平哥de店开发手册

ver1.0

|  |  |  |  |
| --- | --- | --- | --- |
| **最后版本修改日期** | 2020.12.31 | **审核人** | 平哥 |
| **版本参与修订者** | 平哥、王二、张三、李四 | | |

一、环境搭建要求：

1. 生产环境：

服务器系统要求采用32核CPU，服务器内存不能小于32G， 系统为centos7， python3.6～python3.7版本，mysql为5.7版本

2. 开发环境

开发人员首先搭建自己的开发的开发环境，然后提交到开发版本库， 在内部测试环境下进行开发测试

3. 内测环境

内测环境与生产环境保持一致，并使用线上数据脱敏测试，只有通过内测环境测试通过的版本，才能提交到生产环境中

二、开发流程

需求制定 - 需求审核 - 产品设计 - 原型审核 - 开发任务排期 - 开发 - 测试 - 验收 - 上线 - 维护

开发人员应当按照需求文档设计要求进行开发

三、账号管理

开发人员不能分配root级用户，如果需要root级操作，应当进行申请

内测环境数据库的root用户不能用于程序测试，必须给程序分配合适权限的账号

开发人员离职前需要进行账号销毁确认

四. 商城简介

平哥de店是目前知名的土特产直销平台，拥有会员上亿，每日日活量超千万，日访问数十亿

目前，平哥de店具有会员管理，商品管理，订单管理等模块，将来开发分级营销、数据分析等功能

平哥de店基于flask-mall系统进行开发，该系统具有功能完善，开源、扩展强等优势，目前运行状况良好

**flask-mall**系统具有以下特性：

1. 采用MVC模式

通过该模式实现了前后端分离，前端开发人员可以轻松的使用jinja模版语法开发商城界面，

2. API接口

该系统还为移动app，小程序提供了API接口，实现一个后端，匹配多个前端

3. 性能优化

该系统支持使用redis、mongodb作为高速缓存机制，并且支持读写数据分离，既可以实现单件部署，也可以实现容器化部署，从而支持高并发

4. 友善的开发模式

该系统内置日志处理，一旦开启调试模式，开发人员可以通过日志分析系统运行状态

5. 良好的模块化与扩展性

该系统采用的基础框架为弹性极高的flask框架，采用模块化设计，通过提供开源代码以及相应的flask-mall扩展，用户可以迅速的架设一个商城，或者添加新的功能，通过flask-mall提供的云计算服务，还可以进行数据挖掘

6.flask-mall源码获取

git clone git@github.com:luxp4588/flask-mall-base.git

7. flask-mall系统文件结构

**/- static** 静态文件夹

**/- uploads** 上传文件路径

**/- logs** 日志文件夹

**/- alipay\_cert** 支付宝支付接口密钥文件

**/- wechat\_cert** 微信支付接口密钥文件

**/- templates** 模版目录

**/- xp\_mall** 系统内核模块

**app.py** application程序

**wsgi.py** wsgi程序接口

五、xp\_mall核心模块文件结构

/- xp\_mall

/- admin 后台管理功能

/- \_\_init\_\_.py

/- category.py 分类管理功能

/- goods.py 商品管理功能

/- orders.py 订单管理功能

/- uploads.py 图片上传

/- forms 表单文件夹，提供各类表单定义

/- mall 商城前台模块（首页，商品浏览，详情， 订单支付回调接口）

/- \_\_init\_\_.py

/- goods.py 首页、商品列表、详情等

/- order.py 第三方支付完毕回调接口

/- member 会员中心模块

/- auth.py 账号管理，密码修改，取回

/- buy.py 用户订单创建与支付

/- cart.py 购物车

/- center.py 会员中心（购物车管理、订单管理，资料管理）

/- models 模型 定义商城数据模型

/- extensions app扩展

/- filters 自定义过滤器

/- login\_manage.py 用户注册与登陆

/- settings.py 系统配置文件

/- utils.py 工具文件，提供缩略图，分类数据等帮助性函数

六、模型类库

建议使用事务型数据库，比如mysql的innodb数据存储引擎

1. 会员模型 - member

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| user\_id | int |  | 用户id | 自增长 |
| username | varchar | 30 | 用户名 | 唯一 |
| password\_hash | varchar | 128 | 密码 |  |
| email | varchar | 50 | 邮箱 |  |
| sex | varchar | 2 | 性别 |  |
| mobile | varchar | 12 | 手机 |  |
| reg\_date | datetime |  | 注册日期 | 不可修改 |
| reg\_ip | varchar | 20 | 注册ip | 不可修改 |
| last\_login\_ip | varchar | 20 | 上次登陆ip |  |
| last\_login\_time | datetime |  | 上次登陆时间 |  |
| is\_approve | BOOLEAN | 1 | 是否认证 |  |
| is\_admin | BOOLEAN | 1 | 是否管理员 |  |

1. 商品模型 - goods

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| goods\_id | Int |  | 商品id | 自增长 |
| goods\_title | varchar | 100 | 商品名称 |  |
| goods\_subhead | varchar | 100 | 商品副标题 |  |
| category\_id | int |  | 分类id | 外键，指向goods\_category.id |
| thumb | varchar | 100 | 缩略图路经名称 |  |
| main\_pic | varchar | 500 | 主图路经名称 |  |
| price | decimal |  | 价格 |  |
| detail | text |  | 商品详情 |  |
| create\_time | datetime |  | 商品信息创建时间 |  |

1. 购物车模型 – cart

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| id | int |  | 购物车id | 自增长 |
| goods\_id | int |  | 商品id | 外键，指向goods.goods\_id |
| amount | int |  | 商品数量 | 默认为0 |
| user\_id | int |  | 用户id | 外键，指向  member.user\_id |

4. 分类模型 - goods\_category

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| id | int |  | 商品分类id | 自增长 |
| name | varchar | 30 | 商品分类名称 |  |
| order\_id | int |  | 商品分类顺序 |  |
| parent\_id | int |  | 商品分类的父类 | 自关联，指向自身的id |

5. 物流数据模型 - logistics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| id | int |  | 订单物流id | 自增长 |
| order\_id | int |  | 订单id | 外键，指向order.id |
| receifver | varchar | 20 | 收件人姓名 |  |
| mobile | varchar | 30 | 收件人手机 |  |
| address | varchar | 100 | 收件地址 |  |
| status | varchar | 100 | 物流状态 |  |

6. 订单模型 - order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| id | int |  | 订单id | 自增长 |
| order\_no | varchar | 50 | 订单编号 |  |
| subject | varchar | 100 | 订单说明 |  |
| total\_price | decimal |  | 订单总价格 |  |
| status | varchar | 10 | 订单状态 | 等待支付，等待发货，等待收货，已收货 |
| seller | varchar | 50 | 卖家账号 |  |
| buyer | varchar | 50 | 买家账号 |  |
| createTime | datatime |  | 订单创建时间 |  |
| payment | varchar | 30 | 支付方式 |  |
| paytime | datetime |  | 订单付款时间 |  |

7. 订单商品详细 - order\_goods

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| id | int |  | 商品分类id | 自增长 |
| goods\_id | int |  | 商品id |  |
| order\_id | int |  | 订单id | 外键，指向order.id |
| price | decimal |  | 商品价格 |  |
| order\_price | decimal |  | 订单价格 |  |
| order\_active | varchar | 255 | 订单是否激活 | 默认为None |
| amount | int |  | 商品数量 |  |
| discount | float |  | 订单折扣 |  |

7. 地区模型 – area

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 字段名注释 | 备注 |
| place\_id | int |  | 地区id | 自增长 |
| place\_name | varchar | 50 | 地区名称 |  |
| place\_order | int |  | 地区顺序 |  |
| parent\_id | int |  | 地区的父类 | 自关联，指向自身的id |

七、各模块视图函数接口

1.AuthManage模块

1.1 登陆接口

url : <http://domain/login>

请求方式： GET， POST

接口说明：GET方式返回登陆界面

POST方式接受登陆数据完成登陆验证

函数名：**login**

POST参数：

| 参数名 | 约束 | 说明 |
| --- | --- | --- |
| username | 字母与数字组成 | 用户名 |
| password |  | 密码 |
| remember | 布尔值 | 是否一周内免登陆 |
|  |  |  |

返回：

GET ： 登陆页面

POST： 登陆成功跳转，失败返回登陆页面

1.2 注册接口

url： <http://domain/register>

请求方式：GET， POST

接口说明：GET 返回注册页面

函数名：**register**

**post**参数：

| 参数名 | 约束 | 说明 |
| --- | --- | --- |
| username | 字母开头，字母与数字组成，长度不能超过30 | 用户名 |
| password | 密码长度6～15 | 密码 |
| email | email格式 | 需要通过邮箱验证 |
| sex | 枚举：男，女 |  |
| mobile | 手机格式 | 需要通过手机验证码验证 |

2.Member模块

2.1 auth.py

2.1.1 用户退出

2.2. cart.py

2.2.1 加入购物车

url : <http://domain/member/cart/add/goods_id/> （goods\_id为商品id）

请求方式：post

post参数：无

返回数据：json格式

添加成功 {“result”: False}

添加失败:{“result”:True}

2.2.2 删除购物车

url: <http://domain/member/cart/cart_id> (cart\_id为待删除购物车id)

请求方式: delete

返回数据：json格式

添加成功 {“result”: False}

添加失败:{“result”:True}

2.2.3 购物车商品数修改

url：<http://domain/member/cart/cart_id> (cart\_id为修改购物车id)

请求方式：put

put数据：amount - 整型>0, 商品数量

返回数据：json格式

修改成功 {“result”: False}

修改失败:{“result”:True}

2.2.4 获取购物车数据

url: <http://domain/member/cart/>

请求方式: get

返回数据：json格式

{“id”:””, “goods\_id”:””,”amount”:””, “user\_id”:”” }

2.3 buy.py