数据库课程大作业实验报告

一、 实现背景

随着互联网的全民普及，网络平台已经成为普通百姓出门在外预订不熟悉的酒店时的主要途径。然而，现在网上的酒店预订平台众多，酒店房间信息纷繁。对于庞大的消费者群体而言，如何综合考量酒店的 舒适程度、交通的便捷与否、房间价格的高低等因素并结合自己的实际需求快速地选择预订房间资源就 成为一个重要的问题。

现有的网络平台如携程、去哪儿网、马蜂窝大多作为处于酒店商家和消费者之间的唯一中介而存在，因而大多是面向酒店考虑的。因此便存在不够面向用户、贴近用户的问题。具体而言：

缺少类似电子商务中介于用户和淘宝京东之间的"慢慢买网"这样的比价平台，无法对不同平台上地 理位置类似的酒店进行其他方面的比较考量，也无法在时间上进行纵向的比较。

不同于一般的商品，酒店的预订和其在城市中的地理位置高度相关，而现有的网络预订平台大多仅 仅对地理位置基于一个文本性的描述。消费者在这些平台上几乎无法直观地看到一个城市的酒店的 价格随地段的分布情况。交通出行的便利程度等等。 本产品将尽力改进上述不足之处并发扬自己的优势，当然，作为一门课程的大作业，我们无法在有限的 时间内实现一个真正完善实用的酒店信息比较与预订系统。但我们将会完成其核心功能并为未来可能的开发留好相应的接口。

二、 实现环境

a) 前端

   Vue

b) 后端

   Django

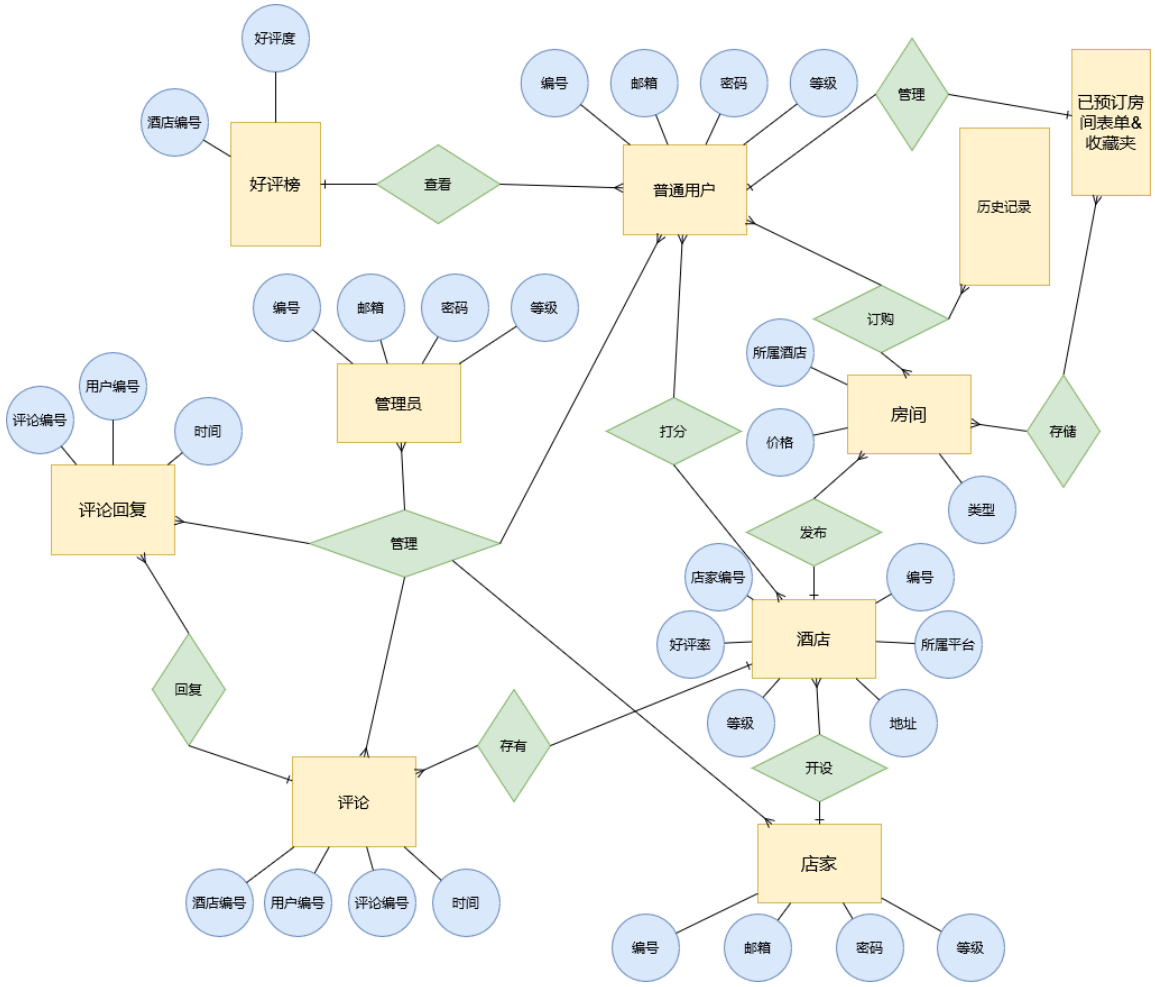
c) 爬虫其他

   Selenium、Pyquery、Pandas

d) 数据库

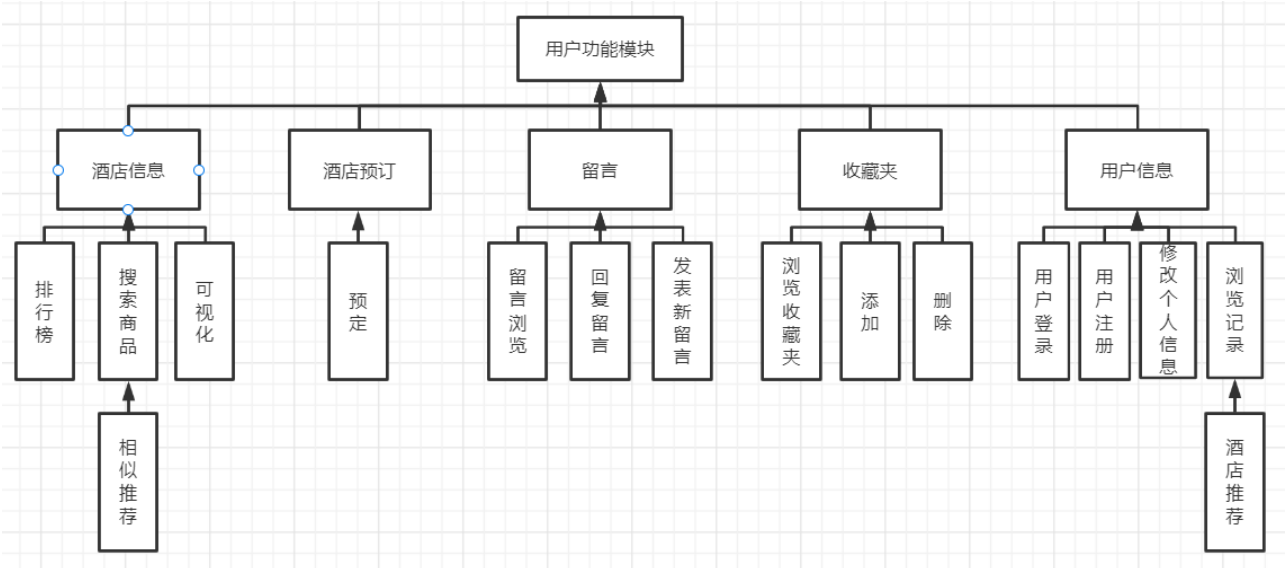
   MySQL、Navicat

三、 概念模式（E-R 图）

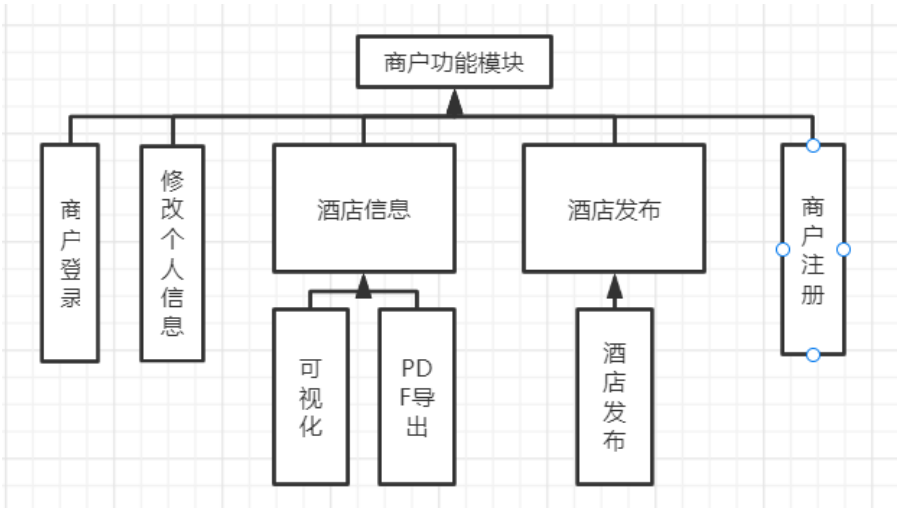


四、 系统功能结构图（画图）

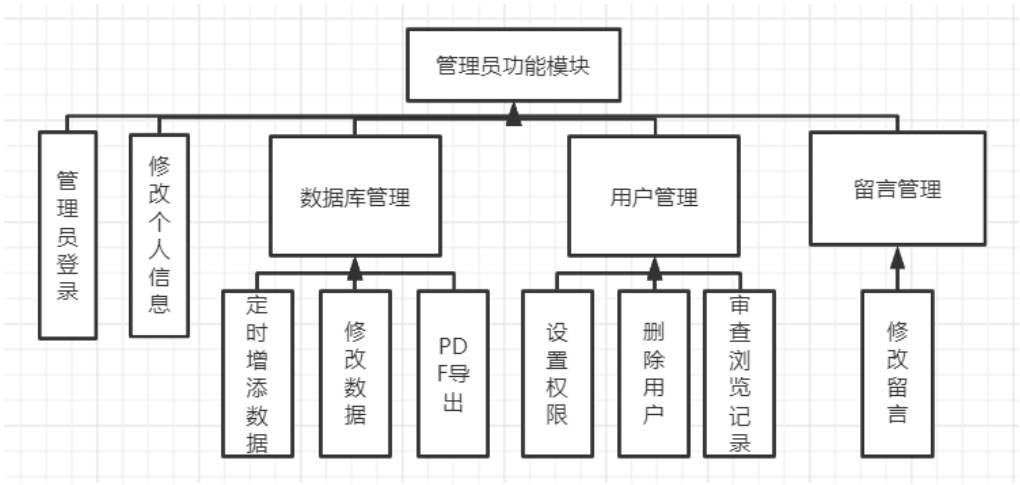
a) 用户功能模块



b) 商家功能模块



c) 管理员功能模块



五、 数据库基本表及约束（范式、约束）

（1） 用户数据信息 (User)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	用户编号	int	4	无	主键	否
name	用户名	varchar	32	无特殊字符	无	否
email	用户邮箱	varchar	32	Unique	无	否
password	密码	varchar	32	无	无	否
date	注册时间	date	无	无	无	否

（2） 商家数据信息(Business)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	商家编号	int	4	无	主键	否
name	商家名	varchar	32	无特殊字符	无	否
email	商家邮箱	varchar	32	Unique	无	否
password	密码	varchar	32	无	无	否
date	注册时间	date	无	无	无	否

（3） 平台数据信息(Platform)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	平台编号	int	32	无	主键	否
name	平台名	varchar	32	无特殊字符	无	否

（4） 酒店数据信息(Hotel)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	酒店编号	int	4	无	主键	否
name	酒店名	varchar	32	无特殊字符	无	否
platform_id	平台编号	int	4	无	外键引用 Platform(id)	否
business_id	商家编号	int	32	无	外键引用 Business(id)	否
picture	酒店图片	varchar	256	无	无	是
type	酒店类型	varchar	32	无	无	否
price	基础价格	int	16	无	无	否
address	酒店地址	varchar	128	无	无	否
longitude	酒店经度	double	(16,8)	无	无	是
latitude	酒店纬度	double	(16,8)	无	无	是
score_num	评分	double	(4,2)	无	无	是
score	评分人数	int	8	无	无	否

sales	总销量	int	16	无	无	否
popular	总人气值	int	16	无	无	否

(5) 酒店每日数据记录(HotelDetail)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	酒店编号	int	4	无	主键(id, date)	否
date	日期	date	无	无	主键(id, date)	否
price	当日价格	int	8	无	无	否
sales	当日销量	int	16	无	无	否
score	当日评分	double	(4,2)	无	无	否

(6) 房间信息数据（Room)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	房间编号	int	4	无	主键	否
hotel_id	酒店编号	int	32	无	外键 Hotel(id)	否
type	类型	varchar	16	无	无	否

(7) 房间每日数据记录（RoomDetail)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	房间编号	int	4	无	主键(id, date)	否
date	日期	date	无	无	主键(id, date)	否
price	当日价格	int	8	无	无	否
count	当日剩余数量	int	16	无	无	否

(8) 评论信息数据 (Comment)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	评论编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否
User_id	用户编号	int	4	无	外键 User(id)	否
date	评论时间	date	无	无	无	否
context	评论内容	varchar	999	无	无	否

（9） 评论回复数据 (Reply)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	回复编号	int	4	无	主键	否
comment_id	评论编号	int	4	无	外键 Comment(id)	否
User_id	用户编号	int	4	无	外键 User(id)	否
date	回复时间	date	无	无	无	否
context	回复内容	varchar	999	无	无	否

（10） 历史记录数据 (History)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	历史记录编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否
User_id	用户编号	int	4	无	外键 User(id)	否
date	浏览时间	date	无	无	无	否

（11） 收藏夹数据 (Collections)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	收藏编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否
User_id	用户编号	int	4	无	外键 User(id)	否
date	收藏时间	date	无	无	无	否

（12） 预订记录数据 (Booking)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	预订编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否
User_id	用户编号	int	4	无	外键 User(id)	否
date	预订时间	date	无	无	无	否

（13） 低价榜 (PriceRank)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值

id	排行榜编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否

（14）评分榜（ScoreRank）

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	排行榜编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否

（15）销量榜（SalesRank)

数据项名	说明	数据类型	数据长度	值约束	键约束	允许空值
id	排行榜编号	int	4	无	主键	否
hotel_id	酒店编号	int	4	无	外键 Hotel(id)	否

六、 SQL 代码说明（代码说明）

a) 存储指令

存储过程 checkadmin 在管理员前端界面登录中被调用。前端接收到用户名和密码的信息通过后端发送给数据库，在 Admin 表中进行查询，若存在对应用户，则返回用户的 enabled 属性（为 0 或 1），即代表了管理员权限，后面会用来对应前端不同视图。若无匹配用户，则返回 -1，以便后端进行相应错误处理操作。

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `checkadmin`(IN adname VARCHAR(10),IN adpwd VARCHAR(32))
Begin
    if exists (select * from admins where ad_name = adname and ad_pwd = adpwd)
    then select enabled from admins where ad_name = adname and ad_pwd = adpwd;

    ELSE SELECT -1;

    end if;
End
```

b) 触发器

实现了人气榜、价格榜、评分榜的自动更新功能，使用了触发器。以人气榜举例，首先从 collection 表中选出收藏人数最多的前五个商品作为人气榜中的商品。 具体实现方式为，在每一个用户对一个商品点击收藏操作之后，则将 popular（人气榜）

表中的数据清空，并选取 collection 中收藏人数最多的前五个商品插入到 popular 表中，并将人气值设置为收藏人数 * 100。 相关语句为:

```
create trigger updatePopularAuto_hotel
after insert on collection for each row
begin
    delete  from popular_hotel;
    insert into popular (select comid, count(comid) * 100 from collection group by comid LIMIT 5);
end
```

c) 函数

为实现管理员对某个酒店删除时需要删除有关该酒店的所有相关信息时(即数据库中表的某一个属性为酒店的 id)时，需要相关函数，如下：

```
create function deleteHotel(comid int) returns int
begin
    delete  from Hotel where Hotel.id = comid;
    delete  from price where price.comid = comid;
    delete  from collection where collection.comid = comid;
    delete  from history where history.comid = comid;
    delete  from replymessage where replymessage.leavemessageid in (select id from leavemessage where
leavemessage.comid = comid);
    delete from leavemessage where leavemessage.comid = comid;
    delete from popular where popular.id = comid;
    return 0;
end
```

七、 主要技术介绍

a) 实现概述

项目主要采用 Vue.js + django 框架进行前后端分离式 Web 开发。前端分为 管理员界面、商户界面和 用户界面 ，通过 API 向后端请求或发送数据进行交互，后端再通操作数据库传递数据。

本项目中设计了爬虫，每日自动爬取驴妈妈、途牛、艺龙等数据信息导入数据库，实现动态更新商品信息。并且使用高德地图的 api 完成了对于每个酒店的经纬度信息的精确获取，并且实现将钻取到的数据呈现在地图上以便直观显示。

b) 数据获取论述

本书项目主要设计了对于驴妈妈、途牛、艺龙三个网站的爬虫，每日定时获取最新酒店信息。通过 selenium、pandas、pyquery 和 chromedriver 等成熟的 python 库完成对于数据的爬取。之后利用高德地图提供的经纬度转换以及地图可视化等 api、完成对于

酒店经纬度、酒店可视化地图显示等功能的实现。再利用正则表达式等基本操作进一步规范数据。

c) 前端技术论述

前端采用 Vue.js 进行开发。HTML 组件采用 ElementUI 模板，图表的展示采用 VCharts 模板。设计独立的$store 模块进行数据缓存(用户信息、酒店信息等)，$axios 模块集成网络 API 与后端进行数据的交互。

d) 后端技术论述

后端我们主要是使用了 Django 来完成数据的获取、分析、传输。Django 是一个开放源代码的 Web 应用框架，由 Python 写成。采用了 MTV 的框架模式，即模型 M，视图 V 和模版 T。它最初是被开发来用于管理劳伦斯出版集团旗下的一些以新闻内容为主的网站的，所以有着几乎现成的管理员模式。

通过解析前端传输的 URL 和 request，解析 json 包，获取前端传输的信息。并根据信息，利用 Django 的 models 以及数据库查询函数，完成对于数据的检索。之后封装成 json 格式传回前端。

```
urlpatterns = [
    path('Login/', login),
    path('hotels/', hotels),
    path('popular/', popular),
    path('getAllCollections/', get_all_collections),
    path('rankAll/', rankAll),
    path('addHistory/', addHistory),
    path('price/', price),
    path('leavemessage/', leavemessage),
    path('replymessage/', replymessage),
    path('user/', user),
    path('getAllHistories/', getAllHistory),
    path('delAllHistories/', delAllHistories),
    path('delAllCollections/', delAllCollections),
    path('updateUser/', updateUser),
    path('updateLocHotel/', updateLocHotel),
    path('addLeaveMessage/', addLeaveMessage),
    path('addReplyMessage/', addReplyMessage),
    path('addScore/', addScore),
    path('addCollection/', addCollection),
    path('upload_logo/', update_logo),
    path('MapUpdate/', map_update),
    path('queryRoomInfo/', queryRoomInfo),
```

e) 数据库技术论述

数据库我们主要使用了 MySQL 数据库，MySQL 是一种关联数据库管理系统，关联数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加

了速度并提高了灵活性。并且利用 Django 的 model 部分完成对于数据库表的映射，通过 migrate 等操作语句，完成建库等一系列基础操作，实现了数据库的自动化建立。在查询、搜索、修改等操作方面，Django 并不需要我们来显式地使用 SQL，为了提高普适性，设计了对应的数据库操作语言，例如 create，delete，filter 等操作，极大地方便了我们对于数据库的操作。

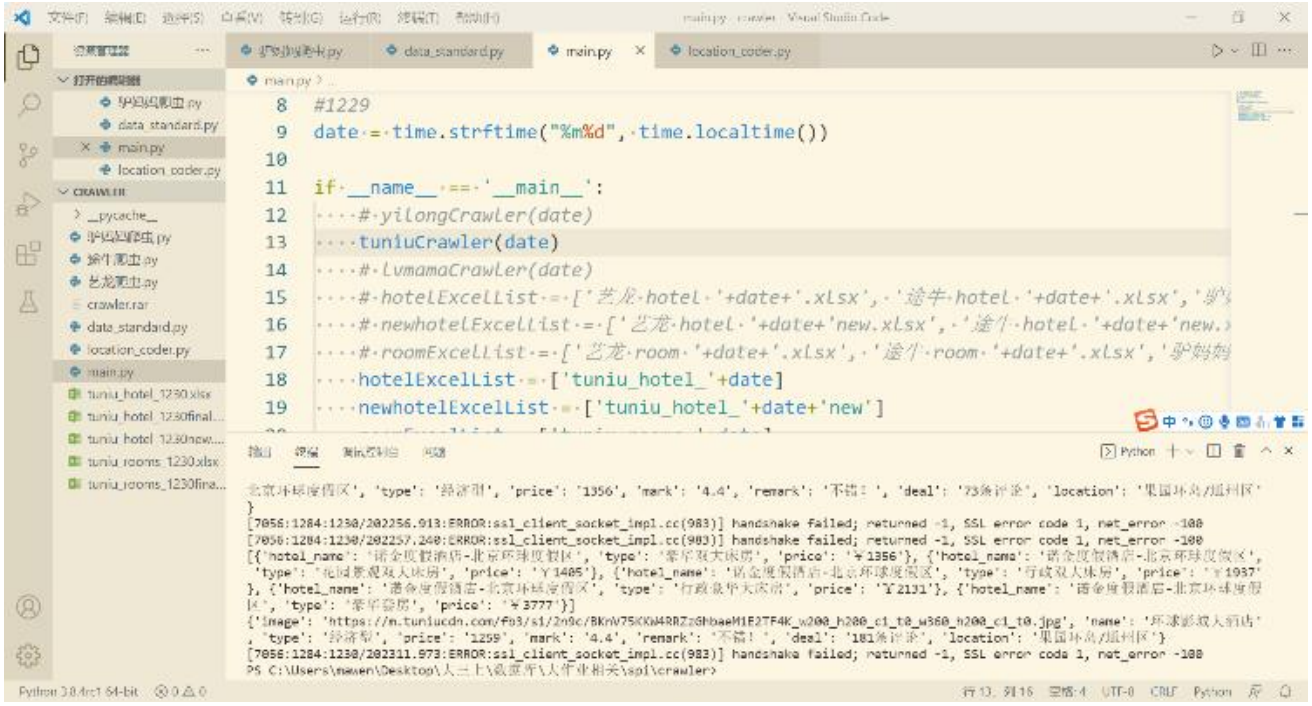
```python
def login(request):
    body_json = request.body.decode()
    body_dict = json.loads(body_json)
    print(body_dict)
    email = body_dict.get('mail')
    password = body_dict.get('password')
    info = None
    code = 0
    if User.objects.filter(email=email).exists():
        user = User.objects.get(email=email)
        if user.password == password:
            code = 200
            msg = "登录成功"
```

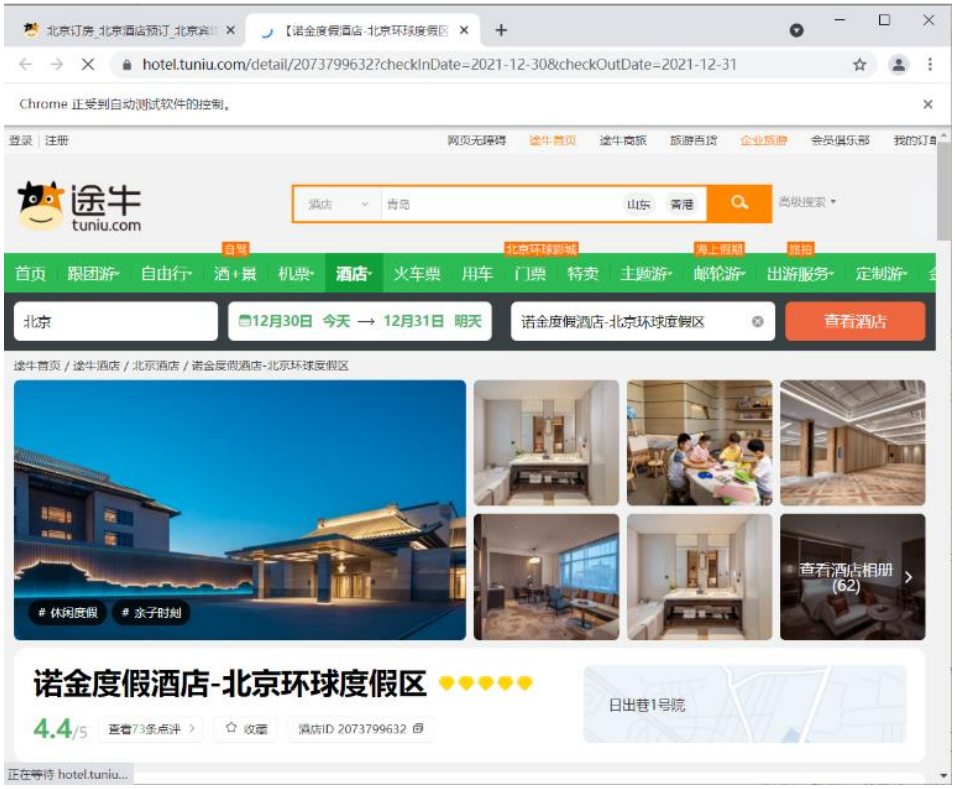
八、 系统功能演示

a) 数据获取

i. 网站爬取相关数据

运行 crawler.py 程序，会自动调用驴妈妈、途牛、艺龙三家酒店的爬虫，完成全自动爬取，存储在 xlsx 文件中；





之后会自动调用 data_standard.py 完成对于爬取信息的标准化，方便直接导入数据库，满足其规范；

```
##price
ori = data.iloc[i,3]
matchObj = re.match( r'.*?(\d+).*', ori, re.M|re.I)
data.iloc[i,3] = matchObj.group(1)
##mark
ori = data.iloc[i,4]
matchObj = re.match( r'.*?(\d+\.\d+).*', ori, re.M|re.I)
data.iloc[i,4] = matchObj.group(1)
#deal
ori = data.iloc[i,6]
matchObj = re.match( r'.*?(\d+).*', ori, re.M|re.I)
data.iloc[i,6] = matchObj.group(1)
```

最后调用 location_coder 函数完成调用高德的经纬度转换 api，实现酒店经纬度的获取。

ii. 经纬度

经纬度功能利用了高德地图的地理编码功能的接口，通过使用申请到的 Key 向特定的 url 发送请求，获取数据。

地理编码

- 地理编码 API 服务地址

URL	https://restapi.amap.com/v3/geocode/geo?parameters
请求方式	GET

地址	经纬度
万明路11	116.388916,39.887960
建国门内	116.428215,39.910156
【古北水	117.264674,40.647706
【古北水	117.276270,40.653689
中关村大	116.322519,39.970387
永丰路与	116.243307,40.073297
北京 东城	116.398218,39.895243
【古北水	117.276270,40.653689
西八里庄	116.290616,39.931922
阜外大街	116.343619,39.922589
农大南路	116.298555,40.024883
沙滩后街	116.401544,39.925486
永安路17	116.385851,39.886615
兴隆庄甲	116.534326,39.910828
知春路25	116.344999,39.977268
北京市西	116.375547,39.898767
【古北水	117.276270,40.653689
祁家豁子	116.381643,39.980434
东升镇马	116.371137,40.039769
南街29号	117.118061,40.154084
金融街金	116.362960,39.915159
白桥大街	116.438901,39.898938
挂甲屯5号	116.299504,39.996273
芍药居16	116.433396,39.983296
白云路西	116.341443,39.905502
大栅栏西	116.392654,39.893854

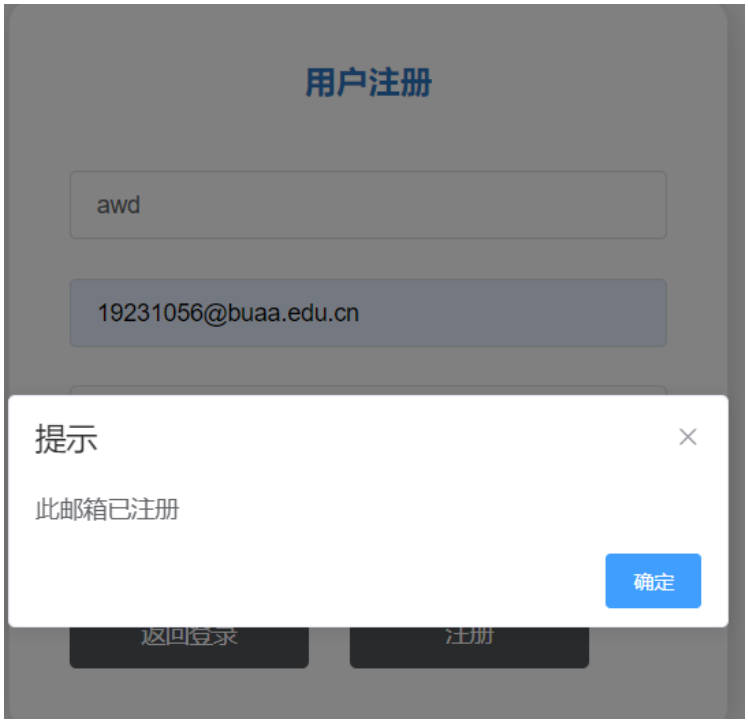
b) 用户模式

(1) 登录功能

用户根据邮箱和密码进行登录，当邮箱未注册或密码错误则不会登录成功，同时会显示错误信息。

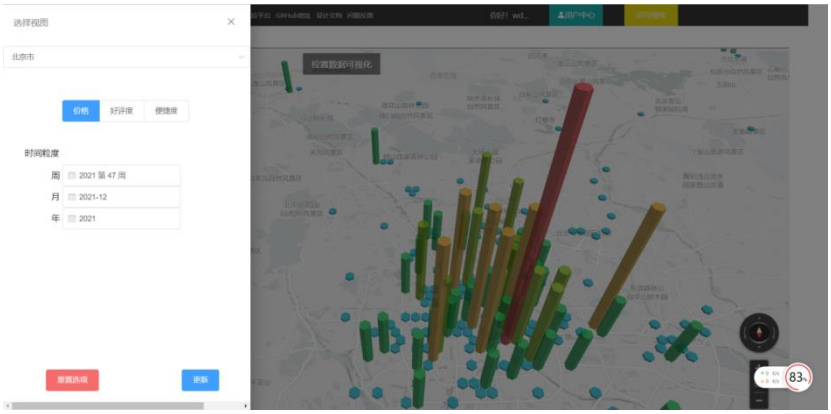
(2) 注册功能

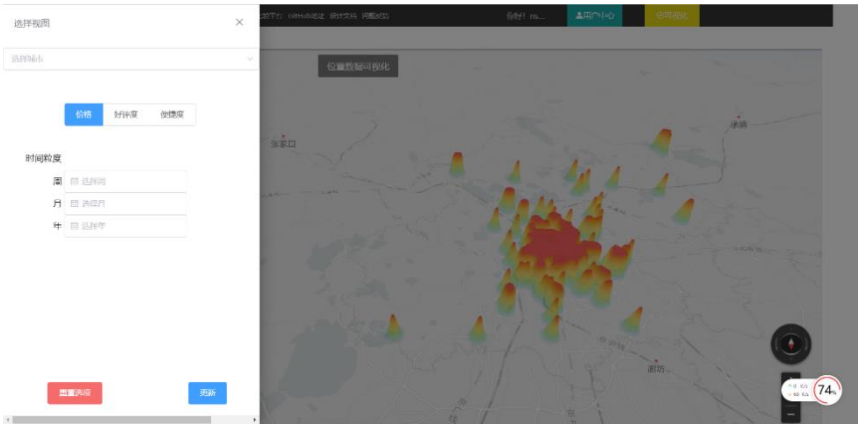
未注册账号的用户可以根据邮箱进行注册，邮箱不能重复，否则会显示错误信息，用户名、邮箱和密码字段都不能为空。用户可以根据实际需要可以选择注册普通用户/商家。普通用户只能获取平台的酒店信息，而商家具有在平台发布酒店信息的功能，后面会详细介绍。



(3) 可视化界面

进入主页面后，点击主页面上方的可视化按钮，即可进入可视化界面。可视化界面与高德地图连接，可实时更新酒店的最新情况。用户可以选择地点和时间粒度，并选择价格、好评度或便捷度的区间进行筛选，可将筛选后的结果进行可视化展示。



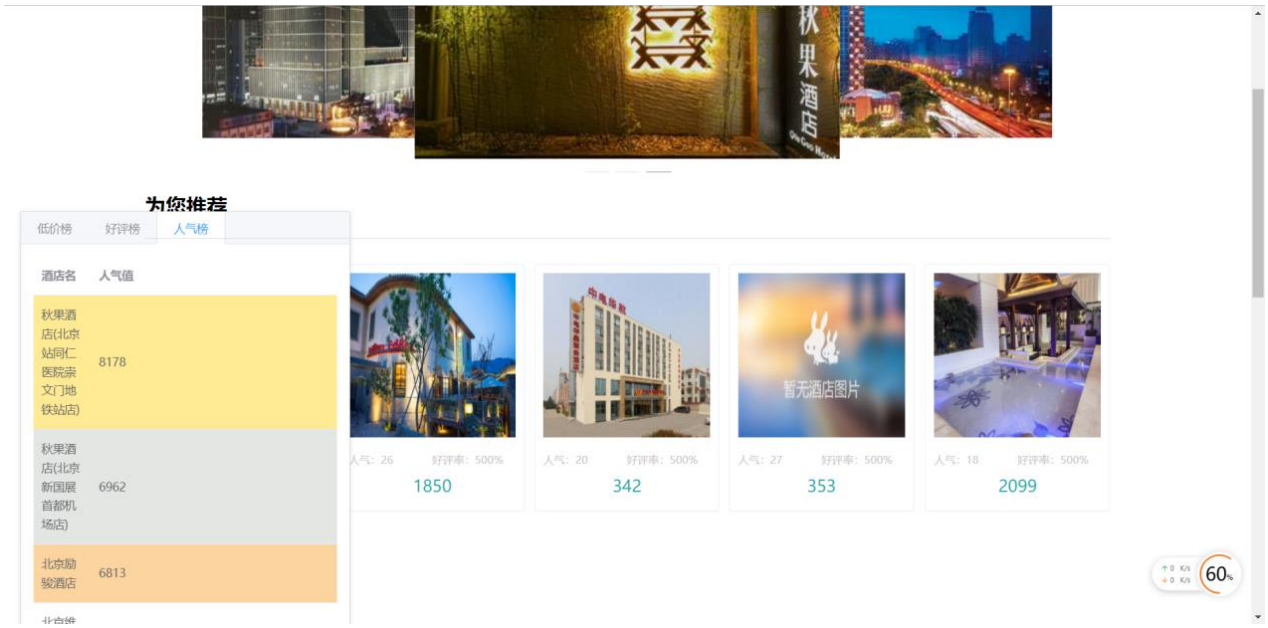


(4) 人气榜、低价榜与评分榜

主页面设置了人气榜、低价榜、评分榜的展示，便于用户选择酒店预订。人气榜根据商品收藏人数进行更新，显示出当前人气值排名前 3 的酒店；低价榜则根据酒店当天价格进行更新，显示出当日价格最低的酒店；评分榜则根据酒店的评分和评分人数进行更新，显示当前评分排名前 3 的酒店。（详细见触发器功能）

(5) 酒店展示

展示所有酒店，包括酒店的图片、酒店名称、酒店评分、最新价格等信息，鼠标点击图片会跳转到新的页面，显示出酒店的完整信息。



(6) 关键词搜索

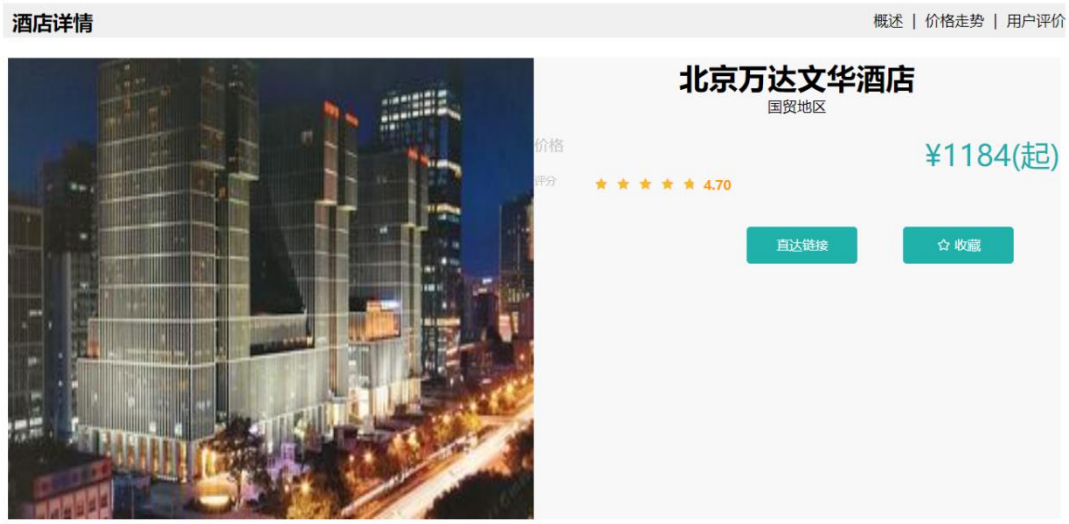
用户可以根据上方的搜索栏输入关键词进行搜索。关键词需要包含酒店名称信息，系统将会自动显示相似度最高的酒店信息页面。搜索功能使用的是 django 自带的模糊检索功能，大致算法为：如果酒店名包含搜索字符串，则显示该酒店名；若所有的酒店名都不包含搜索字符串，则有很大可能是部分字符出错。此时从搜索字符串中去除某个字符，将剩下的字符串再进行模糊检索。

(7) “为你推荐”

系统根据用户的收藏夹信息和预订记录等信息，以酒店类型和价格为标准，计算出和用户记录中相似度最高的酒店，并推荐在主页面下方。

(8) 酒店详细信息展示

用户在主页面点击某一个酒店图片后进入酒店详细信息界面。在酒店介绍部分将包含酒店名称、酒店地址、最新价格、评分、用户评价等信息。

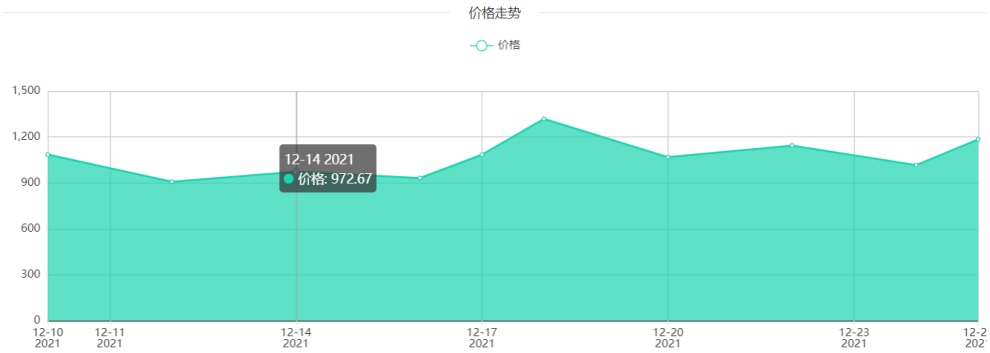


(9) 直达链接

用户想要进入酒店原始发布页面，可以点击酒店信息界面中的“直达链接”按钮，将跳转到酒店页面。

(10) 酒店近两周价格趋势

在酒店介绍的下方会显示酒店近两周以来的价格变化趋势图，可以让用户了解酒店价格的变化情况，便于用户进行预订时间选择。



(11) 预订房间

价格走势图的下方有预定房间功能，用户可查看此酒店的不同类型房间的价格和不同日期所剩数目信息，并选择入住日期和房间类型进行预订。

(12) 评分功能展示

预定房间的下方有酒店评分功能，用户可以根据自己的入住体验对酒店进行评分，评分后会会自动更新酒店评分数据，并触动触发器更新评分榜。

(13) 留言和回复功能

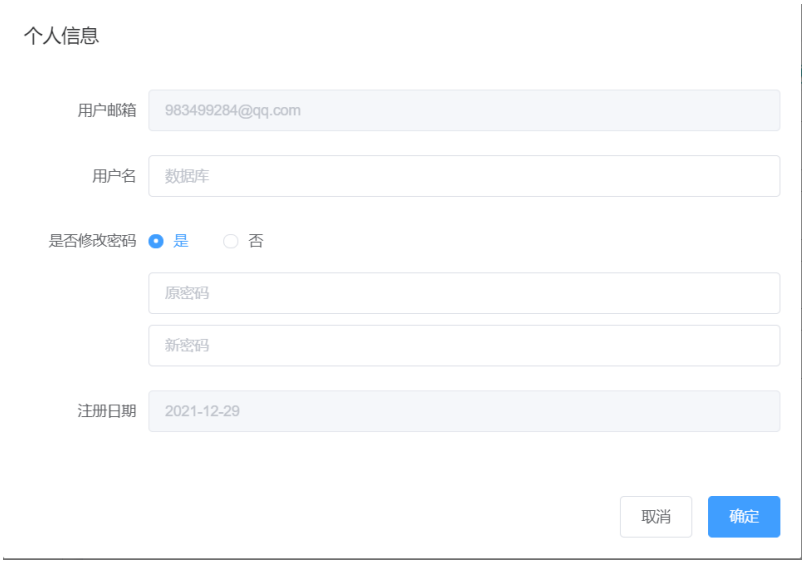
在评分栏的下方有留言功能和留言回复功能，用户可以点击相应按钮添加留言和进行留言回复。对于留言内容，进行了是否为合法内容的审核，若留言内容包含非法字符，则会显示

(14) 用户中心

点击主页面上方的用户名或者用户中心即可进入用户中心，里面会显示用户详细信息。包括用户名、邮箱、用户类型和注册时间等信息。

(15) 个人信息修改

进入用户中心后可修改用户的个人信息，除了用户邮箱和注册日期之外，用户名和密码都可以进行修改。且修改密码会进行密码验证，如果输入不符合会显示密码错误。



(16) 收藏夹显示

在酒店界面中，用户可实现将酒店加入收藏夹的功能。进入用户中心后，用户可根据用户中心左侧的选择栏查看收藏夹，并查看酒店的详细信息，进行删除或清空等操作。

(17) 浏览记录展示

用户每次进入一个酒店页面，就会更新一次浏览记录。如果此酒店之前浏览过，则更新浏览时间；否则在浏览记录中创建一个新的记录。进入用户中心后，用户可根据用户中心左侧的选择栏查看收藏夹，并查看酒店的详细信息，进行删除或清空等操作。

(18) 登出

用户可点击左侧的选择栏中的登出进行登出。登出后用户所能体验的功能将受到限制，比如不能留言和回复、不能收藏商品、不能进入个人中心等。

c) 商家模式

商家模式具有用户模式下的所有功能，此外还具有发布酒店和查看所发布酒店的功能。

（1）查看和更改已发布酒店信息

商家在用户界面中可以查看已发布的酒店信息，会显示酒店名、酒店地址、酒店经纬度、酒店注册时间、不同房间类型比例等信息，并将房间数量可视化。商家也可以对这些信息进行更改。

（2）发布酒店

商家可以在平台自行发布酒店，发布酒店需要填写酒店名、酒店地址、酒店图片，房间信息等信息，点击 logo 后可以从本地获取图片信息上传到网页。酒店发布成功后即可在平台中查到酒店信息。

(3) 导出酒店信息

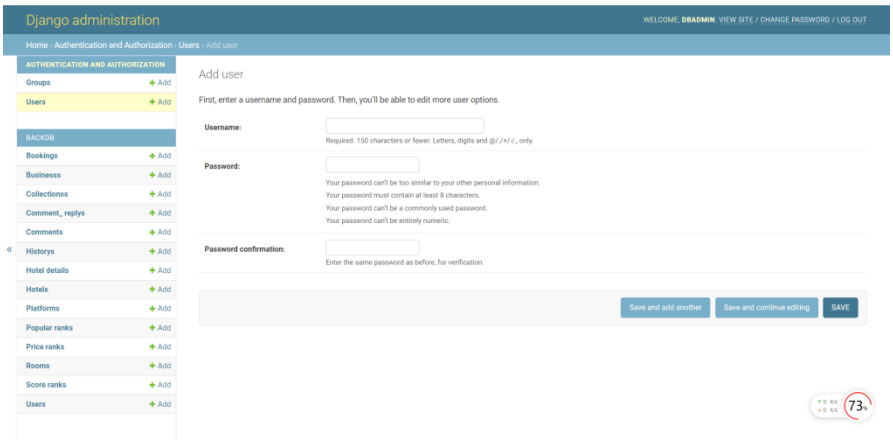
系统可以统计近几周时间里酒店的价格、销量，不同房间类型的价格和销量与日期的数据关系，商家能据此导出 Excel 或 PDF 表。

d) 管理员模式

管理员模式使用 django 自带基于 web 的 admin 管理工具，可以实现对数据的增加、删除、查看、审核等功能。

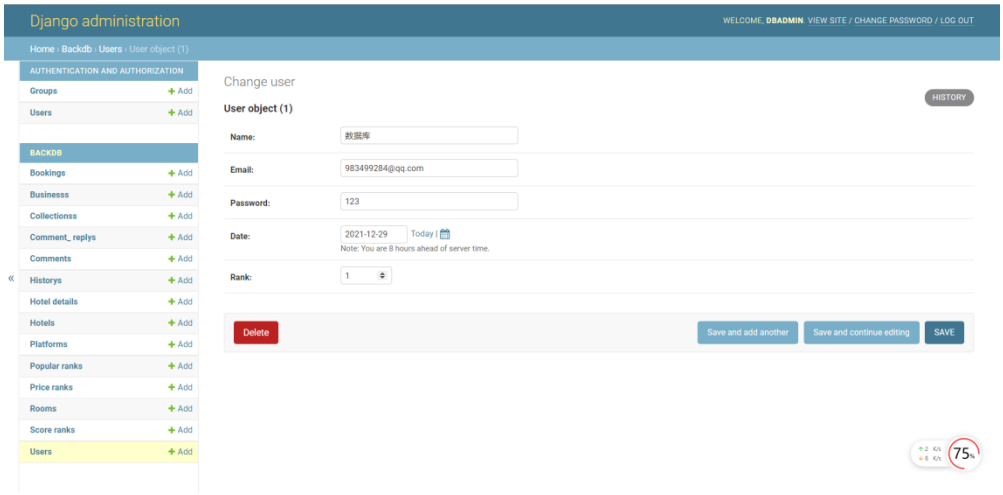
(1) 管理员注册

在命令行下注册一个超级用户后，登录管理员界面可以实现注册普通管理员，填入用户名、密码后即可完成注册。



(2) 数据审核、查看、删除

管理员模式下可以查看数据、对数据进行审核、删除等操作。



九、 亮点介绍

a) 功能全覆盖

通过上述功能展示，我们基本证明了平台的功能完整性。特别是我们对于基本上所有的用户，都设计了可以满足其需求的用户模式，以及相应的功能。

对于用户而言在当下这个数据爆炸、平台价格不透明的环境下，在一个平台即可分析对比三家平台的酒店信息，并且基本完全覆盖了特定地区的全部酒店，并且可以集中可视化，更包含了收藏、排行等人性化功能，这无疑对于酒店的选择大大地进行了简化。

对于商家而言，平台上酒店的发布无疑是复杂的，而本项目设计的商家模式可以十分方便简单的完成酒店的发布。并且，商家还可以将酒店的信息以 excel 的格式进行导出，这对于酒店规划、销量分析来说无疑是极大的利好。并且，商家还可

以在本平台通过酒店的信息进一步分析出品台的信息，这对于其之后选择发布平台也十分有用。

b) 服务器

我们将 Django 后端和数据库都部署在了宝塔服务器上，可以实现每天定时爬取酒店数据以及后端的实时响应、数据库的随时访问等等。将庞大的后端以及数据放上云端无疑是为了之后进一步维护、方便升级等等操作的一大利好，并且，也是因为数据库放上了服务器，我们前端的可视化界面是可以按照用户的喜好来进行实时更新的，对于可视化这一功能来说不可或缺。

c) 可视化

前端上的可视化地图可以根据用户的喜好来选择，根据喜爱值、收藏值等信息，和时间信息完成对于酒店的数据钻取，并且实时在地图上显示出来。这不但需要我们的后端完成对于数据库视图的操作，并且还借助了数据库在服务器上可以被实时读取这一特性。我们的可视化地图支持鼠标的拖动操作、滚轮的放大缩小操作，并且其热力值也会随之变化，对于希望对于酒店分布有一个直观印象的用户以及商家来讲用处极大。

d) 安全性

i. 支持多个类别的用户

将所有可能的用户都进行了分类，最终形成了管理员、商家、酒店三个用户类别。并且将所需要的功能完成了分类提供。

ii. 提供用户的注册、审核、权限分配、登录等功能

对于酒店订购、发布等等的设计金钱的功能，我们都对其设立了权限，只有当用户的类别符合、登录信息正常时才能令其操作。并且对于收藏这一类人性化的服务，我们也对用户开放。

iii. 能够记录和获取用户的访问日志，用户对于数据的操作

一个平台的稳定性以及安全性的最大保障实际上要依赖于溯源功能，也就是对于用户操作、访问日志的记录。有了这一功能，将极大地方便日后的进一步开发，例如对于用户日志的分析以防恶意评论，对于用户规律的分析找出活跃用户、封禁不正常用户等等。

iv. 安全管理人员对日志进行浏览、查询和统计分析

对于网站的管理人员来讲，其可以方便地使用我们设计的管理员模式对于用户数据、酒店信息、数据艰苦信息进行分析、操作。并且可以将数据以 PDF 格式进行导出，方便了酒店的管理。

十、 源程序简要说明

前后端所有接口如下图所示，这里简要说明部分接口的功能。

```
path('login/', login),
path('hotels/', hotels),
path('popular/', popular),
path('getAllCollections/', get_all_collections),
path('rankAll/', rankAll),
path('addHistory/', addHistory),
path('price/', price),
path('leavemessage/', leavemessage),
path('replymessage/', replymessage),
path('user/', user),
path('getAllHistories/', getAllHistory),
path('delAllHistories/', delAllHistories),
path('delAllCollections/', delAllCollections),
path('updateUser/', updateUser),
path('updateLocHotel/', updateLocHotel),
path('addLeaveMessage/', addLeaveMessage),
path('addReplyMessage/', addReplyMessage),
path('addScore/', addScore),
path('addCollection/', addCollection),
path('upload_logo/', update_logo),
path('MapUpdate/', map_update),
path('queryRoomInfo/', queryRoomInfo),
path('book/', book),
path('delHistory/', delHistory),
```

(1) Login

登录：前端传来邮箱和密码信息，后端检测信息是否正确，并将信息返回给前端。

(2) Hotels/popular

推荐酒店：前端传来用户信息，后端根据用户信息推荐相关酒店。

(3) Getallcollections

显示收藏夹：前端传来用户信息，后端根据用户信息加载收藏夹信息。

```
getAllCollections() {
    this.$axios.post('/getAllCollections/', this.ColAndHis)
    .then(resp=>{
        if (resp && resp.data.code === 200) {
            this.collections = resp.data.data
        }
        else {
            console.log(resp.data.code);
            this.$router.replace('/mainpage')
        }
    })
},
```

```
def get_all_collections(request):
    body_json = request.body.decode()
    body_dict = json.loads(body_json)
    user_id = body_dict.get('userid')
    collections = list(Collections.objects.filter(user_id=user_id))
    hotels = [collection.hotel for collection in collections]
    data = []
    for x in hotels:
        dic = x.to_dict()
        dic.pop('url')
        dic['time'] = Collections.objects.get(Q(user_id=user_id) & Q(hotel_id=x.id)).time
        data.append(dic)
    data = {'code': 200, 'msg': 'success', "data": data}
    resp = JsonResponse(dict(data))
    return resp
```

(4) AddLeaveMessage

留言：前端将留言信息传送给后端，后端将留言信息添加到数据库中。

```
addLeaveMessage() {
    if (this.$store.state.user.id === '') {
        this.$alert('请先登录', '提示', {
            confirmButtonText: '确定'
        })
        return
    }
    this.$refs.leaveMessageArea.dialogFormVisible = true
    this.$refs.leaveMessageArea.leaveMessage = {
        userid:this.$store.state.user.id,
        comid:this.$store.state.commodity.id,
    }
},
```

```
def addLeaveMessage(request):
    body_json = request.body.decode()
    body_dict = json.loads(body_json)
    print(body_dict)
    user_id = body_dict.get('userid')
    hotel_id = body_dict.get('comid')
    date = body_dict.get('time')
    context = body_dict.get('message')
    Comment.objects.create(user_id=user_id, hotel_id=hotel_id, date=date, context=context)
    data = {'status': 200, 'msg': '成功'}
    resp = JsonResponse(dict(data))
    return resp
```

(5) Price

价格趋势图：前端将酒店信息传送至后端，后端将此酒店对应的价格随时间数据返回给前端，其效果在前面已展示。

十一、 工作分工

a	b	c
前期调研分析系统需求 管理员界面 用户界面 商家界面 安全性部分 大报告书写	设计数据库关系模式 规范数据关系 3NF 撰写设计报告 后端相关工作 数据库管理 大报告书写	前期调研分析系统需求 爬虫代码编写 可视化部分编写 服务器相关工作 部分后端工作 大报告书写

十二、 收获体会

学习到了前端、后端、数据库之间的交互过程，知道了三者之间协同的重要性。各个部分有不同的功能，开项目设计阶段就要对功能进行划分并商定如何交互，之后三者之间完美协作才能做成一个好的项目。

学习到了数据库设计对于后端和前端功能实现的重要性，数据库是整个项目的基础。数据库的设计将直接决定后端的实现，所以在项目设计阶段就要对数据库进行一个良好的设计，避免之后出现较大的改动。