WELCOME BACK
DAY 4

<Cre8tive Devs. Software/>

Spring/Spring Boot Crash Course

For TD Bank

MEET YOUR CRASH COURSE TEAM

TANGY F.
CEO

HAL M.
DEVELOPER

WILLIAM D.
DEVELOPER

AGENDA

LESSON 4

1. Recap Lesson of 3:
   >>Rapid review of lesson 3

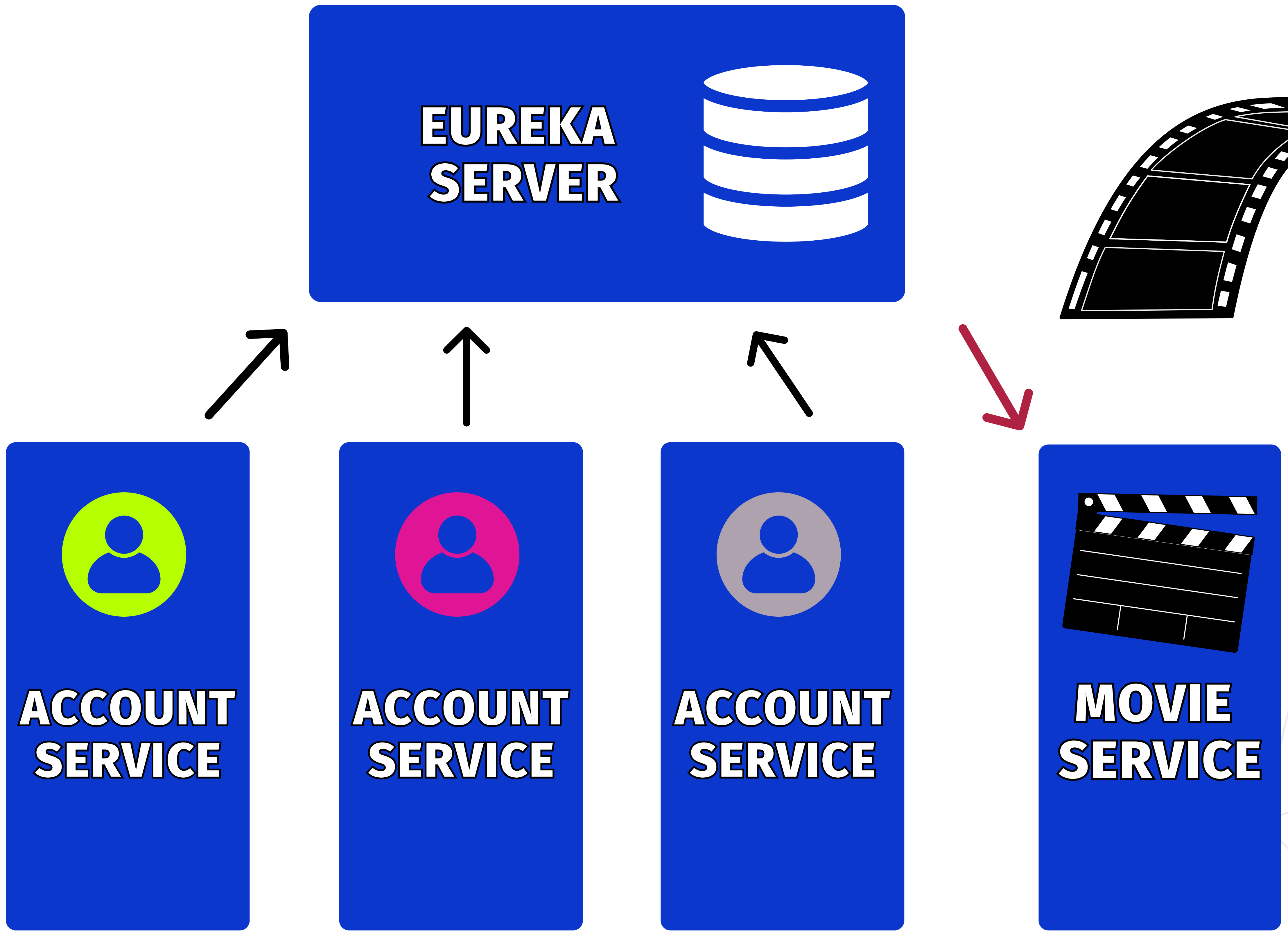
2. Current Lesson- Microservices (Movie Service)

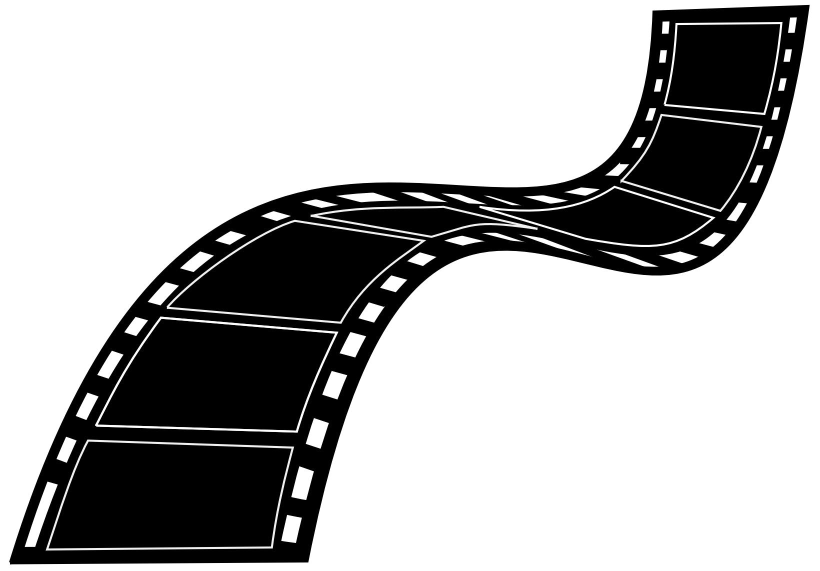
3. Q.A

<Cre8tive Devs. Software/>

WE ARE BUILDING

BITE SIZE
MOVIE REVIEW APP





<Cre8tive Devs. Software/>

WELCOME TO LESSON 4

Rapid Review of LESSON 1-3

Rapid Review Trivia

1. Name three constraints available to use with Jakarta validation.

2. Which method from PasswordEncoder do we use to encrypt a string using our specified encryption method?

<Cre8tive Devs. Software/>
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

Rapid Review of LESSON 1-3

WELCOME TO LESSON 4

Rapid Review Trivia

Productivity Principles:

1. **What is separation of concern in software development?**

2. **What does guardrails mean in software development?**

<Cre8tive Software/> Devs.
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

WELCOME TO LESSON 4

Rapid Review of

LESSON 3

Rapid Review

Spring Security I

<Cre8tive Devs. Software/>
MOBILE APPS•WEBSITES•AR•VIDEO GAMES•SECURITY

LESSON 4

Microservices

LESSON 4

MICROSERVICES WITH MOVIE SERVICE

1 What is a Microservice?

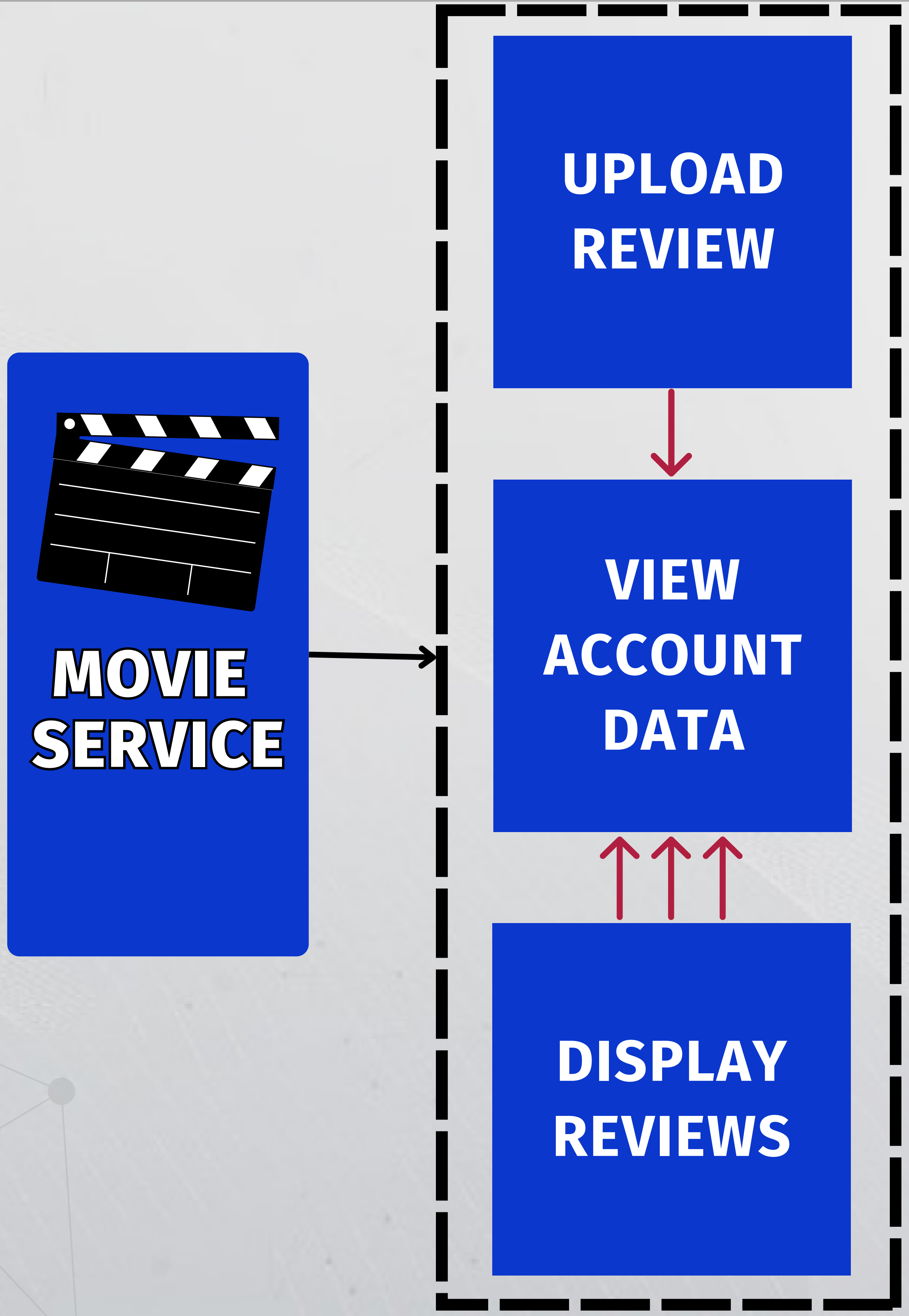
2 Service implementation in Movie Service

3 12 Factors and Design Patterns used

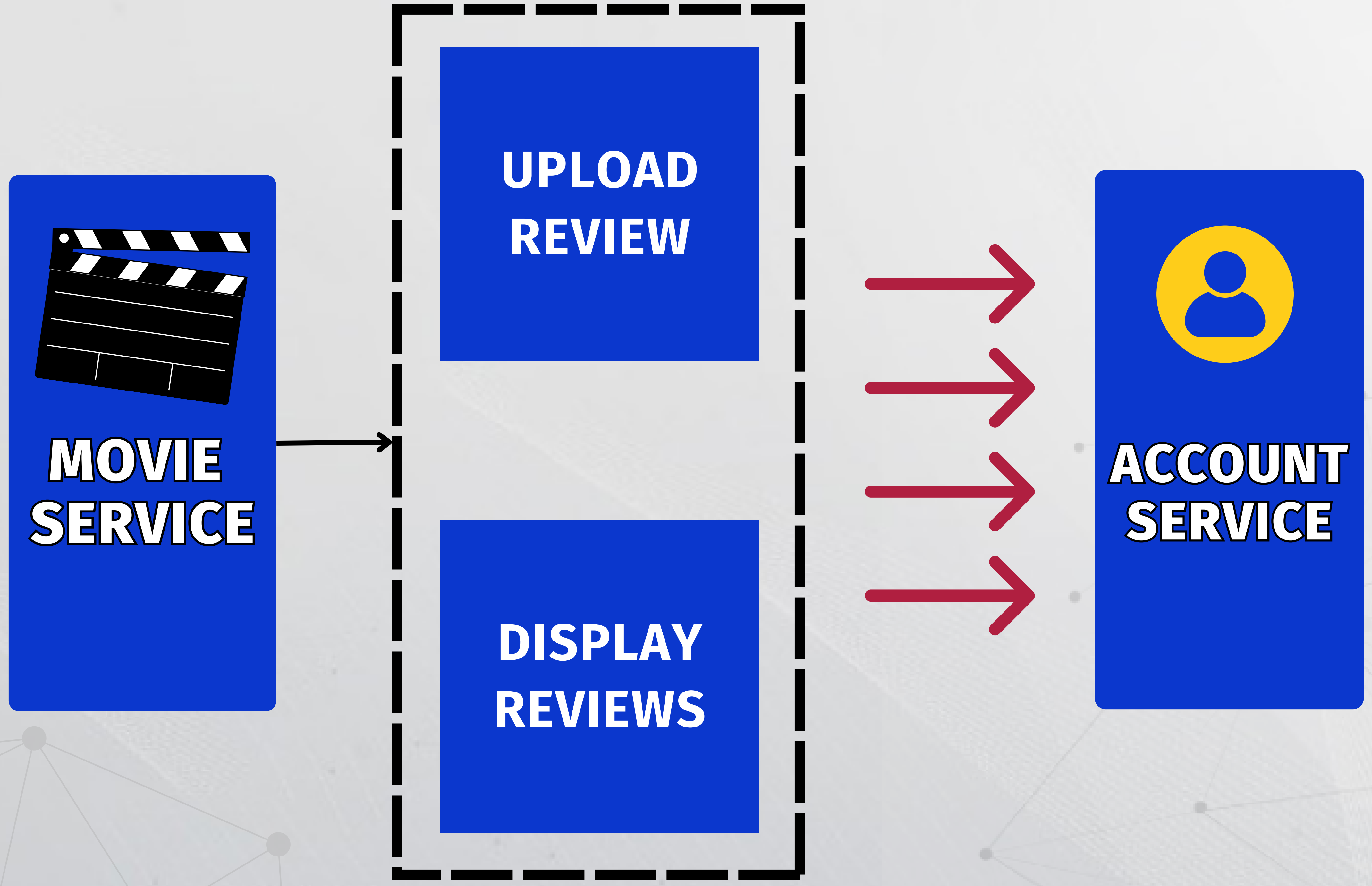
4 Advantages of Microservices

<Cre8tive Software/>

WHY USE MICROSERVICES?



WHY USE MICROSERVICES?



Microservices

Use case

Implementation

Services in Movie Service application

Project model

REST API's implementation

12 Factor App

Requirements

Pros & Cons of Microservices

Advantages

Disadvantages

DESIGN PATTERNS USED

HOW WERE THEY USED?

- Singleton Pattern
- Private Class Data Pattern
- Chain of Responsibility Pattern

12-FACTOR USED

Factor 2: Dependencies

Watch for how we declare Gradle dependencies, and how we don't require Gradle dependencies to be installed system-wide.

Factor 7: Port Binding

Watch for how we use HTTP ports when building & running our gradle application.

Factor 3: Config

Watch for where we store our config information, and consider how we could share this code to a repository without exposing sensitive information.

Factor 4: Backing Services

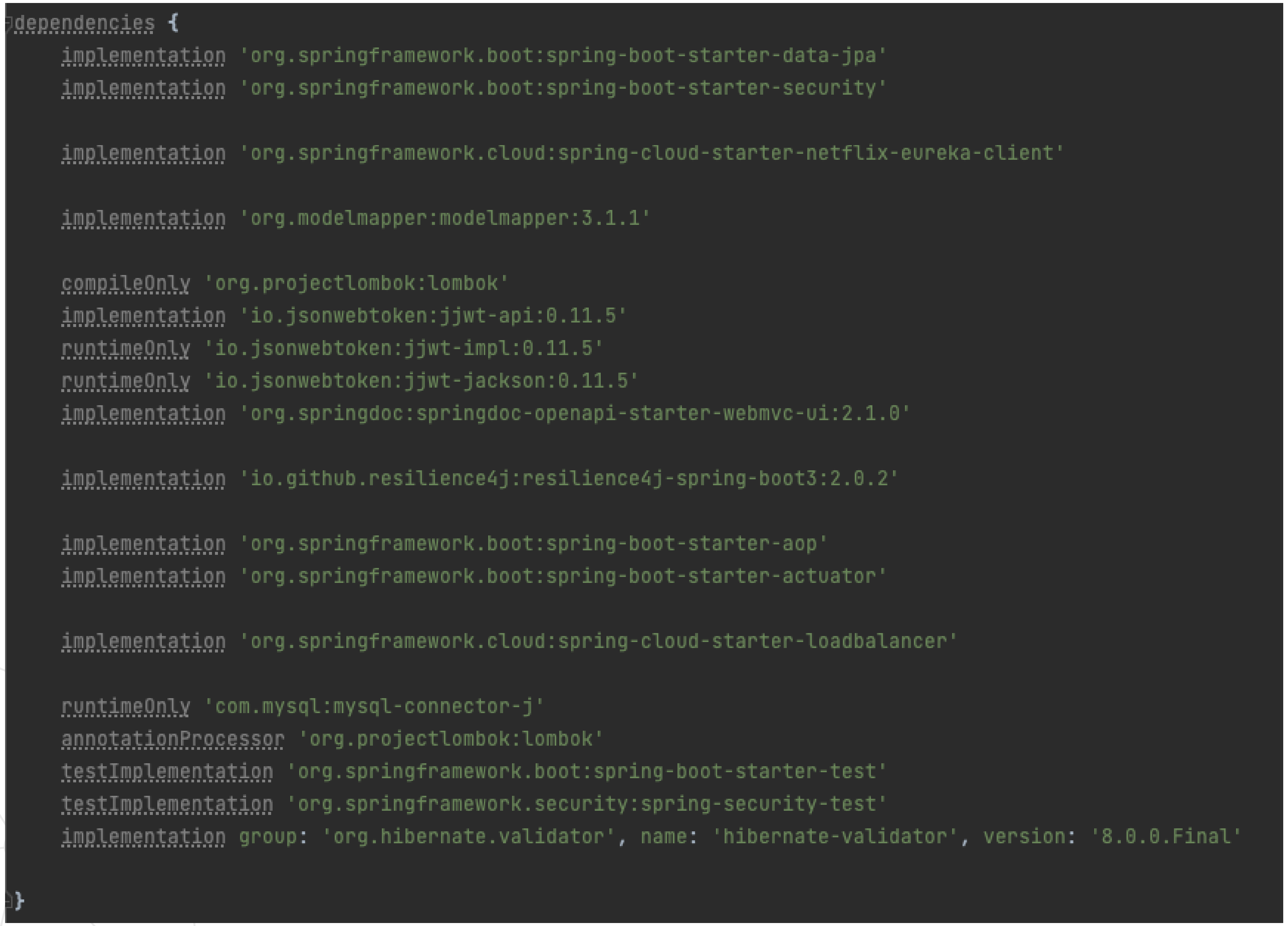
Take note of all the locations that we reference MySQL in the application config.

Factor 6: Processes

Take note of how the Movie and Review entities are stored at the end of each function's processing.

CODING EXERCISE #1

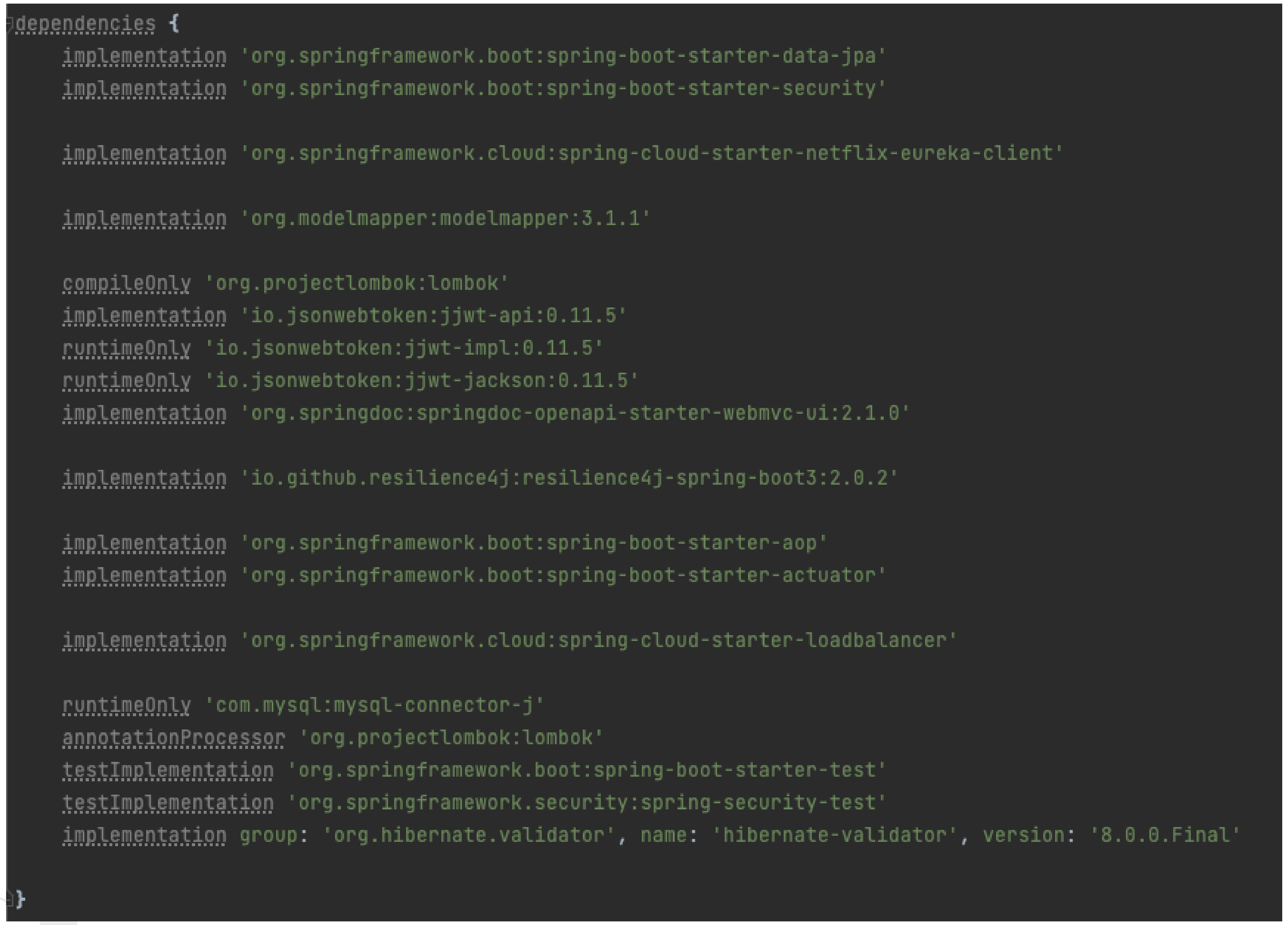
The below code is from our build.gradle file, and lists our dependencies used. Write code that will add a dependency for spring-boot-starter-web, from the org.springframework.boot group.



```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-security'

    implementation 'org.springframework.cloud:spring-cloud-starter-netflix-eureka-client'

    implementation 'org.modelmapper:modelmapper:3.1.1'

    compileOnly 'org.projectlombok:lombok'
    implementation 'io.jsonwebtoken:jjwt-api:0.11.5'
    runtimeOnly 'io.jsonwebtoken:jjwt-impl:0.11.5'
    runtimeOnly 'io.jsonwebtoken:jjwt-jackson:0.11.5'
    implementation 'org.springdoc:springdoc-openapi-starter-webmvc-ui:2.1.0'

    implementation 'io.github.resilience4j:resilience4j-spring-boot3:2.0.2'

    implementation 'org.springframework.boot:spring-boot-starter-aop'
    implementation 'org.springframework.boot:spring-boot-starter-actuator'

    implementation 'org.springframework.cloud:spring-cloud-starter-loadbalancer'

    runtimeOnly 'com.mysql:mysql-connector-j'
    annotationProcessor 'org.projectlombok:lombok'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    testImplementation 'org.springframework.security:spring-security-test'
    implementation group: 'org.hibernate.validator', name: 'hibernate-validator', version: '8.0.0.Final'

}
```

CODING EXERCISE #1

The below code is from our build.gradle file, and lists our dependencies used. Write code that will add a dependency for spring-boot-starter-web, from the org.springframework.boot group.

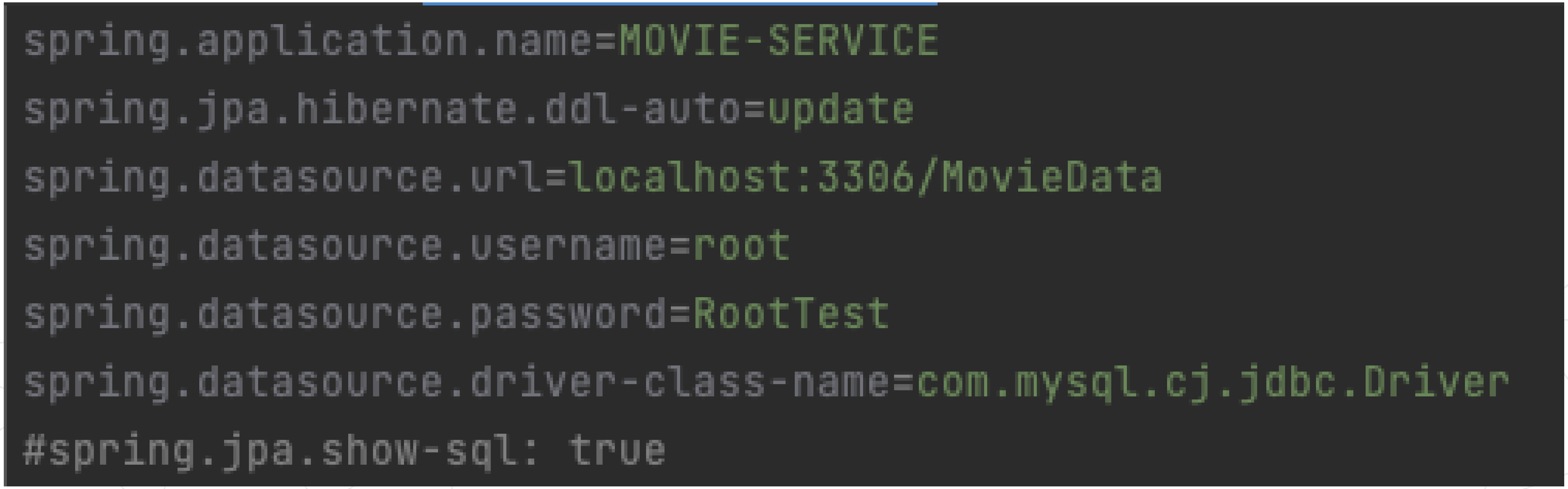


```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-security'

    implementation 'org.springframework.cloud:spring-cloud-starter-netflix-eureka-client'

    implementation 'org.modelmapper:modelmapper:3.1.1'

    compileOnly 'org.projectlombok:lombok'
    implementation 'io.jsonwebtoken:jjwt-api:0.11.5'
    runtimeOnly 'io.jsonwebtoken:jjwt-impl:0.11.5'
    runtimeOnly 'io.jsonwebtoken:jjwt-jackson:0.11.5'
    implementation 'org.springdoc:springdoc-openapi-starter-webmvc-ui:2.1.0'

    implementation 'io.github.resilience4j:resilience4j-spring-boot3:2.0.2'

    implementation 'org.springframework.boot:spring-boot-starter-aop'
    implementation 'org.springframework.boot:spring-boot-starter-actuator'

    implementation 'org.springframework.cloud:spring-cloud-starter-loadbalancer'

    runtimeOnly 'com.mysql:mysql-connector-j'
    annotationProcessor 'org.projectlombok:lombok'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    testImplementation 'org.springframework.security:spring-security-test'
    implementation group: 'org.hibernate.validator', name: 'hibernate-validator', version: '8.0.0.Final'

}
```

Answer: implementation 'org.springframework.boot:spring-boot-starter-web'

CODING EXERCISE #2

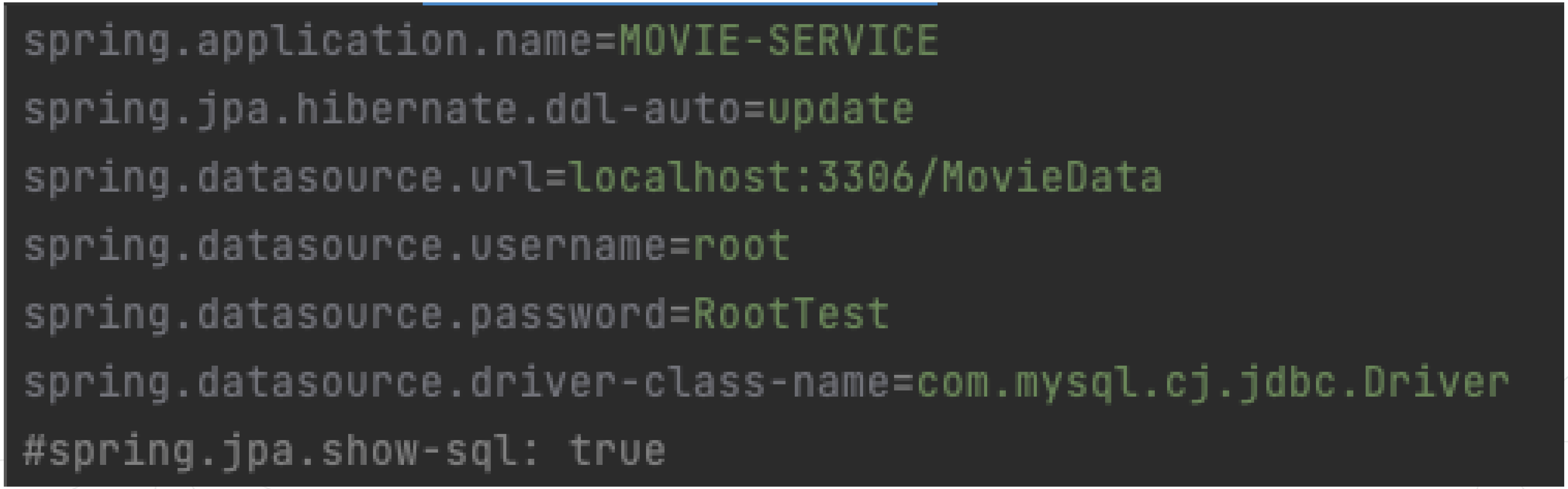
Below is a snippet from our application.properties file. We want to connect our service to a MySQL database called MovieData, that is present at localhost:3306, with a username and password of root and RootTest respectively. Which part of this code is incorrect?



```
spring.application.name=MOVIE-SERVICE
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=localhost:3306/MovieData
spring.datasource.username=root
spring.datasource.password=RootTest
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
#spring.jpa.show-sql: true
```

CODING EXERCISE #2

Below is a snippet from our application.properties file. We want to connect our service to a MySQL database called MovieData, that is present at localhost:3306, with a username and password of root and RootTest respectively. Which part of this code is incorrect?



```
spring.application.name=MOVIE-SERVICE
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=localhost:3306/MovieData
spring.datasource.username=root
spring.datasource.password=RootTest
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
#spring.jpa.show-sql: true
```

Datasource URL does not use jdbc properly. The correct line is the following.

spring.datasource.url=jdbc:mysql://localhost:3306/MovieData

THANK YOU

<Cre8tive Devs. Software/>

Crash Course

We will see you Tomorrow