Contents

[Java 1](#_Toc171365928)

[Spring Boot 1](#_Toc171365929)

[Rest Controller 1](#_Toc171365930)

[Bean 1](#_Toc171365931)

[Componen 1](#_Toc171365932)

[Service 2](#_Toc171365933)

[JPA 2](#_Toc171365934)

[Composite PK 2](#_Toc171365935)

[Pagination 2](#_Toc171365936)

[Others 2](#_Toc171365937)

[Mapper 2](#_Toc171365938)

[Date handling 2](#_Toc171365939)

[Testing 2](#_Toc171365940)

[Unit testing 2](#_Toc171365941)

[Controller 2](#_Toc171365942)

[Service 2](#_Toc171365943)

[JPA 2](#_Toc171365944)

[Entity manager 2](#_Toc171365945)

[Utility Class 2](#_Toc171365946)

[Integration test 2](#_Toc171365947)

[Wire Mock 2](#_Toc171365948)

# Java

## Stream Api

## Collection

# Spring Boot

## Rest Controller

## Bean

Component

## Service

## JPA

### Composite PK

### Pagination

## Batch Job

## Others

### Mapper

### Date handling

# Testing

## Unit testing

## Controller

@WebMVC @ContextConfiguration @MockBeans @MockMultipartFile

MainApplication(SpringBoot)

@ExtendsWith (SpringExtens.java)

## Service

@ExtendsWith (MokitoExtention.java)

@InjectMocks mainService

@Mocks /~~@MockBean~~ Other Service

MockedStatic,ReflectiontestUtil

Exception :arretThrows(Exceptionxyz.class()->{})

Envvaribale

1: Create application.yml in test and add env variable with dummy value

2: @TestPropertySource(properties = {"myproperty = foo"})

Or

static {

System.setProperty("myproperty", "foo");//direct in test method withour static

}

@Capture

ArgumentCapture<ABC.Class> classAbc;

File : Path fileclass;

@SnekyThrows

## JPA

## Entity manager

## Utility Class

@SpringBootTest

## Integration test

## Wire Mock

# Kafka

# NOS SQL DB

# Dev Ops

# Architecture

# Micro services

## 12 factors App