# 实验十一 SSM框架整合

## 实验目的

1. 掌握SSM框架整合环境的配置；
2. 掌握SSM框架整合应用程序的编写

## 实验内容

创建user表存储用户信息（id，姓名，密码），应用SSM框架整合，实现用户的注册与登录。

## 实验代码：

### Jdbc.xml

jdbc.driver=com.mysql.jdbc.Driver  
jdbc.url=jdbc:mysql://192.168.120.20:3306/spring?useSSL=false  
jdbc.username=root  
jdbc.password=root

### JdbcConfig.java

public class JdbcConfig {  
 @Value("${jdbc.driver}")  
 private String driver;  
 @Value("${jdbc.url}")  
 private String url;  
 @Value("${jdbc.username}")  
 private String username;  
 @Value("${jdbc.password}")  
 private String password;  
  
 @Bean  
 public DataSource dataSource(){  
 DruidDataSource dataSource = new DruidDataSource();  
 dataSource.setDriverClassName(driver);  
 dataSource.setUrl(url);  
 dataSource.setUsername(username);  
 dataSource.setPassword(password);  
 return dataSource;  
 }  
  
 @Bean  
 public PlatformTransactionManager transactionManager(DataSource dataSource){  
 DataSourceTransactionManager ds = new DataSourceTransactionManager();  
 ds.setDataSource(dataSource);  
 return ds;  
 }  
}

### MyBatisConfig.java

public class MyBatisConfig {  
 @Bean  
 public SqlSessionFactoryBean sqlSessionFactory(DataSource dataSource){  
 SqlSessionFactoryBean factoryBean = new SqlSessionFactoryBean();  
 factoryBean.setDataSource(dataSource);  
 factoryBean.setTypeAliasesPackage("com.shf.sy11.entity");  
 return factoryBean;  
 }  
  
 @Bean  
 public MapperScannerConfigurer mapperScannerConfigurer(){  
 MapperScannerConfigurer msc = new MapperScannerConfigurer();  
 msc.setBasePackage("com.shf.sy11.dao");  
 return msc;  
 }  
}

### ServletConfig.java

public class ServletConfig extends AbstractAnnotationConfigDispatcherServletInitializer {  
 *//加载Spring配置类* protected Class<?>[] getRootConfigClasses() {  
 return new Class[]{SpringConfig.class};  
 }  
 *//加载SpringMVC配置类* protected Class<?>[] getServletConfigClasses() {  
 return new Class[]{SpringMvcConfig.class};  
 }  
 *//设置SpringMVC请求地址拦截规则* protected String[] getServletMappings() {  
 return new String[]{"/"};  
 }  
 *//设置post请求中文乱码过滤器* @Override  
 protected Filter[] getServletFilters() {  
 CharacterEncodingFilter filter = new CharacterEncodingFilter();  
 filter.setEncoding("utf-8");  
 return new Filter[]{filter};  
 }  
}

### SpringConfig.java

@Configuration  
@ComponentScan({"com.shf.sy11.service"})  
@PropertySource("classpath:jdbc.properties")  
@Import({JdbcConfig.class,MyBatisConfig.class})  
@EnableTransactionManagement  
public class SpringConfig {  
}

### SpringMvcConfig.java

@Configuration  
@ComponentScan("com.shf.sy11.controller")  
@EnableWebMvc  
public class SpringMvcConfig extends WebMvcConfigurerAdapter {  
  
 @Bean  
 public InternalResourceViewResolver viewResolver() {  
 InternalResourceViewResolver resolver = new InternalResourceViewResolver();  
 resolver.setPrefix("/WEB-INF/pages/");  
 resolver.setSuffix(".jsp");  
 return resolver;  
 }  
  
 @Override  
 public void addInterceptors(InterceptorRegistry registry) {  
 *// addPathPatterns 是配置需要拦截的方法，如果遇到拦截不同的Controller，  
 // 我们可以使用不同的配置，比如：/user/\* 在这里是可以配置多个拦截器的。* registry.addInterceptor(new SubmitInterceptor()).addPathPatterns("/\*");  
 }  
  
 */\*配置静态资源的处理\*/* @Override  
 public void configureDefaultServletHandling(DefaultServletHandlerConfigurer configurer) {  
 configurer.enable();  
 }  
}

### User.java

@Data  
@ToString  
public class User {  
 private Integer id;  
 private String username;  
 private String password;  
}

### LoginDao.java

@Repository  
public interface LoginDao {  
 @Select("select *count*(*\**) from user where username=#{username} and password=#{password}")  
 int checkUserNameAndPassword(@Param("username") String username,  
 @Param("password") String password);  
  
 @Insert("insert into user (username,password) values (#{username},#{password});")  
 int insertUserNameAndPassword(@Param("username") String username,  
 @Param("password") String password);  
  
}

### LoginService.java

public interface LoginService {  
 int checkUserNameAndPassword(String username,String password);  
  
 int insertUserNameAndPassword(String username, String password);  
}

### LoginServiceImpl.java

@Service  
public class LoginServiceImpl implements LoginService {  
 @Autowired  
 private LoginDao loginDao;  
  
 @Override  
 public int checkUserNameAndPassword(String username,String password) {  
 return loginDao.checkUserNameAndPassword(username,password);  
 }  
  
 @Override  
 public int insertUserNameAndPassword(String username, String password) {  
 return loginDao.insertUserNameAndPassword(username, password);  
 }  
}

### SubmitInterceptor.java

public class SubmitInterceptor implements HandlerInterceptor {  
 @Override  
 public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {  
 request.setCharacterEncoding("UTF-8");  
 response.setCharacterEncoding("UTF-8");  
 response.setContentType("text/html;charset=UTF-8");  
  
 if(request.getRequestURI()=="/login"&&request.getMethod()=="POST" || request.getRequestURI()=="/register"&&request.getMethod()=="POST"){  
 if (request.getSession().getAttribute("flag")=="1"){  
 return true;  
 } else {  
 return false;  
 }  
 }  
 return true;  
 }  
  
 @Override  
 public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler, ModelAndView modelAndView) throws Exception {  
 HandlerInterceptor.super.postHandle(request, response, handler, modelAndView);  
 }  
  
 @Override  
 public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler, Exception ex) throws Exception {  
 HandlerInterceptor.super.afterCompletion(request, response, handler, ex);  
 }  
}

### LoginController.java

@Controller  
public class LoginController {  
 @Autowired  
 LoginService loginService;  
  
 @GetMapping("/login")  
 public String LoginPage(){  
 return "login";  
 }  
  
 @GetMapping("/register")  
 public String RegisterPage(){  
 return "register";  
 }  
  
 @PostMapping("/login")  
 public String Login(User user, Model model, HttpSession httpSession){  
 System.*out*.println(user);  
 int res = loginService.checkUserNameAndPassword(user.getUsername(), user.getPassword());  
 if (res>=1){  
 httpSession.setAttribute("flag", 1);  
 return "loginSuccess";  
 } else {  
 model.addAttribute("errcode","1");  
 return "redirect:/login";  
 }  
 }  
  
 @PostMapping("/register")  
 public String Register(User user,Model model ,HttpSession httpSession){  
 System.*out*.println(user);  
 int res = loginService.insertUserNameAndPassword(user.getUsername(), user.getPassword());  
 if (res>=1){  
 httpSession.setAttribute("flag", 1);  
 return "success";  
 }else {  
 model.addAttribute("errcode","1");  
 return "redirect:/register";  
 }  
  
 }  
}

### register.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <title>Title</title>  
</head>  
<body>  
<h1>  
 <h1>用户注册页面</h1>  
 <%="1".equalsIgnoreCase(request.getParameter("errcode"))?"注册失败，请重试":""%></h1>  
  
 <form action="/register" method="post">  
 用户名：<input type="text" name="username"> <br>  
 密码：<input type="password" name="password"> <br>  
 <input type="submit" value="提交">  
 </form>  
</body>  
</html>

### success.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <title>Title</title>  
</head>  
<body>  
 <h1>用户注册成功页面</h1>  
 <h1>提交成功</h1>  
 <h1>你的用户是：<%=request.getParameter("username")%></h1>  
 <h1><a href="/login">去登录页面</a></h1>  
  
</body>  
</html>

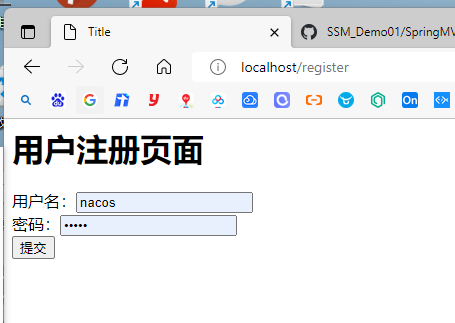
### login.jsp

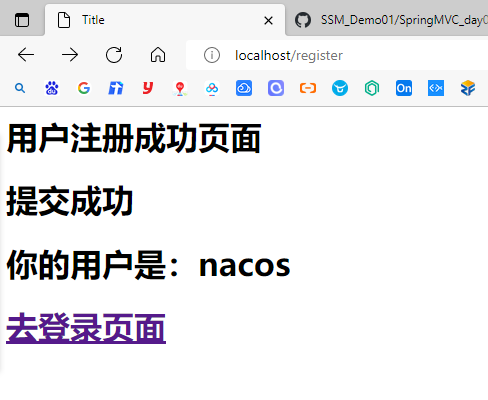
<%@ page language="java" import="java.util.\*" pageEncoding="utf-8"%>  
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
<html>  
<head>  
 <title>Title</title>  
 <meta charset="UTF-8">  
</head>  
<body>  
<h1>用户登录页面</h1>  
<h1>  
 <%="1".equalsIgnoreCase(request.getParameter("errcode"))?"用户名或密码错误":""%></h1>  
<form action="/login" method="post">  
 用户名：<input type="text" name="username"> <br>  
 密码：<input type="password" name="password"> <br>  
 <input type="submit" value="提交">  
 </form>  
</body>  
</html>

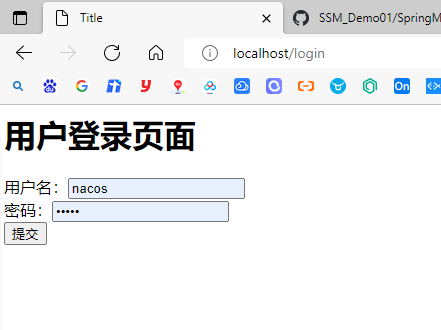
### loginSuccess.jsp

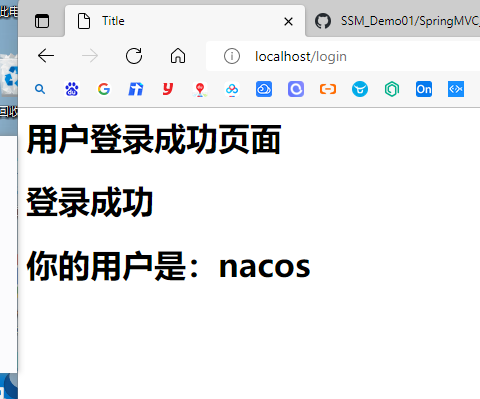
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <title>Title</title>  
</head>  
<body>  
<h1>用户登录成功页面</h1>  
<h1>登录成功</h1>  
<h1>你的用户是：<%=request.getParameter("username")%></h1>  
</body>  
</html>

## 运行截图









# 实验小结

通过本次实验我学会使用SSM进行框架整合，使用Druid数据源对数据进行增删改查，使用MySQL对数据进行持久化，使用mybatis注解版对数据进行CURD操作，使用SpringMVC进行数据的读取返回以及展示。