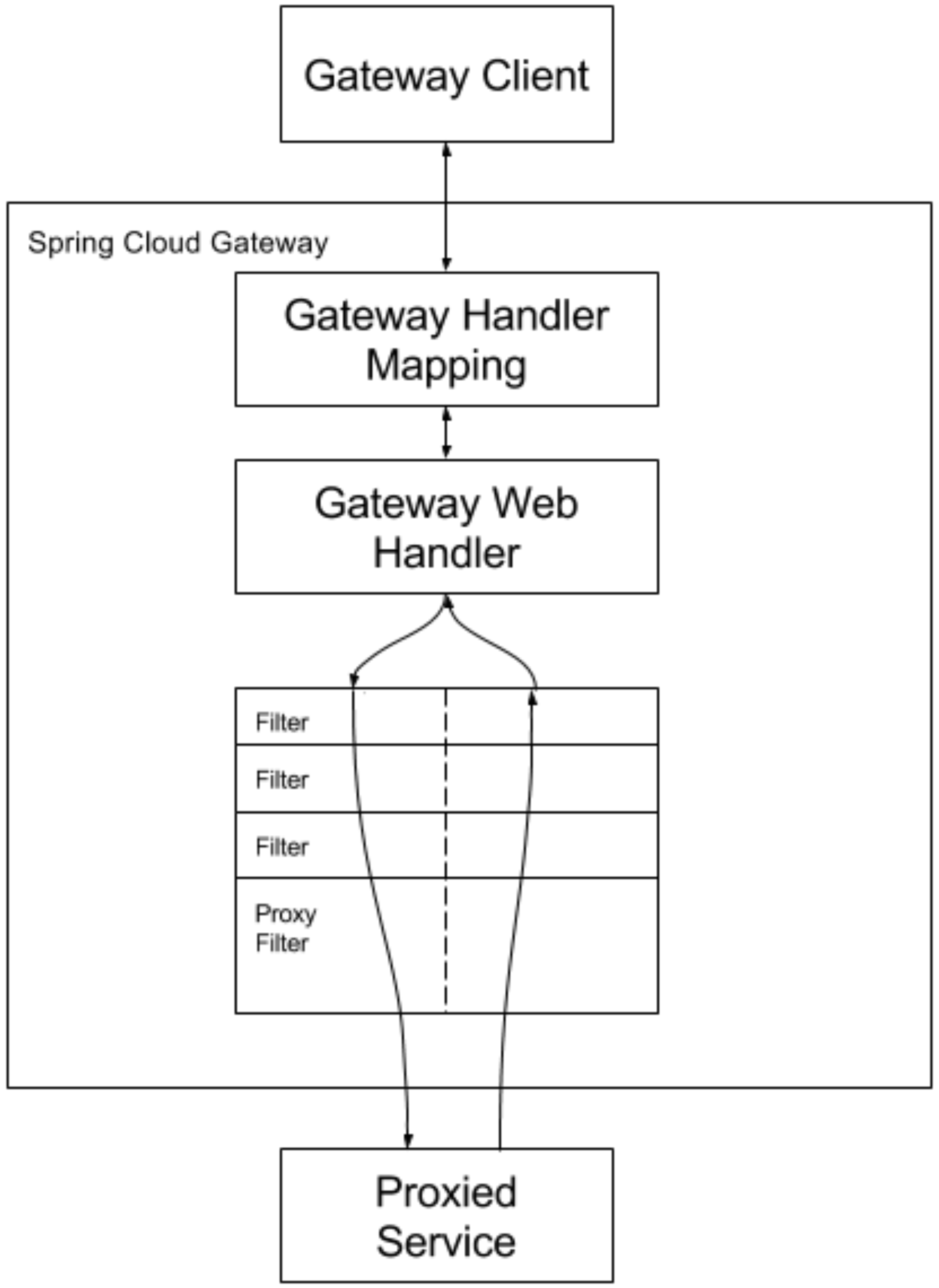


API Portal
Documentation
Registration
Analysis
Community
Internal, Partner & Third-Party Developer Communities
A Complete API Architecture
API Gateway
Security
Caching
Representation
Orchestration
Architectural Layers
Web/Mobile Applications
Resource/ API



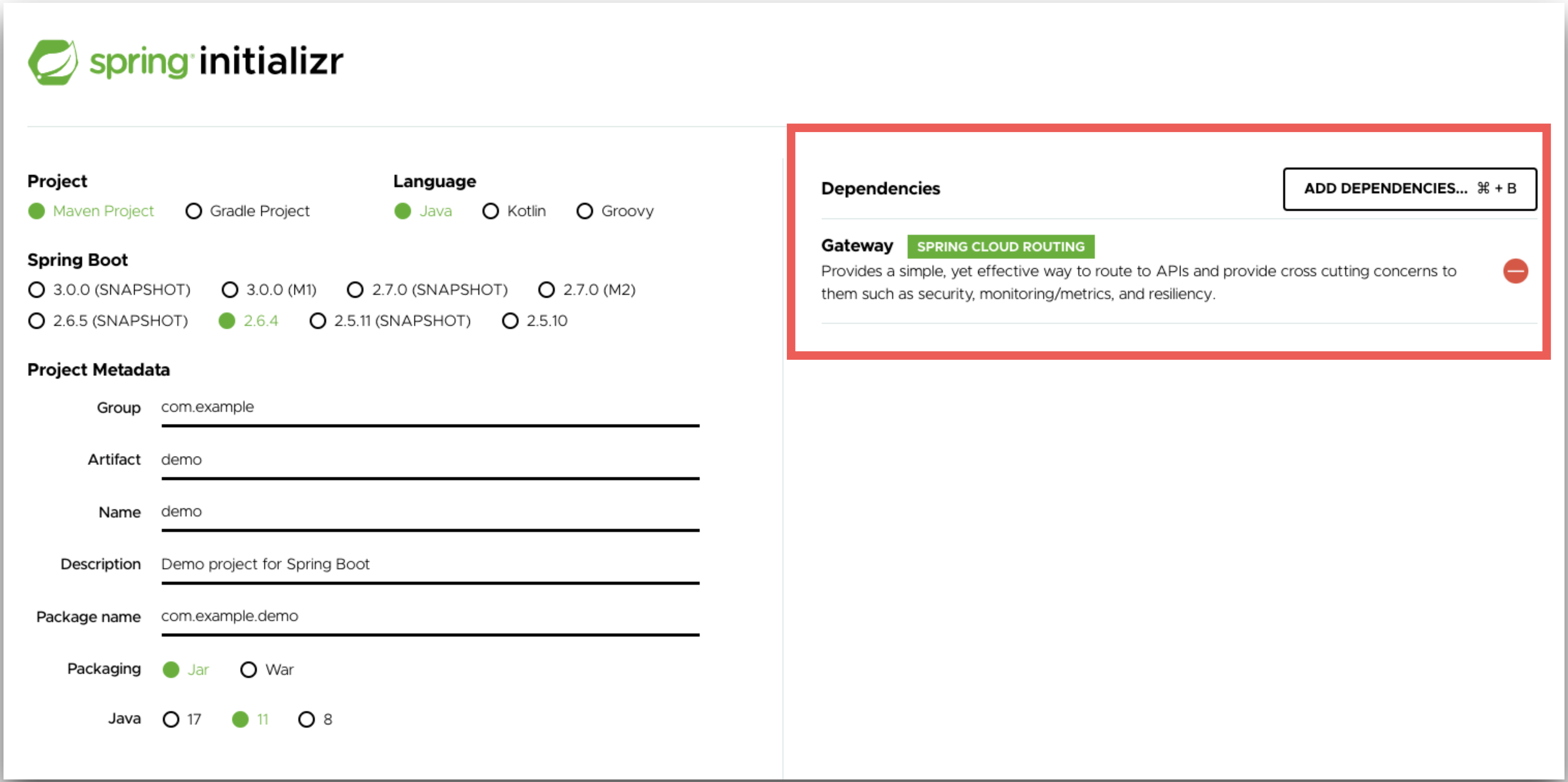
https://landscape.cncf.io/card-mode?category=api-gateway&grouping=category

Spring Cloud Gateway



https://docs.spring.io/spring-cloud-gateway/docs/current/reference/html/

Spring Cloud Gateway


spring initializr

Project
[✓ Maven Project] Gradle Project

Language
[✓ Java] Kotlin Groovy

Spring Boot
3.0.0 (SNAPSHOT) 3.0.0 (M1) 2.7.0 (SNAPSHOT) 2.7.0 (M2)
2.6.5 (SNAPSHOT) [✓ 2.6.4] 2.5.11 (SNAPSHOT) 2.5.10

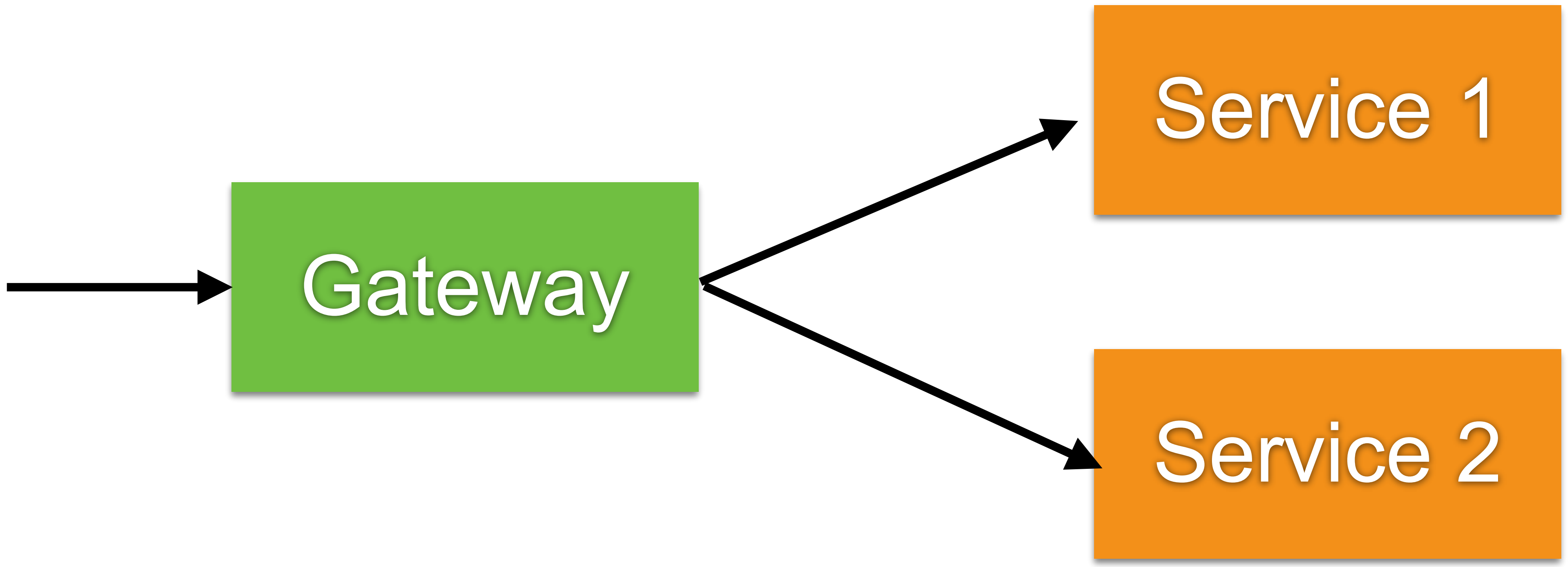
Project Metadata

Group	com.example
Artifact	demo
Name	demo
Description	Demo project for Spring Boot
Package name	com.example.demo
Packaging	[✓ Jar] War
Java	17 [✓ 11] 8

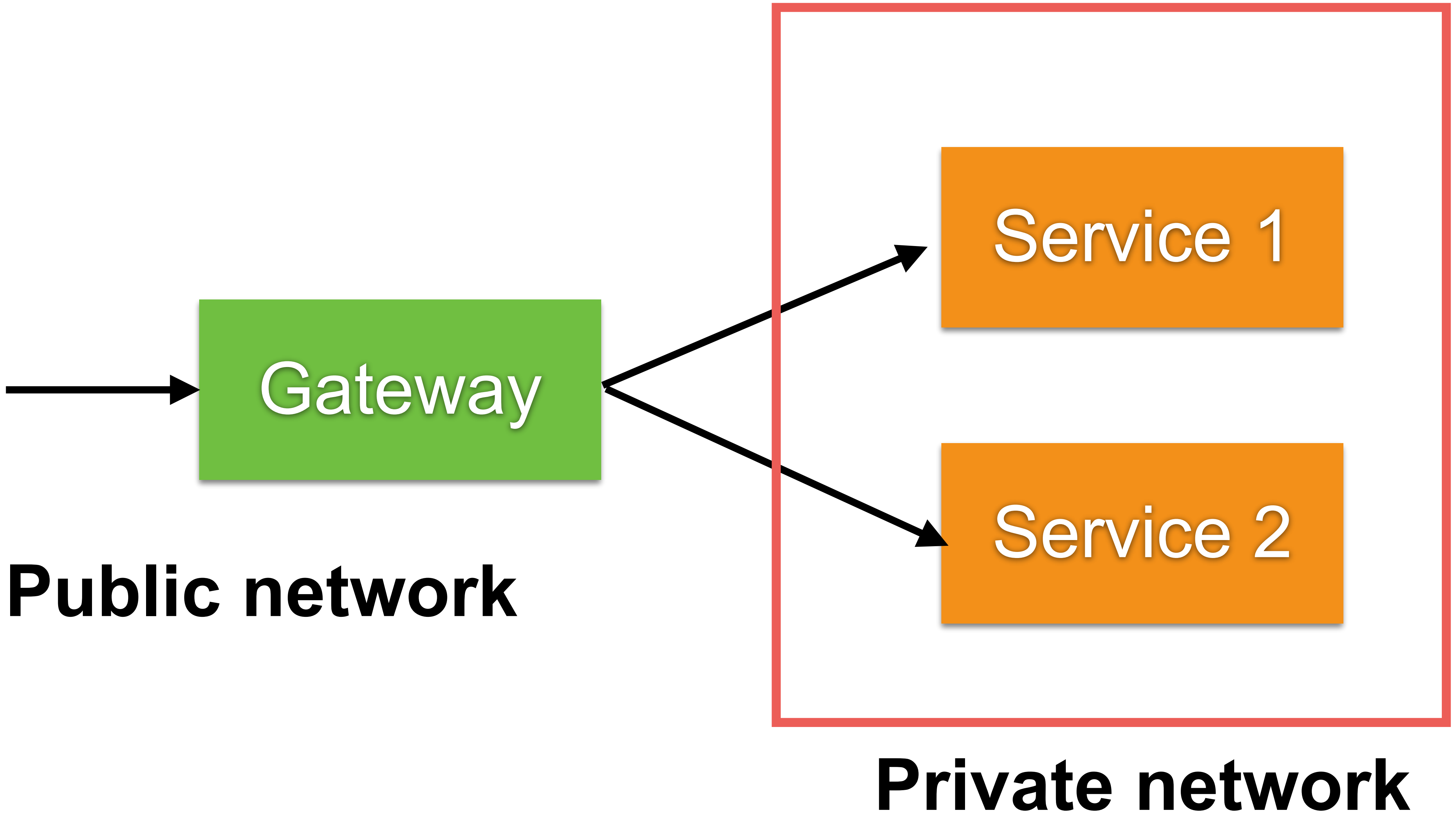
Dependencies

Gateway SPRING CLOUD ROUTING
Provides a simple, yet effective way to route to APIs and provide cross cutting concerns to them such as security, monitoring/metrics, and resiliency.

Spring Cloud Gateway



Spring Cloud Gateway



Q/A