

代码说明

根据分层解耦,将该功能分为三大类: Controller Service Mapper; 还有pojo包,定义User类和Result类; utils工具类, 定义生成JWT令牌

附:1. Result类让登录和注册有统一响应结果

2. 在数据库中建立表格,并在application.properties中添加数据库的坐标及密码等信息
3. 引入Mybatis,lombok,springframework等依赖时千万要注意是否存在**版本冲突**
4. 使用@Slf4j注解, 记录每一步操作的日志,养成好习惯

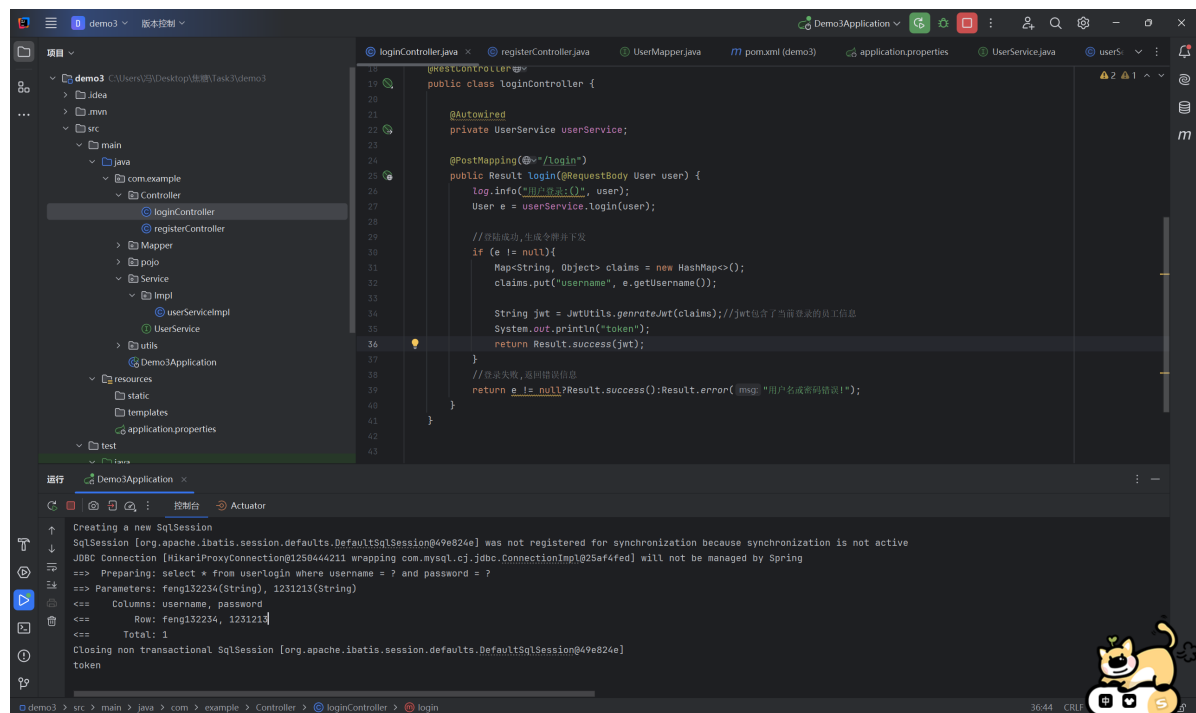
Controller层: 负责接收前端程序的请求(本例中用postman测试), 分为注册和登录两大控制器, 注册主要涉及到添加功能,声明insert方法;

登录功能中,使用getByUsernameAndPassword方法将传入的信息与数据库中的对比,并使用JWT令牌加密返回数据. 两者均有共同响应结果的格式

Service层: 负责具体的业务逻辑,本例中体现为将Controller传入的信息与数据库中的对比并返回结果

Mapper层: 数据持久层, 直接连接目标数据库,注解后添加sql语句来实现相关的CRUD功能, 注意 #{} 占位符的使用, 可一定程度上防范sql注入攻击, 增强安全性

控制台输出结果



The screenshot displays an IDE with two main panels. The top panel shows the source code for `loginController.java`. The code defines a `loginController` class with a `login` method that takes a `RequestBody` of type `User` and returns a `Result`. The method uses `userService.login(user)` to authenticate the user. If successful, it generates a JWT token using `JwtUtils.genrateJwt(claims)` and returns `Result.success(jwt)`. If the user is null, it returns `Result.error(msg: "用户名或密码错误!")`. The bottom panel shows the console output for the `Demo3Application`. It logs the creation of a new `SqlSession`, the preparation of a SQL query to select a user by username and password, and the execution of the query. The output shows the user `fengli32234` with password `1231213`. Finally, it logs the closing of the `SqlSession`.

```
loginController.java
registerController.java
UserMapper.java
pom.xml (demo3)
application.properties
UserService.java
userS...
utils
Demo3Application
resources
static
templates
application.properties
test

loginController.java
19 @RestController
20 public class loginController {
21
22     @Autowired
23     private UserService userService;
24
25     @PostMapping("/login")
26     public Result login(@RequestBody User user) {
27         log.info("用户登录:{}", user);
28         User e = userService.login(user);
29
30         //登录成功,生成令牌并下发
31         if (e != null) {
32             Map<String, Object> claims = new HashMap<>();
33             claims.put("username", e.getUsername());
34
35             String jwt = JwtUtils.genrateJwt(claims); //jwt包含了当前登录的员工信息
36             System.out.println("token");
37             return Result.success(jwt);
38         }
39         //登录失败,返回错误信息
40         return e != null ? Result.success() : Result.error(msg: "用户名或密码错误!");
41     }
42
43 }

控制台
Demo3Application
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@49e824e] was not registered for synchronization because synchronization is not active
JDBC Connection [HikariProxyConnection@1250444211 wrapping com.mysql.cj.jdbc.ConnectionImpl@25af4fed] will not be managed by Spring
==> Preparing: select * from userLogin where username = ? and password = ?
==> Parameters: fengli32234(String), 1231213(String)
<== Columns: username, password
<== Row: fengli32234, 1231213
<== Total: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@49e824e]
token
```

postman测试结果

