————————————————web.xml————————————————

```xml
<!DOCTYPE web-app PUBLIC
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
        <display-name>Archetype Created Web Application</display-name>

        <!--编码过滤器-->
        <filter>
                <filter-name>CharacterEncodingFilter</filter-name>
                <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
                <init-param>
                        <param-name>encoding</param-name>
                        <param-value>UTF-8</param-value>
                </init-param>
                <init-param>
                        <param-name>forceEncoding</param-name>
                        <param-value>true</param-value>
                </init-param>
        </filter>
        <filter-mapping>
                <filter-name>CharacterEncodingFilter</filter-name>
                <url-pattern>/</url-pattern>
        </filter-mapping>


        <!-- 加载spring上下文 -->
        <context-param>
                <param-name>contextConfigLocation</param-name>
                <param-value>classpath:spring/appContext-*.xml</param-value>
        </context-param>
        <listener>
                <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
        </listener>

        <!--首页-->
        <welcome-file-list>
                <welcome-file>index.jsp</welcome-file>
                <welcome-file>index.html</welcome-file>
        </welcome-file-list>

        <!--前端控制器-->
        <servlet>
                <servlet-name>aiairedian</servlet-name>
                <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
                <init-param>
                        <param-name>contextConfigLocation</param-name>
                        <param-value>classpath:spring/spring-*.xml</param-
value>
                </init-param>
        </servlet>
                <servlet-mapping>
                <servlet-name>aiairedian</servlet-name>
                <url-pattern>/</url-pattern>
        </servlet-mapping>

        <!--shiro过滤器-->
        <filter>
                <filter-name>shiroFilter</filter-name>
                <filter-
class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
                <init-param>
```

```xml
                <param-name>targetFilterLifecycle</param-name>
                <param-value>true</param-value>
            </init-param>
        </filter>

        <filter-mapping>
            <filter-name>shiroFilter</filter-name>
            <url-pattern>/admin/*</url-pattern>
        </filter-mapping>

</web-app>
```

————————————————appContext-mybatis.xml—————————————————

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:context="http://www.springframework.org/schema/context"
xmlns:p="http://www.springframework.org/schema/p"
       xmlns:aop="http://www.springframework.org/schema/aop" xmlns:tx="http://
www.springframework.org/schema/tx"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://
www.springframework.org/schema/beans/spring-beans-4.0.xsd
        http://www.springframework.org/schema/context http://
www.springframework.org/schema/context/spring-context-4.0.xsd
        http://www.springframework.org/schema/aop http://www.springframework.org/
schema/aop/spring-aop-4.0.xsd http://www.springframework.org/schema/tx http://
www.springframework.org/schema/tx/spring-tx-4.0.xsd
        http://www.springframework.org/schema/util http://www.springframework.org/
schema/util/spring-util-4.0.xsd">

            <!-- DRUID是阿里巴巴开源平台上一个数据库连接池实现 -->
            <!-- 加载配置文件 -->
            <context:property-placeholder order="1"
location="classpath:*.properties" />
            <!-- 配置数据库连接池 -->
            <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource"
destroy-method="close">
                    <property name="driverClassName"
value="com.mysql.jdbc.Driver" />
                    <property name="url" value="${mysql.url}" />
                    <property name="username" value="${mysql.username}" />
                    <property name="password" value="${mysql.password}" />
                    <property name="maxActive" value="10" />
                    <property name="minIdle" value="5" />
            </bean>

            <!-- MyBatis，配置sqlsessionFactory -->
            <bean id="sqlSessionFactory"
class="org.mybatis.spring.SqlSessionFactoryBean">
                    <property name="configLocation" value="classpath:mybatis/
mybatis-config.xml"></property>
                    <property name="dataSource" ref="dataSource"></property>
            </bean>
            <!-- 配置扫描包，加载mapper接口 -->
            <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
                    <property name="basePackage" value="com.aiairedian.dao"></
property>
            </bean>


</beans>
```

————————————————spring-mvc.xml————————————————

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-4.2.xsd
        http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc-4.2.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context.xsd
        http://www.springframework.org/schema/aop
        http://www.springframework.org/schema/aop/spring-aop-4.2.xsd
        http://www.springframework.org/schema/tx
        http://www.springframework.org/schema/tx/spring-tx.xsd">

        <!-- 扫描所有的Controller -->
        <context:component-scan base-package="com.aiairedian.controller"> </
context:component-scan>

        <mvc:resources mapping="/css/**" location="/resource/css/"> </
mvc:resources>
        <mvc:resources mapping="/js/**" location="/resource/js/"> </
mvc:resources>
        <mvc:resources mapping="/img/**" location="/resource/img/"> </
mvc:resources>

        <!-- 一个配置节解决映射器和适配器的配置注解配置 -->
        <mvc:annotation-driven> </mvc:annotation-driven>

        <!-- 配置视图解析器
        -->
        <bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver" >
                <!--配置前缀和后缀，也可以不指定-->
                <property name="prefix" value="/WEB-INF/jsp/"/>
                <property name="suffix" value=".jsp"/>
        </bean>

</beans>
```

————————————————mybatis-config.xml————————————————

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
        PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
        <mappers>
                <mapper resource="mapper/aiaiPostsMapper.xml" />
                <mapper resource="mapper/aiaiUsersMapper.xml" />
        </mappers>
</configuration>
```

————————————————appContext-shiro.xml————————————————

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://
www.springframework.org/schema/beans/spring-beans.xsd">
```

```xml
        <bean id="lifecycleBeanPostProcessor"
class="org.apache.shiro.spring.LifecycleBeanPostProcessor"> </bean>

        <!-- 缓存管理 -->
        <bean id="cacheManager"
class="org.apache.shiro.cache.MemoryConstrainedCacheManager"> </bean>

        <!-- 授权 认证 -->
        <bean id="aiaiUserRealm" class="com.aiairedian.util.UserRealm">
                <property name="credentialsMatcher" ref="credentialsMatcher"/>
        </bean>

        <!-- 加密算法 -->
        <bean id="credentialsMatcher"

class="org.apache.shiro.authc.credential.HashedCredentialsMatcher">
                <property name="hashAlgorithmName" value="md5" />
                <property name="hashIterations" value="1" />
        </bean>

        <!-- Shiro安全管理器 -->
        <bean id="securityManager"
class="org.apache.shiro.web.mgt.DefaultWebSecurityManager">
                <property name="realm" ref="aiaiUserRealm"> </property>
                <property name="cacheManager" ref="cacheManager"> </property>
        </bean>

        <!--shiro过滤器-->
        <bean id="shiroFilter"
class="org.apache.shiro.spring.web.ShiroFilterFactoryBean">
                <property name="securityManager" ref="securityManager"> </property>
                <property name="loginUrl" value="/admin"> </property>
                <property name="successUrl" value="/admin/successUrl"> </property>
                <!-- 用户访问未对其授权的资源时，所显示的连接 -->
                <property name="unauthorizedUrl" value="/admin/unauthorizedUrl"> </
property>
                <property name="filterChainDefinitions">
                        <value>
                                /admin/permission=roles[admin]
                                /admin/**=user
                        </value>
                </property>
        </bean>

</beans>
```

————————————————自定义realm————————————————
```java
package com.aiairedian.util;

import com.aiairedian.dao.aiaiUsersMapper;
import com.aiairedian.dao.aiaiRoleMapper;
import org.apache.shiro.SecurityUtils;
import org.apache.shiro.authc.*;
import org.apache.shiro.authz.AuthorizationInfo;
import org.apache.shiro.authz.SimpleAuthorizationInfo;
import org.apache.shiro.realm.AuthorizingRealm;
import org.apache.shiro.subject.PrincipalCollection;
import org.springframework.beans.factory.annotation.Autowired;

import java.util.HashSet;
import java.util.Set;

public class UserRealm extends AuthorizingRealm
{
    @Autowired(required = false)
```

```java
    private aiaiUsersMapper usersMapper;

    @Autowired(required = false)
    private aiaiRoleMapper roleMapper;

    private String username;

    @Override
    protected AuthorizationInfo doGetAuthorizationInfo(PrincipalCollection arg0)
    {
        String username =
((UsernamePasswordToken)arg0.getPrimaryPrincipal()).getUsername();
        // 从数据库中查找该用户的角色和权限
        SimpleAuthorizationInfo sainfo = new SimpleAuthorizationInfo();
        String userRole=usersMapper.selectByLoginName(username).getUser_nicename();
        sainfo.addRole(userRole);
        return sainfo;
    }

    @Override
    protected AuthenticationInfo doGetAuthenticationInfo(AuthenticationToken arg0)
            throws AuthenticationException
    {
        try {
            username = arg0.getPrincipal().toString();
        }catch (NullPointerException e){
            return null;
        }
        // 根据username从数据库查找用户，得到密码
        // 假设找到的用户如下
        // User user = userService.findByUsername(username)
        String userPass = usersMapper.selectByLoginName(username).getUser_pass();
        return new SimpleAuthenticationInfo(arg0, userPass, username);
    }

}
```