

**2023-2024 学年 第 2 学期**

**框架技术开发 实验报告**

**专 业： 软件工程**

**班 级： 5班**

**学 号：2021105220250**

**姓 名： 李艺杰**

**指导教师： 丁钰**

**地 点： 11-206**

**日 期： 2024.4.29**

|  |
| --- |
| **一、实验题目**  实验八Service层实现类文件的建立 |
| **二、实验目的**  1、掌握Service层实现类文件的建立方法。  2、学习接口的实现类的建立方法。  3、理解Annotation注解配置在实现类中的应用。  4、掌握Service实现类与Controller层控制器的关系。 |
| **三、实验内容**  1、创建Service实现类：为实验七中定义的每个Service接口创建对应的实现类。  2、实现接口方法：确保实现类中的方法与接口中定义的方法完全一致。  3、注解配置：在实现类中应用适当的注解，如@Service，以标识实现类为Spring容器管理的Service组件。  4、异常处理：在实现类的方法声明中包含throws Exception;，以处理可能发生的异常。  5、与Controller层的交互：理解Service实现类如何被Controller层调用。 |
| 1. **实验步骤**   1、创建Service实现类的包：  在项目中创建一个新的包，命名为service.impl。  2、为每个Service接口创建实现类：  在service.impl包中，为UserService、MovieService、ReviewService等接口创建对应的实现类，如UserServiceImpl、MovieServiceImpl、ReviewServiceImpl。  3、实现接口方法：  对于UserServiceImpl，实现UserService接口中定义的所有方法，如registerUser、loginUser、listUsersByPage等。  确保实现类中的方法签名与接口中的定义完全一致。  4、应用注解：  在每个Service实现类的上方添加@Service注解，以便Spring容器能够识别并管理这些组件。  5、编写业务逻辑：  在Service实现类的每个方法中，编写调用DAO层接口的业务逻辑代码。  例如，在UserServiceImpl的registerUser方法中，调用UserMapper的insertUser方法。  6、异常处理：  在每个可能抛出异常的方法声明末尾添加throws Exception;。  7、测试Service实现类：  编写单元测试或集成测试，验证Service实现类的业务逻辑是否正确。  确保Service实现类能够与DAO层正确交互。  8、与Controller层集成：  在Controller层中，通过Spring的依赖注入机制注入Service实现类。  例如，在UserController中注入UserService，并在相应的处理方法中调用Service层提供的方法。  9、配置Spring容器：  确保Spring配置文件中包含了对Service实现类的扫描，使用@ComponentScan注解指定包路径。  10、运行和测试整个应用：  运行Spring应用，并使用浏览器或Postman等工具测试Controller层的接口。  验证Service层的业务逻辑是否按预期工作。 |
| 1. **遇到的问题及处理记录**   1、问题：Service实现类未被Spring容器管理。  处理：确保Service实现类上使用了@Service注解，并且Spring配置文件中包含了@ComponentScan注解指向了实现类的包路径。  2、问题：Service实现类与接口方法不匹配。  处理：检查Service实现类中的方法是否与接口中定义的方法完全一致，包括方法名、参数列表和返回类型。确保遵循接口的契约。  3，问题：异常未被捕获或未正确处理。  处理：在Service实现类的方法中声明throws Exception;，并在方法内部适当捕获和处理异常，或者在Controller层中统一处理异常。 |
| **六、实验结果及分析**  实验结果：  1、Service实现类成功创建：  所有实验七中定义的Service接口都有了对应的实现类，例如UserServiceImpl、MovieServiceImpl等。  2、接口方法正确实现：  实现类中的方法与接口中定义的方法名、参数、格式完全一致，符合接口契约。  3、注解配置正确应用：  @Service注解已正确应用于所有Service实现类，使得Spring容器能够识别并管理这些Bean。  4、Service与Controller层交互顺畅：  Controller层能够通过依赖注入成功调用Service实现类中的方法，业务逻辑流转顺畅。  分析：  实现类与接口的一致性保证了代码的可维护性和可测试性，也使得团队协作更加顺畅。  异常的正确处理保证了应用的稳定性和健壮性，避免了因未捕获异常导致的系统崩溃。  Service层与Controller层的顺畅交互是实现业务逻辑的关键，这表明了良好的软件架构设计。  测试覆盖的全面性保证了Service层的实现质量，减少了生产环境中的潜在错误。 |
| **评语：**  **分数： 教师签字： 日期： 年 月 日** |

备注：学生用黑色签字笔填写，于实验项目结束后按时完成。