# 设计

## 1.数据库设计

### 1.MySql

#### 1.用户表user

id: int unsigned，主键

username: varchar(10)，用户名

password:char(32)，密码，使用md5加密

addresses:json，地址的json数组，[{name, phone, address}]

is\_seller:boolean，用户是否有商家身份

is\_delivery\_man:boolean，用户是否有骑手身份

avatar\_filename:varchar(37)，用户头像的文件名称

create\_time:datatime，用户创建的时间

## 2.店铺表restaurant

id:int unsigned，主键

name:varchar(15)，店名

introduction:varchar(200)，简介

sale\_num:int unsigned，销售量

rate:int unsigned，总评分

rate\_count:int unsigned，总评价数

delivery\_price:int unsigned，配送费

categories:json，菜品分类

[

{

分类名，

菜单:[

菜品名，

价格，

描述，

图片文件名

]

}

]

image\_filename:varchar(37)，头像文件名

create\_time:datetime，店铺创建时间

## 3.用户店铺关系表

id:int unsigned，主键

user\_id:int unsigned，用户id

restaurant\_id:int unsigned，店铺id

## 4.店铺评价表restaurant\_comment

id:int unsigned，主键

user\_id:int unsigned，用户id

restaurant\_id:int unsigned，店铺id

content:varchar(200)，评价内容

rate:tinyint unsigned，评价分，1到5

images:json, 附带图片文件名数组，[xxx.jpg, yyy.png]

order\_id:char(32)，评价对应的订单号

create\_time:datetime，创建时间

## 5.订单表`order`

id:char(32)，主键，32位订单号

user\_id:int unsigned，用户id

restaurant\_id:int unsigned，店铺id

delivery\_man\_id:int unsigned，骑手id

menus:json，订单的菜信息，[{id, num, price}]

pack\_price:smallint unsigned，打包费

delivery\_price:smallint unsigned，配送费

price:int unsigned，总价

address:varchar(100)，配送地址

### 2.redis

#### 1.token

key: token

value: user\_ id

用户的token和id，保存时间2天

key: refreshToken

value: user\_id

用户的refreshToken和id，保存时间4天

#### 2.order

key: order\_id

value: {id, userId, restaurantId, deliveryManId, menus: [{

id, name, num, price

}], packPrice, deliveryPrice, price, address, createTime, completeTime}

用户提交的订单，等待商家接单，10分钟

## 2.功能设计

说明：

除了登录注册都需要提供Authorization的Bearer认证头通过token认证用户，失败则返回401。

后端根据请求地址的第一个来确定使用的controller。

除了token认证，所有响应都包装成200响应码，成功的响应的code字段为0， message字段为'操作成功'。

## 用户 /user

#### 1.用户登录

请求地址：post /login

请求数据：json

{phone, password}

说明：使用userservice根据phone获取用户，密码加密比对，成功就redis写入token并返回tokens，否则根据不同错误返回。

响应：json

成功；{code , data: {token, refreshToken}}

失败：{code: 1~3, message: "错误信息"}

#### 2.用户注册

请求地址：post /register

请求数据：json

{name, phone, captcha, password}

说明：使用userservice注册，成功就redis写入token并返回，否则说明已经存在手机号，返回错误

响应：json

成功：{code, data: {token, refreshToken}}

失败：{code: 1, message:"手机号已注册"}

#### 3.获取自己信息

请求地址：get /info

请求数据：无

说明：使用userservice根据用户id找到并返回，不会错误。

响应：json

成功：{code, data: User}

失败：无

#### 4.刷新token

请求地址：get /refresh

请求数据：无

说明：根据refreshToken使用userservice删除原有token并创建新token返回。

响应：json

成功：{code, data: {token, refreshToken}}

失败：无

#### 5.修改身份

请求地址：put /role/{isSeller}/{set}

请求数据：isSeller表示目标是商家还是骑手，set表示注册还是注销

说明：使用userservice修改用户目标信息。

响应：json

成功：{code, data: User}

失败：无

#### 6.更改头像

请求地址：put /upload/avatar

请求数据：表单的单个图片文件avatar

说明：使用userservice把上传文件存到阿里云，如果文件错误则失败。

响应：json

成功：{code, data: 文件名}

失败：{code: 1, message: 参数错误}

#### 7.上传评论

请求地址：put /comment/{orderId}

请求数据：orderId表示根据哪个订单评论，并json传来RestaurantComment实体。

说明：使用commentservice根据订单信息判断参数是否合理，正确则加入评论并返回评论，否则失败。

响应：json

成功：{code, data: RestaurantComment}

失败：{code: 1, message: 参数错误}

#### 8.获取历史订单

请求地址：get /orders/{pageOffset}/{pageSize}

请求数据：根据路径参数确定分页

说明：使用orderservice根据分页按照时间降序查找订单返回。

响应：json

成功：{code, data: [Order]}

失败：无

#### 9.请求订单

请求地址：put /order/put

请求数据：json的订单信息

说明：使用orderservice检查订单并添加，如果校验失败则返回错误信息。

响应：json

成功：{code, message}

失败：{code: 1, message: 下单失败，请重试}

#### 10.取消订单

请求地址：delete /order/cancel/{orderId}

请求数据：无

说明：使用orderservice取消订单

响应：json

成功：{code, message}

失败：{code: 1, message: 取消失败，请刷新重试}

#### 11.修改用户名

请求地址：put /update/username

请求数据：{username}

说明：使用userservice校验用户名，成功就保存

响应：json

成功：{code, message}

失败：{code: 1, message: 更新失败，请检查是否符合校验}

## 店铺 /restaurant

#### 1.通过id获取店铺信息

请求地址：get /info/{restaurantId}

请求数据：restaurantId表示店铺的id

说明：使用restaurantservice根据id获取店铺信息，可能不存在

响应：json

成功：{code, data: Restaurant}

失败：{code: 1, message: 目标不存在}

#### 2.获取多个店铺信息

请求地址：get /info/{pageOffset}/{pageSize}

请求数据：pageOffset和pageSize分页查询

说明：使用restaurantservice分页查询，没有就返回失败信息

响应：json

成功：{code, data: [Restaurant]}

失败：{code: 1, message: 未查询到结果}

#### 3.注册店铺

请求地址：post /register

请求数据：json {restaurantName, description}

说明：使用restaurantservice验证并创建店铺，失败返回错误信息。

响应：json

成功：{code, data: Restaurant}

失败：{code: 1, message: 创建失败，请换名重试}

#### 4.添加菜品分类

请求地址：patch /category/add/{name}

请求数据：name为分类名

说明：使用restaurantservice添加新的分类，总分类名长度加分类数-1的字符不能过100

响应：json

成功：{code, message}

失败：{code, message: 请检查分类添加规则}

#### 5.删除菜品分类

请求地址：patch/category/delete/{index}

请求数据：index指分类的索引

说明：使用restaurantservice删除指定索引的分类并删除菜品数据，索引越界出错

响应：json

成功：{code, data: Restaurant}

失败：{code: 1, message: 删除失败，请检查参数}

#### 6.修改分类名称

请求地址：patch /category/update/{index}/{name}

请求数据：index指分类索引，name指新名字

说明：使用restaurantservice把指定分类索引的名字改为新名字，越界或不符合分类规则出错。

响应：json

成功：{code, data: Restaurant}

失败：{code, message: 请检查分类规则}

#### 7.修改店铺图片

请求地址：put /upload/image

请求数据：表单图片文件image

说明：使用restaurantservice修改店铺的图片，文件不符合规则出错

响应：json

成功：{code, data: 文件名}

失败：{code: 1, message: 失败，请检查文件格式}

#### 8.接单

请求地址：put /order/take/{orderId}

请求数据：orderId指目标订单号

说明：使用orderservice从redis获取订单校验成功就接单写入mysql，否则报错

响应：json

成功：{code, data: Order}

失败：{code: 1, message: 接单失败}

#### 9.添加菜品

请求地址：patch /menu/add/{index}

请求数据：json Menu，index是分类的索引

说明：商家接口，使用menuservice添加新的菜

响应：json

成功：{code, message}

失败：{code, message: 请检查菜品规则}

#### 10.修改菜品

请求地址：patch /menu/update/{categoryIndex}/{menuIndex}

请求数据：json Menu，两个路径变量是分类索引和菜单索引

说明：商家接口，使用menuservice修改菜品

响应：json

成功：{code,message}

失败：{code, message: 请检查菜品规则}

#### 11.删除菜品

请求地址：patch /menu/delete/{categoryIndex}/{menuIndex}

请求数据：两个路径变量是分类索引和菜单索引

说明：商家接口，使用menuservice删除菜品

响应：json

成功：{code, message}

失败：无

#### 12.修改菜品图像

请求地址：put /menu/upload/image/{categoryIndex}/{menuIndex}

请求数据：两个路径变量是分类索引和菜单索引，额外传表单图片文件

说明：商家接口，使用menuservice上传图片

响应：json

成功：{code, data: 文件名}

失败：{code, message: 请检查文件格式}