客户名字和logo

餐厅订餐系统—系统详细设计文档

Xu LIN

日期：2014-12-30

文档版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 修订日期 | 修订人 | 审核人 | 变更内容 |
| 1.0 | 2014-11-23 | Xu Lin |  | 初始化文档结构 |
| 1.1 | 2014-12-5 | Xu Lin |  | 新增用户管理模块和数据库 |
| 1.2 | 2014-12-8 | Xu Lin |  | 用户管理模块的用户需求更新和新增类说明 |
| 1.3 | 2014-12-26 | LUO ZHI |  | 1. 增加章节2 开发环境配置大纲 2. 增加章节3项目目录结构说明大纲 3. 增加章节4用户管理模块大纲 |
| 1.4 | 2014/12/30 | Xu Lin |  | 增加章节4内容 |

# 目录

[目录 3](#_Toc407740679)

[1 文档介绍 1](#_Toc407740680)

[1.1 目的 1](#_Toc407740681)

[1.2 范围 1](#_Toc407740682)

[1.3 缩写词列表 1](#_Toc407740683)

[1.4 参考内容 1](#_Toc407740684)

[2 开发环境配置 2](#_Toc407740685)

[2.1 下载及安装JDK 2](#_Toc407740686)

[2.2 下载及安装Eclipse 2](#_Toc407740687)

[2.3 下载及安装Maven 2](#_Toc407740688)

[2.4 下载及安装MySQL 2](#_Toc407740689)

[2.5 下载及安装Tomcat 2](#_Toc407740690)

[2.6 下载及安装Git 2](#_Toc407740691)

[2.7 导入项目 2](#_Toc407740692)

[2.8 编译项目 2](#_Toc407740693)

[3 项目目录结构说明 3](#_Toc407740694)

[4 用户管理模块 4](#_Toc407740695)

[4.1 用户登录 4](#_Toc407740696)

[4.1.1 代码构成 4](#_Toc407740697)

[4.1.2 类说明 4](#_Toc407740698)

[4.1.3 配置文件说明 7](#_Toc407740699)

[4.1.4 数据库脚本说明 14](#_Toc407740700)

[4.1.5 时序图 14](#_Toc407740701)

[5 模块设计 15](#_Toc407740702)

[5.1 功能模板设计说明 15](#_Toc407740712)

[5.2 用户管理 15](#_Toc407740713)

[5.2.1 目标 15](#_Toc407740714)

[5.2.2 用户需求 15](#_Toc407740715)

[5.2.3 类图 18](#_Toc407740716)

[5.2.4 时序图 18](#_Toc407740717)

[5.2.5 组件 20](#_Toc407740718)

[5.2.6 类说明 20](#_Toc407740719)

[5.2.7 其他说明 24](#_Toc407740720)

[6 数据库设计 24](#_Toc407740721)

[6.1 用户表 24](#_Toc407740722)

[6.2 角色表 24](#_Toc407740723)

[6.3 功能表 25](#_Toc407740724)

[6.4 用户角色关联表 25](#_Toc407740725)

[6.5 角色功能关联表 25](#_Toc407740726)

[6.6 菜品分类表 25](#_Toc407740727)

[6.7 菜品表 26](#_Toc407740728)

[6.8 订单表 26](#_Toc407740729)

[6.9 订单条目表 27](#_Toc407740730)

[6.10 会员等级表 27](#_Toc407740731)

[7 接口设计 27](#_Toc407740732)

[7.1 内部接口 27](#_Toc407740738)

[7.2 外部接口 27](#_Toc407740739)

[7.2.1 接口说明 27](#_Toc407740740)

[7.2.2 调用方式 27](#_Toc407740741)

[8 系统认证安全设计 27](#_Toc407740742)

[8.1 访问控制设计 27](#_Toc407740743)

[8.2 访问控制流程图 27](#_Toc407740744)

[8.3 访问控制流程说明 27](#_Toc407740745)

[8.4 系统性能设计 28](#_Toc407740746)

[8.5 概念 28](#_Toc407740747)

[8.6 EnCache配置 28](#_Toc407740748)

[9 系统出错处理 28](#_Toc407740749)

[9.1 出错信息输出 28](#_Toc407740750)

[9.2 出错处理对策 28](#_Toc407740751)

[10 附录 28](#_Toc407740752)

# 文档介绍

## 目的

## 范围

## 缩写词列表

## 参考内容

《餐厅订餐系统\_系统需求文档.docx》

# 开发环境配置

## 下载及安装JDK

## 下载及安装Eclipse

## 下载及安装Maven

## 下载及安装MySQL

## 下载及安装Tomcat

## 下载及安装Git

## 导入项目

## 编译项目

# 项目目录结构说明

# 用户管理模块

## 用户登录

用户使用系统合法的用户名和密码登录到系统。对于不正确的用户名和密码信息进行提示。详细需求说明见UC\_005-用户登录

### 代码构成

|  |  |  |
| --- | --- | --- |
| 文件名 | 文件路径 | 文件说明 |
| BaseVO.java | \java\com\innovaee\eorder\module \vo | 所有VO（值对象）类的父类源文件 |
| RoleLinkVo.java | \java\com\innovaee\eorder\module \vo | 页面链接值对象 |
| BaseService.java | \java\com\innovaee\eorder\module  \service | 所有Servicwe类的基类 |
| AuthenticationService  .java | \java\com\innovaee\eorder\module  \service\security | 验证服务 |
| AuthorizationService  .java | \java\com\innovaee\eorder\module  \service\security | 授权服务 |
| SecurityMetadataSource Service.java | \java\com\innovaee\eorder\module  \service\security | 安全元数据源服务 |
| BaseAction.java | \java\com\innovaee\eorder\web\action | 所有Action类的基类 |
| LoginAction.java | \java\com\innovaee\eorder\web\action \login | 登录模块的Action |
| struts.xml | com\innovaee\eorder\web\action \login | 登录模块的Action配置文件 |
| login.jsp | eorder\src\main\webapp\pages\login | 登录页面 |

### 类说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.vo | | |
| 类名 | BaseVo | | |
| 父类 | n/a | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| n/a |  |  | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| copy(Object) | void | Object orig | 将传入对象的属性值复制到本对象 |
| equals(Object) | boolean | Object obj | 指示其他某个对象是否与此对象“相等”。 |
| hashCode() | int | n/a | 返回该对象的哈希码值。 |
| toString() | String | n/a | 返回该对象的字符串表示。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.vo | | |
| 类名 | RoleLinkVo | | |
| 父类 | com.innovaee.eorder.module.vo. BaseVo | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| order | String | 菜单项排序号 | |
| id | Integer | 菜单ID | |
| name | String | 菜单名 | |
| functionName | String | 功能名称 | |
| link | String | 链接地址 | |
| flag | String | 菜单等级，1为父节点，2为子节点 | |
| list | List<RoleLinkVo> | 子节点菜单列表 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| n/a |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service | | |
| 类名 | BaseService | | |
| 父类 | n/a | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| n/a |  |  |  |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| n/a |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service.security | | |
| 类名 | AuthenticationService | | |
| 父类 | com.innovaee.eorder.module.service. BaseService | | |
| 实现接口 | org.springframework.security.core.userdetails.UserDetailsService | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象 | |
| securityMetadataSourceService | SecurityMetadataSourceService | 安全元数据源服务对象 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getUserInfo(String) | User | String username | 通过用户名查找用户对象。 |
| checkUserPassword(String, String) | boolean | String username, String password | 校验用户名密码是否匹配，如果匹配成功，返回true，否则返回false。 |
| loadUserByUsername(String) | UserDetails | String username | 通过用户名查找用户详细信息对象。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service.security | | |
| 类名 | AuthorizationService | | |
| 父类 | com.innovaee.eorder.module.service. BaseService | | |
| 实现接口 | org.springframework.security.access.AccessDecisionManager | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象 | |
| securityMetadataSourceService | SecurityMetadataSourceService | 安全元数据源服务对象 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| decide | void | Authentication authentication, Object object, Collection<ConfigAttribute> configAttributes | AccessDecisionManager 使用方法参数传递所有信息，在真实的安全方法期望调用的时候，传递安全Object启用那些参数。如果访问被拒绝，实现将抛  出一个AccessDeniedException 异常。 |
| supports | boolean | ConfigAttribute attribute | 在启动的时候被AbstractSecurityInterceptor调用， 来决定AccessDecisionManager 是否可以执行传递ConfigAttribute 。 |
| supports | boolean | Class<?> clazz | 此方法被安全拦截器实现调用， 包含安全拦截器将显示的AccessDecisionManager 支持安全对象的类型。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service.security | | |
| 类名 | SecurityMetadataSourceService | | |
| 父类 | com.innovaee.eorder.module.service. BaseService | | |
| 实现接口 | org.springframework.security.web.access.intercept.FilterInvocationSecurityMetadataSource | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象 | |
| securityMetadataSourceService | SecurityMetadataSourceService | 安全元数据源服务对象 | |
| functionDao | FunctionDao | 功能数据访问对象 | |
| roleDao | RoleDao | 角色数据访问对象 | |
| userRoleDao | UserRoleDao | 用户角色数据访问对象 | |
| roleFunctionDao | RoleFunctionDao | 角色功能数据访问对象 | |
| allConfigAttributes | Collection<ConfigAttribute> | 配置属性集合 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getAllFunctions | List<Function> | n/a | 返回所有功能列表 |
| getUserRoles | List<UserRole> | String username | 根据用户名称返回用户角色列表 |
| findRoleFunctionsByRoleId | List<RoleFunction> | Integer roleId | 根据用户角色ID查找角色功能列表 |
| getUserFunctions | List<UserFunctionVo> | String username | 根据用户名称查找用户功能值对象列表 |

### 配置文件说明

struts.xml

<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"

"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>

<package name="com.innovaee.eorder.web.action.login" namespace="/login"

extends="struts-base">

<action name="login" class="com.innovaee.eorder.web.action.login.LoginAction"

method="login">

<result name="success">/pages/login/login.jsp</result>

</action>

<action name="doLogin" class="com.innovaee.eorder.web.action.login.LoginAction"

method="doLogin">

<result type="redirect">../user/doUser.action</result>

</action>

<action name="doLogout" class="com.innovaee.eorder.web.action.login.LoginAction"

method="doLogout">

<result name="success">/pages/main/login.jsp</result>

</action>

</package>

</struts>

applicationContext-security.xml

<?xml version="1.0" encoding="UTF-8"?>

<beans:beans xmlns="http://www.springframework.org/schema/security"

xmlns:beans="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-3.0.xsd

http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-security-3.0.xsd">

<global-method-security pre-post-annotations="enabled" />

<beans:bean id="myFilter"

class="org.springframework.security.web.access.intercept.FilterSecurityInterceptor">

<beans:property name="authenticationManager" ref="authenticationManager" />

<beans:property name="accessDecisionManager" ref="authorizationService" />

<beans:property name="securityMetadataSource" ref="securityMetadataSourceService" />

</beans:bean>

<http auto-config="true" use-expressions="true" access-denied-page="/pages/global/accessDenied.jsp">

<form-login login-page="/login/login.action"

authentication-failure-url="/login/login.action?error=true"

default-target-url="/login/doLogin.action" />

<logout logout-success-url="/default.html" />

<remember-me />

<session-management invalid-session-url="/default.html">

<concurrency-control max-sessions="1"

error-if-maximum-exceeded="true" />

</session-management>

<intercept-url pattern="/resources/\*\*" filters="none" />

<intercept-url pattern="/default.html" filters="none" />

<intercept-url pattern="/login/login.action" filters="none" />

<custom-filter ref="myFilter" before="FILTER\_SECURITY\_INTERCEPTOR" />

</http>

<authentication-manager alias="authenticationManager">

<authentication-provider user-service-ref="authenticationService">

<password-encoder hash="md5">

<salt-source user-property="username" />

</password-encoder>

</authentication-provider>

</authentication-manager>

<beans:bean id="messageSource"

class="org.springframework.context.support.ReloadableResourceBundleMessageSource">

<beans:property name="basename"

value="classpath:org/springframework/security/messages\_zh\_CN" />

</beans:bean>

</beans:beans>

applicationContext-total.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:aop="http://www.springframework.org/schema/aop" xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="

http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd

http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.0.xsd

http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.0.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd"

default-autowire="byName">

<!-- support spring annotation -->

<context:annotation-config />

<bean id="databaseConfigurer"

class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

<property name="locations">

<list>

<value>classpath:database.properties</value>

</list>

</property>

</bean>

<!-- 配置数据源 -->

<!-- data source 基本的连接池 -->

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"

destroy-method="close">

<property name="driverClass" value="com.mysql.jdbc.Driver" />

<property name="jdbcUrl" value="${url}" />

<property name="user" value="${username}" />

<property name="password" value="${password}" />

<property name="maxPoolSize" value="20" />

<property name="minPoolSize" value="5" />

<property name="maxStatements" value="200" />

<property name="initialPoolSize" value="5" />

<property name="maxIdleTime" value="15" />

<property name="idleConnectionTestPeriod" value="5" />

<property name="testConnectionOnCheckin" value="false" />

<property name="testConnectionOnCheckout" value="false" />

<property name="preferredTestQuery" value="SELECT 1 FROM DUAL" />

</bean>

<!-- Hibernate SessionFactory -->

<bean id="sessionFactory"

class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">

<property name="dataSource" ref="dataSource" />

<property name="packagesToScan">

<list>

<value>com.innovaee.eorder.module.entity</value>

<value>com.innovaee.eorder.module.utils.log</value>

<value>cn.net.msg.model</value>

</list>

</property>

<property name="hibernateProperties">

<props>

<prop key="hibernate.dialect">

${hibernate.dialect}

</prop>

<prop key="hibernate.show\_sql">

${hibernate.show\_sql}

</prop>

<prop key="hibernate.format\_sql">

${hibernate.format\_sql}

</prop>

<prop key="hibernate.use\_sql\_comments">

${hibernate.use\_sql\_comments}

</prop>

<prop key="hibernate.hbm2ddl.auto">

${hibernate.hbm2ddl.auto}

</prop>

<prop key="hibernate.default\_batch\_fetch\_size">

${hibernate.default\_batch\_fetch\_size}

</prop>

<prop key="hibernate.max\_fetch\_depth">

${hibernate.max\_fetch\_depth}

</prop>

<prop key="hibernate.jdbc.batch\_size">

${hibernate.jdbc.batch\_size}

</prop>

<prop key="hibernate.query.factory\_class">

${hibernate.query.factory\_class}

</prop>

<prop key="hibernate.connection.useUnicode">

${hibernate.connection.useUnicode}

</prop>

<prop key="hibernate.connection.characterEncoding">

${hibernate.connection.characterEncoding}

</prop>

<prop key="hibernate.connection.pool\_size">

${hibernate.connection.pool\_size}

</prop>

<prop key="hibernate.cache.provider\_class">

${hibernate.cache.provider\_class}

</prop>

<prop key="hibernate.autoReconnect">true</prop>

</props>

</property>

</bean>

<bean id="txManager" class="org.springframework.orm.hibernate3.HibernateTransactionManager">

<property name="sessionFactory" ref="sessionFactory" />

</bean>

<tx:advice id="txAdvice" transaction-manager="txManager">

<tx:attributes>

<tx:method name="get\*" propagation="SUPPORTS" read-only="true" />

<tx:method name="find\*" propagation="SUPPORTS" read-only="true" />

<tx:method name="query\*" propagation="SUPPORTS" read-only="true" />

<tx:method name="\*" propagation="REQUIRED" rollback-for="java.lang.Throwable" />

</tx:attributes>

</tx:advice>

<aop:config proxy-target-class="false">

<aop:pointcut id="txPointcut"

expression="execution(\* com.innovaee.eorder.module.service..\*.\*(..))" />

<aop:advisor advice-ref="txAdvice" pointcut-ref="txPointcut" />

</aop:config>

<bean id="loggerUtility" class="com.innovaee.eorder.module.utils.log.LoggerUtility" />

<bean id="loggerUtilInterceptor"

class="com.innovaee.eorder.module.utils.interceptor.LoggerUtilInterceptor" />

<aop:config proxy-target-class="true">

<aop:aspect id="loggerAspect" ref="loggerUtilInterceptor">

<aop:pointcut id="logPointcut"

expression="execution(\* com.innovaee.eorder.module..\*.\*(..))" />

<aop:before pointcut-ref="logPointcut" method="startInvoke" />

<aop:after pointcut-ref="logPointcut" method="endInvoke" />

</aop:aspect>

</aop:config>

<bean id="baseDAO" class="com.innovaee.eorder.module.dao.BaseDao"

abstract="true">

<property name="sessionFactory" ref="sessionFactory" />

</bean>

<bean id="roleFunctionDao" class="com.innovaee.eorder.module.dao.RoleFunctionDao"

parent="baseDAO" />

<bean id="functionDao" class="com.innovaee.eorder.module.dao.FunctionDao"

parent="baseDAO" />

<bean id="roleDao" class="com.innovaee.eorder.module.dao.RoleDao" parent="baseDAO" />

<bean id="userDao" class="com.innovaee.eorder.module.dao.UserDao" parent="baseDAO" />

<bean id="userRoleDao" class="com.innovaee.eorder.module.dao.UserRoleDao"

parent="baseDAO" />

<bean id="userLevelDao" class="com.innovaee.eorder.module.dao.UserLevelDao"

parent="baseDAO" />

<bean id="baseService" class="com.innovaee.eorder.module.service.BaseService"

abstract="true" />

<bean id="authorizationService"

class="com.innovaee.eorder.module.service.security.AuthorizationService"

parent="baseService" />

<bean id="authenticationService"

class="com.innovaee.eorder.module.service.security.AuthenticationService"

parent="baseService" />

<bean id="securityMetadataSourceService"

class="com.innovaee.eorder.module.service.security.SecurityMetadataSourceService"

parent="baseService" />

<bean id="userService" class="com.innovaee.eorder.module.service.UserService"

parent="baseService" />

</beans>

### 数据库脚本说明

-- ----------------------------

-- Table structure for `t\_user`

-- ----------------------------

DROP TABLE IF EXISTS `t\_user`;

CREATE TABLE `t\_user` (

`user\_id` int(32) NOT NULL AUTO\_INCREMENT,

`username` varchar(20) COLLATE utf8\_bin DEFAULT NULL,

`password` varchar(256) COLLATE utf8\_bin DEFAULT NULL,

`cellphone` varchar(16) COLLATE utf8\_bin DEFAULT NULL,

`user\_score` int(32) DEFAULT NULL,

`level\_id` int(32) DEFAULT NULL,

`user\_status` tinyint(1) DEFAULT '1',

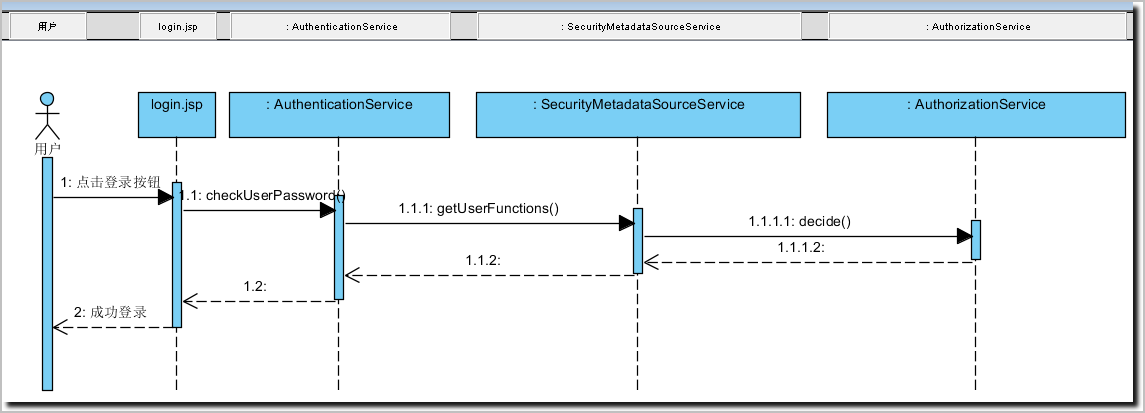
`create\_at` datetime DEFAULT NULL,

`update\_at` datetime DEFAULT NULL,

PRIMARY KEY (`user\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8 COLLATE=utf8\_bin COMMENT='user''s basic information';

### 时序图



# 模块设计



## 功能模板设计说明

## 用户管理

### 目标

管理员通过用户管理模块，对用户进行增删改查操作。

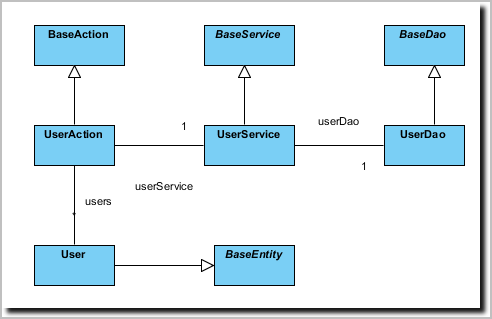
### 用户需求

|  |  |
| --- | --- |
| 用例编号 | UC\_006 |
| 用例描述 | 用户管理 |
| 用例来源 | UR\_010 |
| 用例用户 | USR\_05 –系统管理员 |
| 用户界面 |  |
| 用例图 |  |
| 触发条件 | 管理员得到用户信息更新需求 |
| 前置条件 | UC\_005 |
| 后置条件 | n/a |
| 主流程 | n/a |
| 备选流程 | n/a |
| 例外 | n/a |
| 包含用例 | UC\_007,UC\_008 |
| 非功能性需求 | n/a |
| 其他说明 | n/a |

|  |  |
| --- | --- |
| 用例编号 | UC\_007 |
| 用例描述 | 用户管理-创建用户 |
| 用例来源 | UR\_010 |
| 用例用户 | USR\_05 –系统管理员 |
| 用户界面 | n/a |
| 用例图 |  |
| 触发条件 | 管理员得到增加用户信息需求 |
| 前置条件 | UC\_005 |
| 后置条件 | n/a |
| 主流程 | 管理员点击eOrder订餐管理系统后台页面左侧用户管理菜单，进入用户管理页面  页面显示系统所有的用户信息列表  在增加新用户的页面内，输入用户信息  管理员点击增加创建新用户按钮提交新用户信息，系统显示Msg701  系统显示用户加入了用户列表 |
| 备选流程 | 管理员没输入所有的用户信息，系统提示Msg702  回到主流程4 |
| 例外 | 员工号重复，提示Msg703  手机号重复，提示Msg704 |
| 包含用例 | n/a |
| 非功能性需求 | 用户列表按照用户拼音字母a-z显示 |
| 其他说明 | 用户信息包括：用户名（用户名为姓名的拼音，忽略大小写，相同用户的加数字区分），员工号（为5位数字），手机号，密码（默认密码为手机号后6位），确认密码，用户状态（在职，离职） |

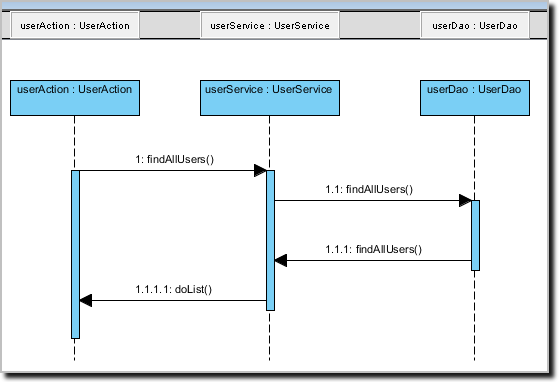
|  |  |
| --- | --- |
| 用例编号 | UC\_008 |
| 用例描述 | 用户管理-更新用户 |
| 用例来源 | UR\_010 |
| 用例用户 | USR\_05 –系统管理员 |
| 用户界面 | n/a |
| 用例图 |  |
| 触发条件 | 管理员得到用户信息更新需求 |
| 前置条件 | UC\_005 |
| 后置条件 | n/a |
| 主流程 | 管理员进入用户信息列表  管理员点击需要更新的用户名对应的编辑按钮  在用户信息的页面内，输入更新需求的用户信息  管理员点击更新用户信息页面的完成按钮提交用户更新信息，系统显示Msg801  系统显示更新的用户信息 |
| 备选流程 | 用户点击删除按钮，系统不显示该用户信息  回到主流程1  4.1管理员删除了用户的信息某一个或者多个字段的值，系统提示Msg802  回到主流程4  管理员提交更新不成功，系统提示Msg803  回到主流程4 |
| 例外 | n/a |
| 包含用例 | n/a |
| 非功能性需求 | 用户列表按照拼音显示 |
| 其他说明 | 用户信息包括：用户名（用户名为姓名的拼音），手机号，密码（默认密码为手机号后6位），用户角色 |

### 类图

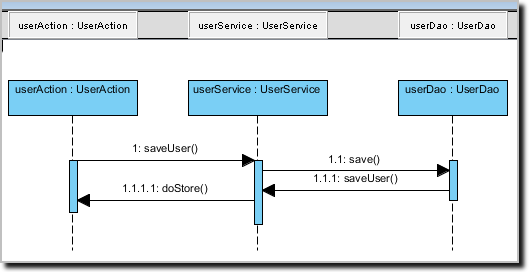


### 时序图

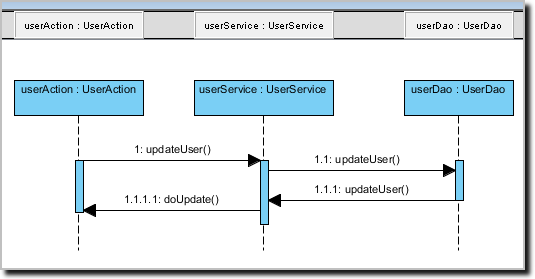
1、查找用户



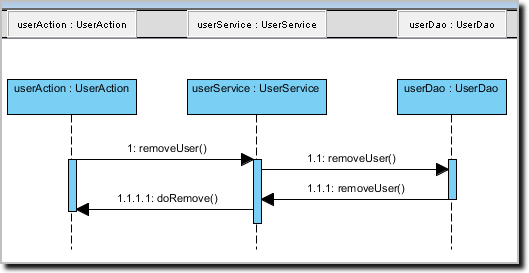
2、新增用户



3、修改用户



4、删除用户



### 组件

* Entity层实现对数据表及关联关系进行封装 ,命名就是数据库表的名字，遵循Java的编码规范；
* Dao接口层实现对数据访问层封装；
* Sevice接口层实现对业务逻辑封装；
* Action层实现对业务事件的封装。

### 类说明

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.entity | | |
| 类名 | User | | |
| 父类 | com.innovaee.eorder.module.entity.BaseEntity | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userId | Integer | 用户ID | |
| username | String | 用户名 | |
| password | String | 密码 | |
| cellphone | String | 手机号码 | |
| levelId | Integer | 用户等级 | |
| userStatus | Boolean | 用户状态 | |
| createAt | Timestamp | 创建时间 | |
| updateAt | Timestamp | 更新时间 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getPK | Serializable | n/a | 用户ID |
| User | n/a | n/a | 默认构造函数 |
| User | n/a | Integer userId | 构造一个新的 User，它具有指定的userId。 |
| User | n/a | String username | 构造一个新的 User，它具有指定的username。 |
| User | n/a | String username, String password, String cellphone, Integer levelId, Boolean userStatus, Timestamp createAt | 构造一个新的 User，它具有指定的username，password，cellphone，levelId，userStatus，createAt。 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.dao | | |
| 类名 | UserDao | | |
| 父类 | com.innovaee.eorder.module.dao.BaseDao | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| getEntityClass() | Class | n/a |  |
| findUsersByUserName() | User | String username | 通过用户名查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| findAllUsers() | List<User> | n/a | 返回所有用户实体对象列表。 |
| getUserByPassword(String, String) |  | String username,  String password | 通过用户名和密码查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| saveUser(User) | User | User | 将传入的User对象写入数据库 |
| updateUser(User) | void | User | 更新数据库中传入的User对象对应的记录 |
| removeUser(User) | void | User | 删除数据库中传入的User对象对应的记录 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.module.service | | |
| 类名 | UserService | | |
| 父类 | com.innovaee.eorder.module.service.BaseService | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| userDao | UserDao | 用户数据访问对象类，用于对用户数据操作 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| loadUser | User | Integer userId | 通过用户ID查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| findAllUsers() | List<User> | n/a | 返回所有用户实体对象列表。 |
| getUserByPassword(String, String) | User | String username,  String password | 通过用户名和密码查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| findUsersByUserName(String) | User | String username | 通过用户名查找指定用户，如果用户存在，返回用户实体对象;如果用户不存在，返回空对象。 |
| saveUser(User) | User | User | 将传入的User对象新增到数据库中 |
| updateUser(User) | void | User | 更新数据库中传入的User对象对应的记录 |
| removeUser(User) | void | User | 删除数据库中传入的User对象对应的记录 |
| removeUser(Integer) | void | Integer userId | 根据userId删除数据库中的对应的User记录 |
| removeUsers(String[]) | void | Integer[] userId | 根据userId数组，批量删除数据库中的对应的User记录 |

|  |  |  |  |
| --- | --- | --- | --- |
| 包名 | com.innovaee.eorder.web.action.admin.user | | |
| 类名 | UserAction | | |
| 父类 | com.innovaee.eorder.web.action.BaseAction | | |
| 实现接口 | n/a | | |
| 成员变量 | | | |
| 变量名 | 数据类型 | 说明 | |
| list | List<RoleLinkVo> | 用户角色链接列表，用于显示用户可以看到的菜单 | |
| userId | String | 用户ID | |
| userIds | String[] | 用户ID数组 | |
| user | User | User对象，用户前台页面和后台数据交互 | |
| users | List<User> | User对象列表，用户前台页面和后台数据交互 | |
| userService | UserService | 用户服务类，用于对User对象的操作 | |
| userRoleService | UserRoleService | 用户角色服务类，用于对UserRole对象的操作 | |
| 成员函数 | | | |
| 函数名 | 返回值 | 参数 | 说明 |
| login() | String | n/a | 进入页面执行的方法。 |
| doLoad() | String | n/a | 根据前台传来的用户ID查到User对象，并保存到user对象中。 |
| doList() | List<User> | n/a | 返回所有用户实体对象列表。 |
| doStore() | User | User | 将传入的User对象新增到数据库中 |
| doUpdate() | void | User | 更新数据库中传入的User对象对应的记录 |
| doRemove() | void | User | 删除数据库中传入的User对象对应的记录 |
| doLeft() | String | n/a | 根据用户信息，生成对应的菜单 |

### 其他说明

# 数据库设计

## 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 用户表 | | t\_user | |
| 数据项描述 | 数据项名 | 类型(长度) | 默认值 | 限制 |
| 用户ID | User\_id | Int(32) | n/a | 主键，自动增加 |
| 用户名 | username | Varchar(128)r | n/a | 非空，英文字母和数字组成不少于6位 |
| 密码 | password | Varchar(128) | n/a | 非空，英文字母和数字组成不少于6位 |
| 手机号 | cellphone | Varchar(16) | n/a |  |
| 会员等级ID | level\_id | Int(32) | n/a |  |
| 用户状态 | User\_status | tinyint | 1 |  |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 角色表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 角色表 | | t\_role | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 角色ID | role\_id | Int(32) | n/a | 主键 |
| 角色名称 | role\_name | Varchar(128) | n/a |  |
| 角色描述 | role\_desc | Varchar(128) | n/a |  |
| 角色状态 | role\_status | tinyint | 1 |  |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 功能表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 功能表 | | t\_function | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 功能ID | function\_id | Int(32) | n/a | 主键 |
| 功能名称 | function\_name | Varchar(128) | n/a | 非空 |
| 功能描述 | function\_desc | Varchar(128) | n/a | 非空 |
| 功能路径 | function\_path | Varchar(128) | n/a | 非空 |
| 父功能ID | function\_parent | Int(32) | n/a | 可为空 |
| 功能排序 | function\_order | Varchar(128) | n/a | 非空 |
| 功能状态 | function\_status | tinyint | 1 | 非空 |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 用户角色关联表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 用户角色关联表 | | t\_user\_role | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 用户角色ID | User\_role\_id | int(32) | n/a | 主键 |
| 用户ID | user\_id | Int(32 | n/a | 外键 |
| 角色ID | role\_id | Int(32 | n/a | 外键 |
| 创建时间 | create\_at | datetime | n/a | 当前日期 |
| 修改时间 | update\_at | datetime | n/a | 当前日期 |

## 角色功能关联表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 角色功能关联表 | | t\_role\_function | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 角色功能ID | Role\_function\_id | int(32) | n/a | 主键 |
| 功能ID | function\_id | Int(32 | n/a | 外键 |
| 角色ID | role\_id | Int(32 | n/a | 外键 |
| 创建时间 | create\_at | datetime | n/a | 当前日期 |
| 修改时间 | update\_at | datetime | n/a | 当前日期 |

## 菜品分类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 菜品分类表 | | t\_dish\_category | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 菜品类别ID | category\_id | Int(32) | n/a | 主键，自动增加 |
| 类别名称 | category\_name | Varchar(r128) | n/a | 非空 |
| 类别图片 | category\_picture | Varchar(r128) | n/a | 非空 |
| 创建时间 | create\_at | datetime | n/a | 当前日期 |
| 修改时间 | update\_at | datetime | n/a | 当前日期 |

## 菜品表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 菜品表 | | t\_dish | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 菜品ID | dish\_id | Int(32) | n/a | 主键，自动增加 |
| 菜品分类ID | category\_id | Int(32) | 0 | 外键 |
| 菜品名称 | dish\_name | Varchar(128) | n/a | 非空 |
| 菜品图片 | dish\_picture | Varchar(128) | n/a | 非空 |
| 价格 | price | double(64) | 0.0 | 精确到小数点后一位 |
| 是否在售 | on\_sell | tinyint | True | 非空 |
| 其他信息 | misc | Varchar(128) | n/a | n/a |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 订单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 订单表 | | t\_order | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 订单ID | order\_id | Int(32) | n/a | 主键，自动增加 |
| 订单流水号 | order\_seq | Varchar(128) | n/a | 格式为<yyyymmddxxxxx>  yyyy四位数年份  mm两位数月份  dd两位日期  xxxxx当天订单序号 |
| 订单状态 | order\_status | Varchar(128) | ‘new’ | 非空 |
| 台号 | table\_number | Int(32) | n/a |  |
| 人数 | attendee\_number | Int(32) | 1 |  |
| 实收金额 | total\_price | double | 0.0 | 精确小数点后一位 |
| 服务员ID | servent\_id | Int(32) | n/a | 非空 |
| 会员ID | member\_id | Int(32) | n/a | n/a |
| 收银员ID | casher\_id | Int(32) | n/a | n/a |
| 创建时间 | create\_at | datetime |  | 当前日期 |
| 修改时间 | update\_at | datetime |  | 当前日期 |

## 订单条目表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 订单条目表 | | t\_order\_item | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 订单条目ID | item\_id | Int(32) | n/a | 主键，自动增加 |
| 订单ID | order\_id | Int(32) | 0 |  |
| 菜品ID | dish\_id | Int(32) | 0 | 非空 |
| 份数 | dish\_account | Int(32) | 1 | 最大99 |
| 创建时间 | create\_at | datetime | 当前日期 | 当前日期 |
| 修改时间 | update\_at | datetime | 当前日期 | 当前日期 |

## 会员等级表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据表名 | 会员等级表 | | t\_user\_level | |
| 数据项描述 | 数据项名 | 类型 | 默认值 | 限制 |
| 会员等级ID | level\_id | Int(32) | n/a | 主键，自动增加 |
| 等级名称 | level\_name | Int(32) | n/a |  |
| 等级分数 | level\_score | Int(32) | n/a |  |
| 等级状态 | level\_status | tinyint | n/a | 非空 |
| 创建时间 | create\_at | datetime | 当前日期 | 非空 |
| 修改时间 | update\_at | datetime | 当前日期 | 非空 |

# 接口设计



## 内部接口

## 外部接口

### 接口说明

### 调用方式

# 系统认证安全设计

## 访问控制设计

## 访问控制流程图

## 访问控制流程说明

## 系统性能设计

## 概念

## EnCache配置

# 系统出错处理

## 出错信息输出

## 出错处理对策

# 附录