华东师范大学软件学院实验报告

实验课程：数据库系统及其应用　　**年级：**大二　　**实验成绩：**

实验名称：课程项目　　**姓名：**李嘉睿

实验编号：project　　**学号：**10175101250　　**实验日期：**2019/5/21

指导教师：宫学庆　　**机号：**50　　**实验时间：**10:00-11:30

一、项目简介

本项目构建了一个餐饮打分和评论系统。通过本网站，用户可以查找餐馆、浏览餐馆的位置信息、评分信息、图片展示信息以及其它用户对该餐馆做出的评论，同时注册账户在登录后能够对餐馆进行评论和打分，也可以对其它用户的评论进行投票。管理账户能够登录后台管理系统查看用户信息、公司信息、评论信息等，并能够进行一定的修改。

本项目后端使用 python 语言，后端框架使用 django，前端框架使用 bootstrap，开发环境使用 PyCharm。

二、数据的处理

1. 本项目使用的数据集是 Yelp Dataset(https://www.yelp.com/dataset)，数据集的基本情况如下：该数据集包含 6685900 条用户对公司的评论数据、192609 条商业公司信息数据、200000 张图片数据、1223094 条用户 tips 数据以及 1637138 条用户数据。

数据提供的格式为 json 格式，例如用户对公司的评论对应的 json 数据格式类似于：

review.json

Contains full review text data including the user_id that wrote the review and the business_id the review is written for.

```
{
    // string, 22 character unique review id
    "review_id": "zdSx_SD6obEhz9VrW9uAWA",

    // string, 22 character unique user id, maps to the user in user.json
    "user_id": "Ha3iJu77CxlrFm-vQRs_8g",

    // string, 22 character business id, maps to business in business.json
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",

    // integer, star rating
    "stars": 4,

    // string, date formatted YYYY-MM-DD
    "date": "2016-03-09",

    // string, the review itself
    "text": "Great place to hang out after work: the prices are decent, and

    // integer, number of useful votes received
    "useful": 0,

    // integer, number of funny votes received
    "funny": 0,

    // integer, number of cool votes received
    "cool": 0
}
```

照片和 tips 的 json 格式数据模板如下：

tip.json

Tips written by a user on a business. Tips are shorter than reviews and tend to convey quick suggestions.

```
{
    // string, text of the tip
    "text": "Secret menu - fried chicken sando is da bombbbbbb Their zapato

    // string, when the tip was written, formatted like YYYY-MM-DD
    "date": "2013-09-20",

    // integer, how many compliments it has
    "compliment_count": 172,

    // string, 22 character business id, maps to business in business.json
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",

    // string, 22 character unique user id, maps to the user in user.json
    "user_id": "49JhAJh8vSQ-vM4Aourl0g"
}
```

photo.json

Contains photo data including the caption and classification (one of "food", "drink", "menu", "inside" or "outside").

```
{
    // string, 22 character unique photo id
    "photo_id": "_nN_DhLXkfwEkwPNxne9hw",
    // string, 22 character business id, maps to business in business.json
    "business_id" : "tnhfDv5Il8EaGSXZGiuQGg",
    // string, the photo caption, if any
    "caption" : "carne asada fries",
    // string, the category the photo belongs to, if any
    "label" : "food"
}
```

通过文本编辑器打开的基本情况如下：

如图所示，数据文件的每一行对应一个 json 记录。

2. 对数据集的基本处理如下：首先构建数据的关系模式，通过编写 python 脚本文件，将数据逐一取出并进行响应裁剪，留下有用的数据，然后插入数据库中。脚本文件大致情况如下：

```
def f_user():
    f = open(r'D:\linux\user.json', "r", encoding='utf-8')
    line = f.readline()
    i = 0
    while line:
        obj = json.loads(line, encoding='utf-8')
        print(i)
        try:
            if User.objects.get(username=i):
                i += 1
                continue
        except ObjectDoesNotExist:
            pass
        user = User(username=i, password=i)
        user_name = obj['name']
        user_identifier = obj['user_id']
        try:
            if UserProfile.objects.get(user_identifier=user_identifier):
                i += 1
                continue
        except ObjectDoesNotExist:
            pass
        user_review_count = obj['review_count']
        user_date = obj['yelping_since']
        user_average_stars = obj['average_stars']
        user_fans = obj['fans']
        user_useful_votes = obj['useful']
        user_funny_votes = obj['funny']
        user_cool_votes = obj['cool']
        user.save()
```

具体可见源码中的 mysite/script.py 文件。

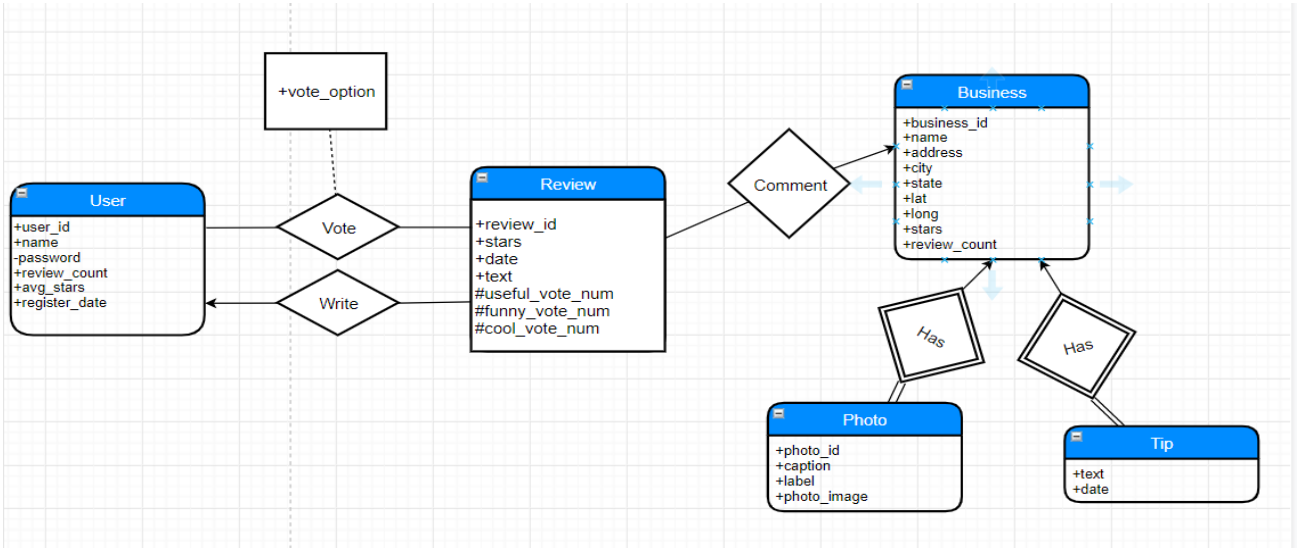
3. 对数据库设计的 ER 图如下：

3.1 对建模数据的假设：

①每个用户对一个同一个商业公司不同时间能评论任意次

②tips 由顾客在公司消费后给出，不对应网站的注册用户，属于公司展示数据，因此和公司是 has 关系

3.2 建模数据的 ER 图



4、数据库设计的关系模式

①business 表

```
CREATE TABLE `scoringsystem_business` (
`id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
`business_id` varchar(22) NOT NULL UNIQUE,
`business_name` varchar(100) NOT NULL,
`business_address` varchar(100) NOT NULL,
`business_city` varchar(100) NOT NULL,
`business_state` varchar(100) NOT NULL,
`business_postal_code` varchar(20)NOT NULL,
`business_lat` double precision NOT NULL,
`business_long` double precision NOT NULL,
`business_stars` double precision NOT NULL,
`business_review_count` integer NOT NULL,
`business_mon` varchar(20) NOT NULL,
`business_tue` varchar(20) NOT NULL,
`business_wed` varchar(20) NOT NULL,
`business_thu` varchar(20) NOT NULL,
`business_fri` varchar(20) NOT NULL,
`business_sat` varchar(20) NOT NULL,
`business_sun` varchar(20) NOT NULL);
```

②review、vote、tip 表

```
CREATE TABLE `scoringsystem_review` (
`id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
`review_stars` double precision NOT NULL,
`review_text` longtext NOT NULL,
`review_date` datetime(6) NOT NULL,
`review_useful_vote` integer NOT NULL,
`review_funny_vote` integer NOT NULL,
`review_cool_vote` integer NOT NULL,
`review_business_id_id` integer NOT NULL,
`review_user_id_id` integer NOT NULL);


CREATE TABLE `scoringsystem_vote` (
`id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
`vote_option` integer NOT NULL,
`vote_review_id` integer NOT NULL,
`vote_user_id` integer NOT NULL);

CREATE TABLE `scoringsystem_tip` (
`id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
`tip_text` varchar(100) NOT NULL,
`tip_date` date NOT NULL,
`tip_business_id_id` integer NOT NULL,
`tip_user_id_id` integer NOT NULL);

```

③photo 表和添加的外键约束

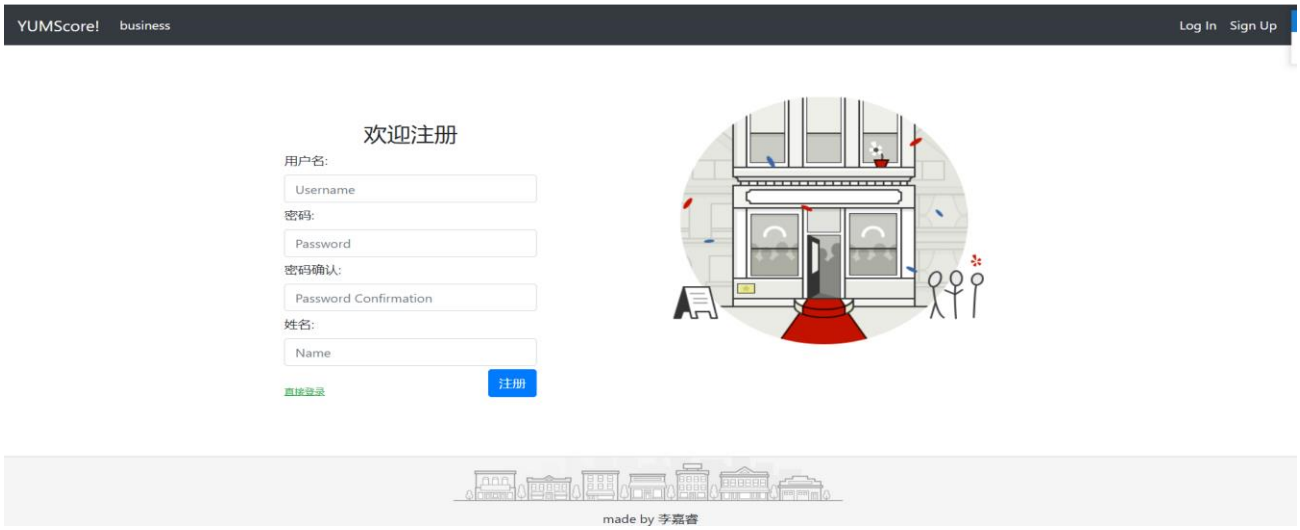
```
CREATE TABLE `scoringsystem_photo` (
`id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
`photo_caption` varchar(100) NOT NULL,
`photo_label` varchar(100) NOT NULL,
`photo_image` varchar(250) NOT NULL,
`photo_business_id_id` integer NOT NULL);

ALTER TABLE `scoringsystem_review` ADD CONSTRAINT `scoringsystem_review_review_business_id_i_d140ea9a_fk_scoringsy` FOREIGN KEY (`review_bu
S `scoringsystem_business` (`id`);
ALTER TABLE `scoringsystem_review` ADD CONSTRAINT `scoringsystem_review_review_user_id_id_e4d104a2_fk_login_use` FOREIGN KEY (`review_user_
n_userprofile` (`id`);
ALTER TABLE `scoringsystem_vote` ADD CONSTRAINT `scoringsystem_vote_vote_review_id_99c24bd6_fk_scoringsy` FOREIGN KEY (`vote_review_id`) RE
review` (`id`);
ALTER TABLE `scoringsystem_vote` ADD CONSTRAINT `scoringsystem_vote_vote_user_id_2adb2b85_fk_login_userprofile_id` FOREIGN KEY (`vote_user_
serprofile` (`id`);
ALTER TABLE `scoringsystem_tip` ADD CONSTRAINT `scoringsystem_tip_tip_business_id_id_0e36de7e_fk_scoringsy` FOREIGN KEY (`tip_business_id_i
ystem_business` (`id`);
ALTER TABLE `scoringsystem_tip` ADD CONSTRAINT `scoringsystem_tip_tip_user_id_id_cc48fb53_fk_login_use` FOREIGN KEY (`tip_user_id_id`) REFE
e` (`id`);
ALTER TABLE `scoringsystem_photo` ADD CONSTRAINT `scoringsystem_photo_photo_business_id_id_6b30974c_fk_scoringsy` FOREIGN KEY (`photo_busin
```

三、Web 接口

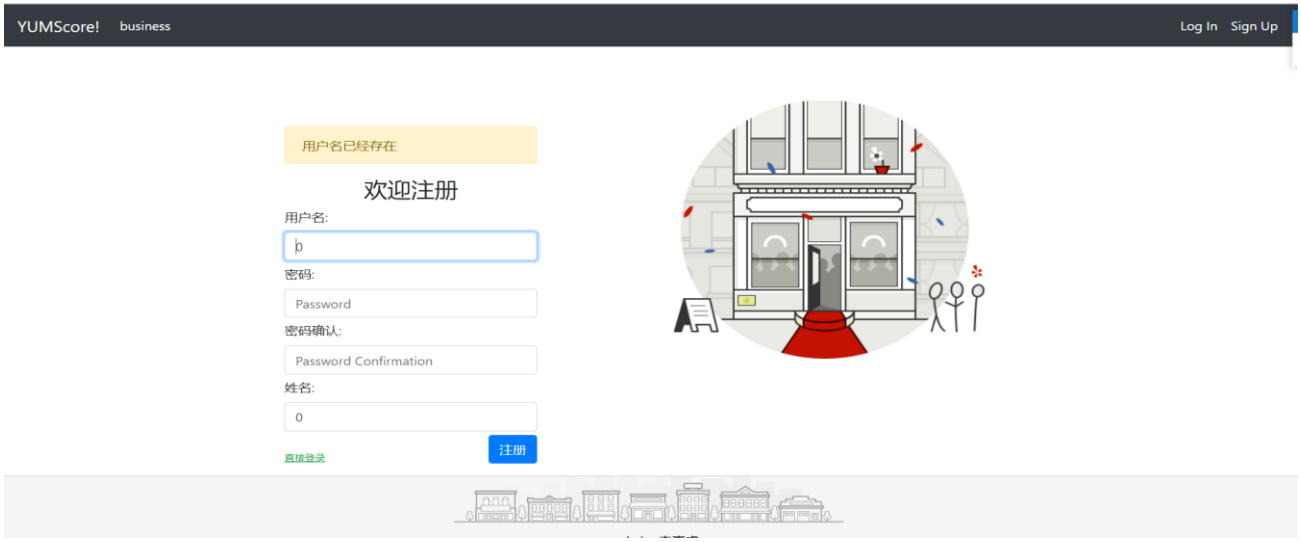
3.1 用户注册

如图，用户通过填写用户注册表单进行注册。系统首先查询数据库查看是否有同名的用户名，若有则注册失败并返回相应信息，若无则注册成功。

①注册页面



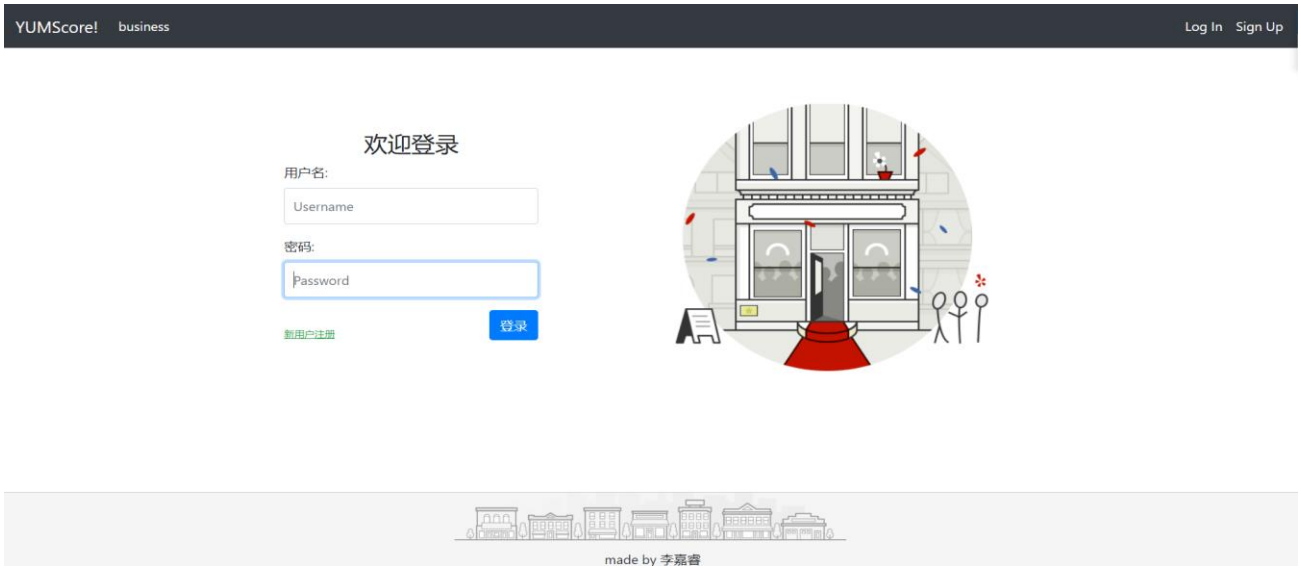
②用户名已经存在



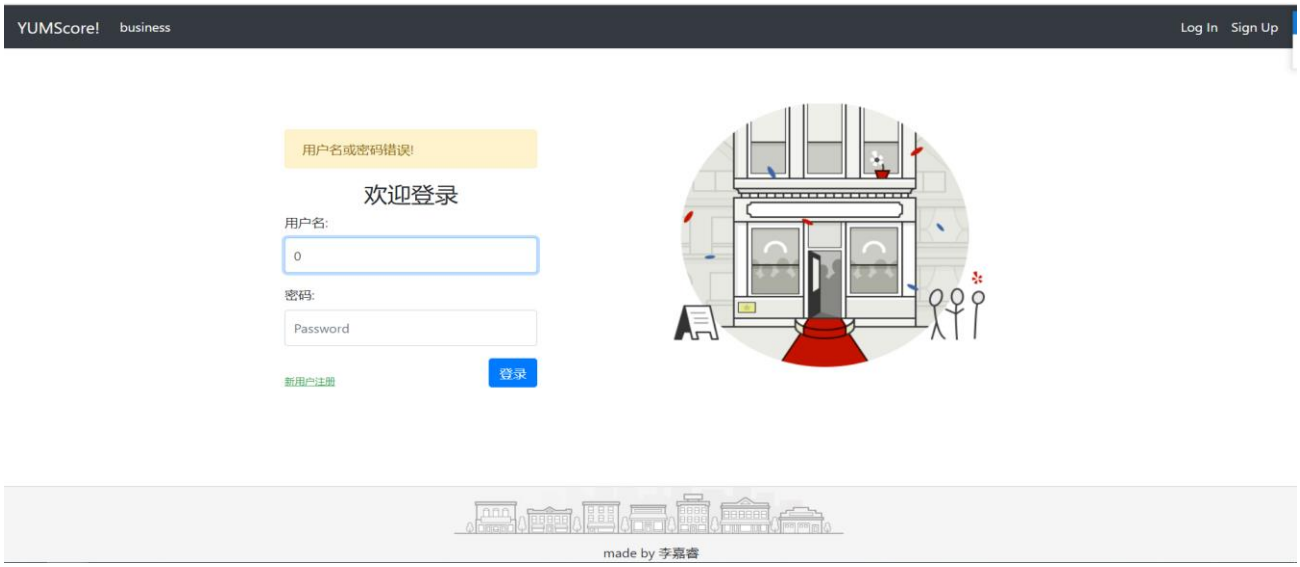
3.2 用户登录及用户信息修改

如图，用户通过填写用户名和密码完成登录，系统在数据库中查找相关数据，若能够匹配。则登录成功，反之登录失败，返回失败信息。

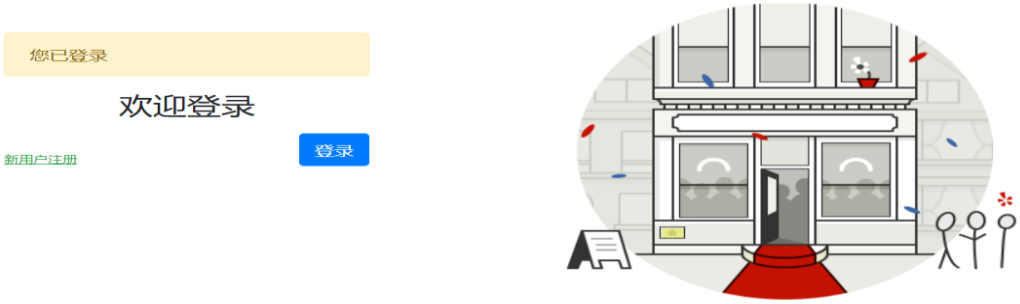
①登录页面



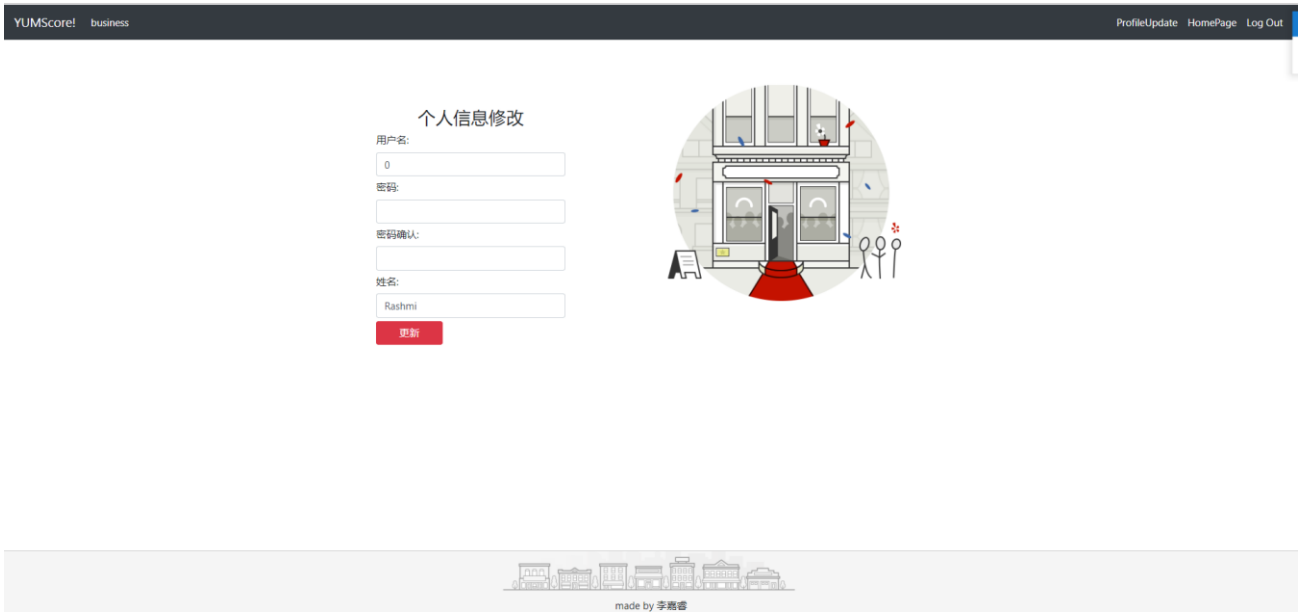
②登录失败



③登录成功



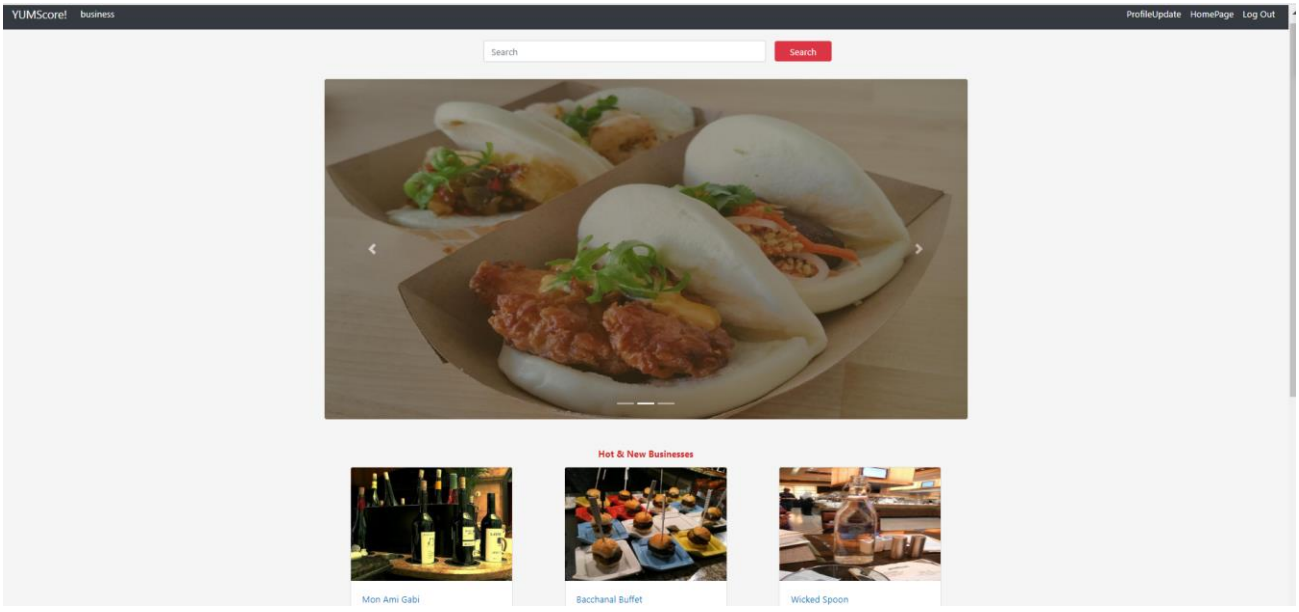
如图，用户登录后，能够通过填写信息修改表单，修改用户个人信息：



3.3 主页及相关操作

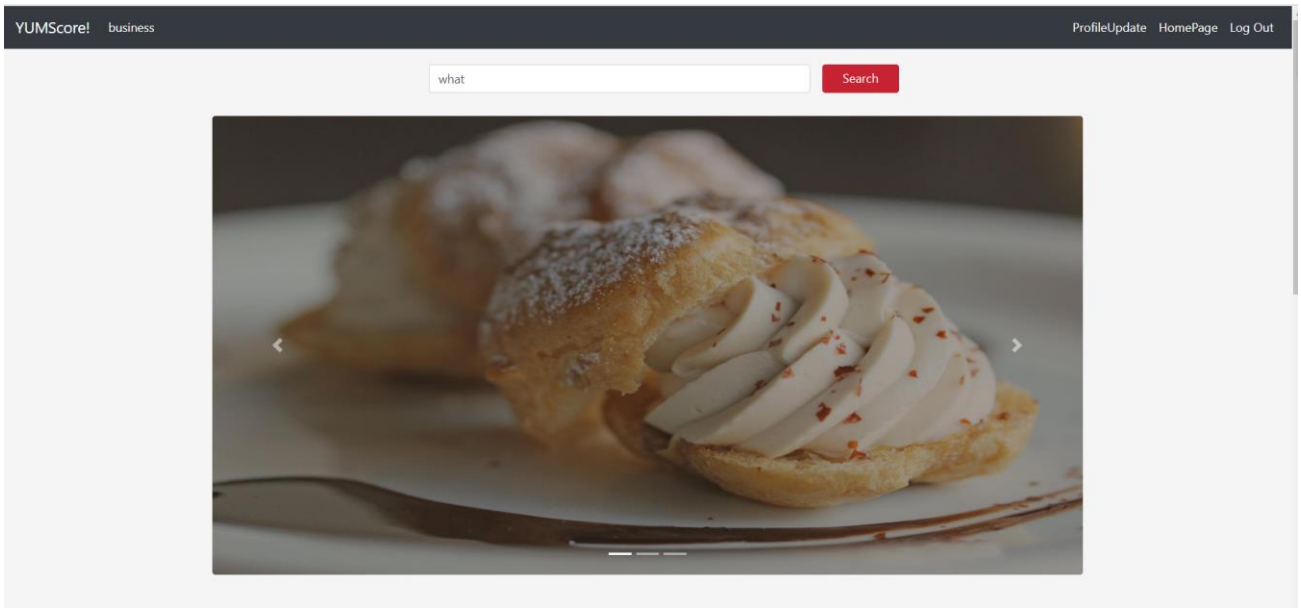
①主页显示

如图，进入主页后系统从数据库中查找餐饮公司信息及公司对应图片的位置信息并显示在网页上。

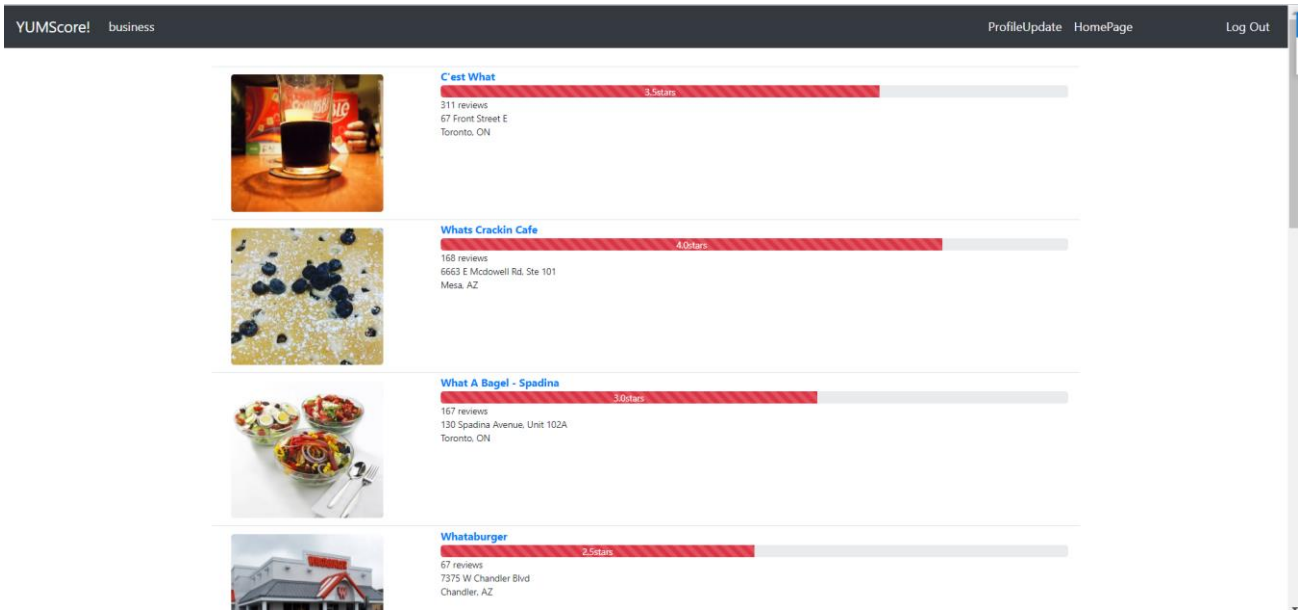


②餐饮公司查询

如图，用户通过填写搜索框关键字进行餐饮店的查找，系统通过关键字在数据库中查询，并将查询结果显示，例如填写‘what’字段的结果如下：

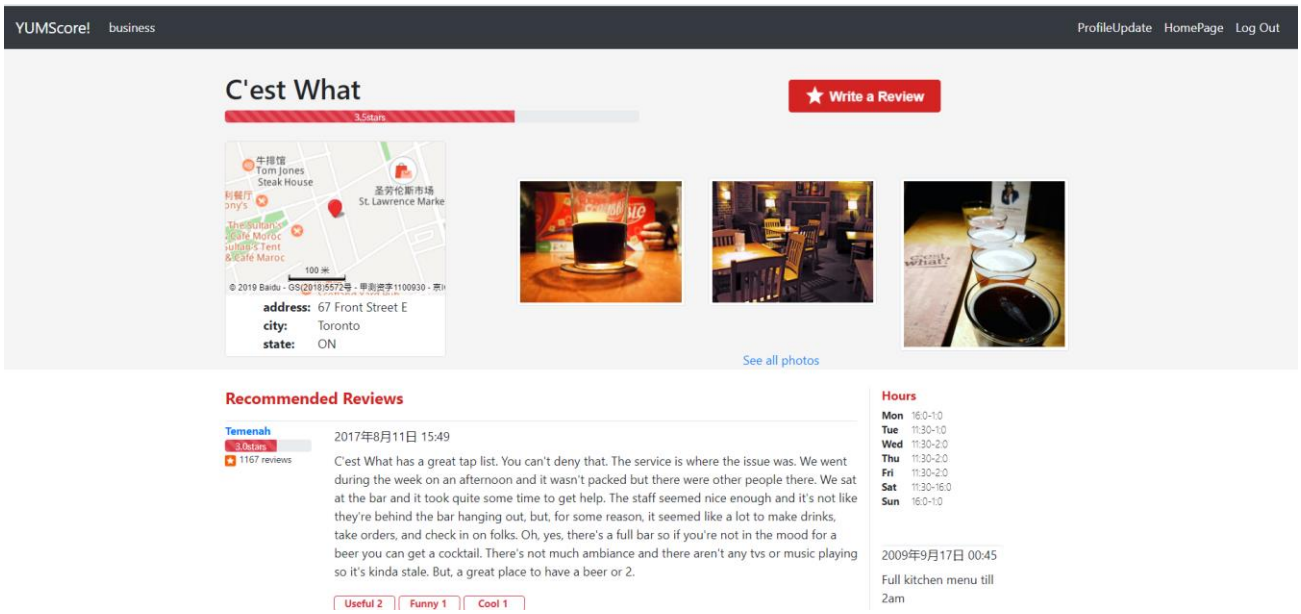


查询结果：



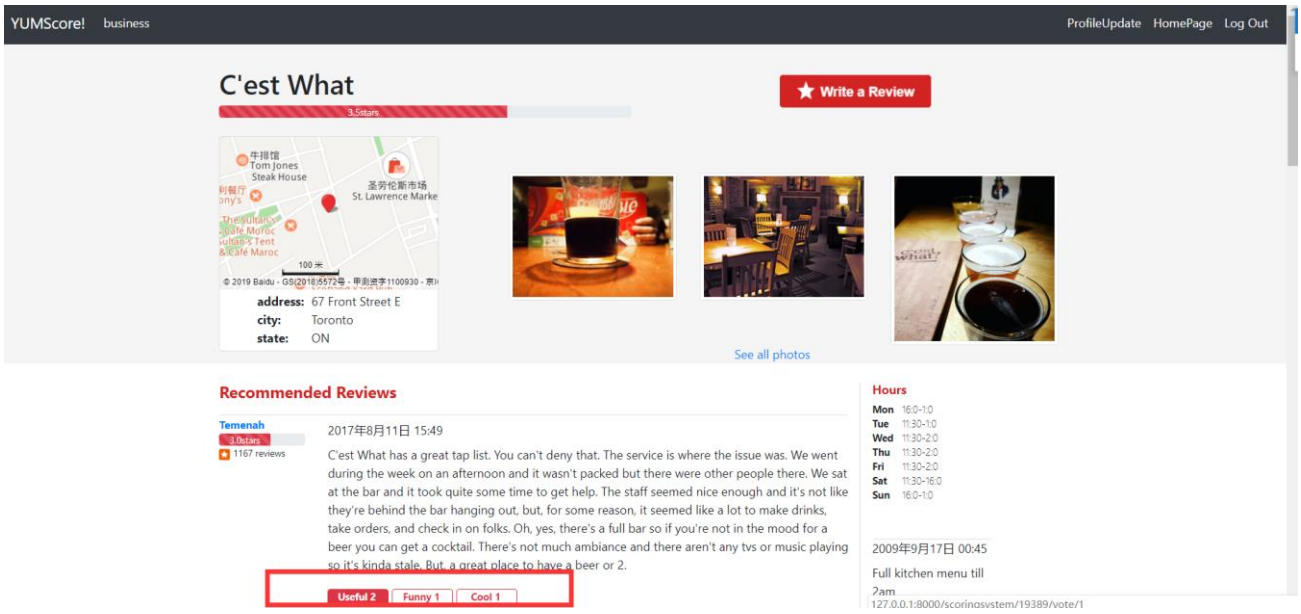
③餐饮公司信息显示

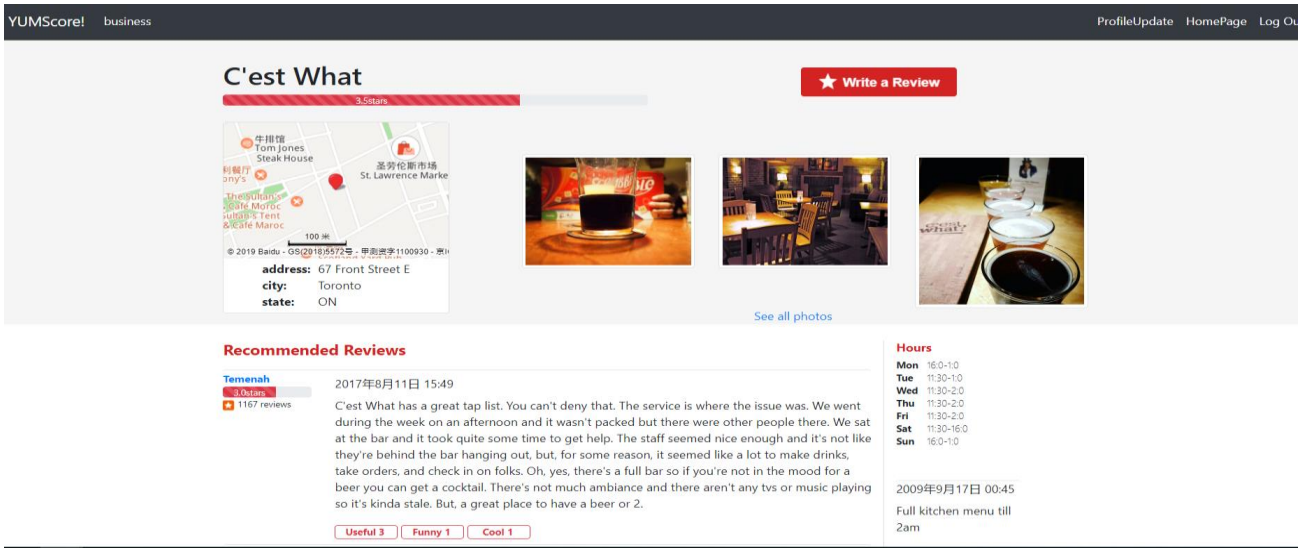
如图，用户点击某个餐饮公司后，进入餐饮公司页面，包含餐饮公司 id 值的请求被发送给系统，系统通过 id 值去数据库中查找餐饮公司的数据，通过数据渲染模板页面后返回：



④评论投票

如图，若非登录状态，点击投票按钮，系统会返回登录界面；若为登录状态则，第一次点击投票按钮会修改数据库中数据，第二次点击会撤销投票结果：

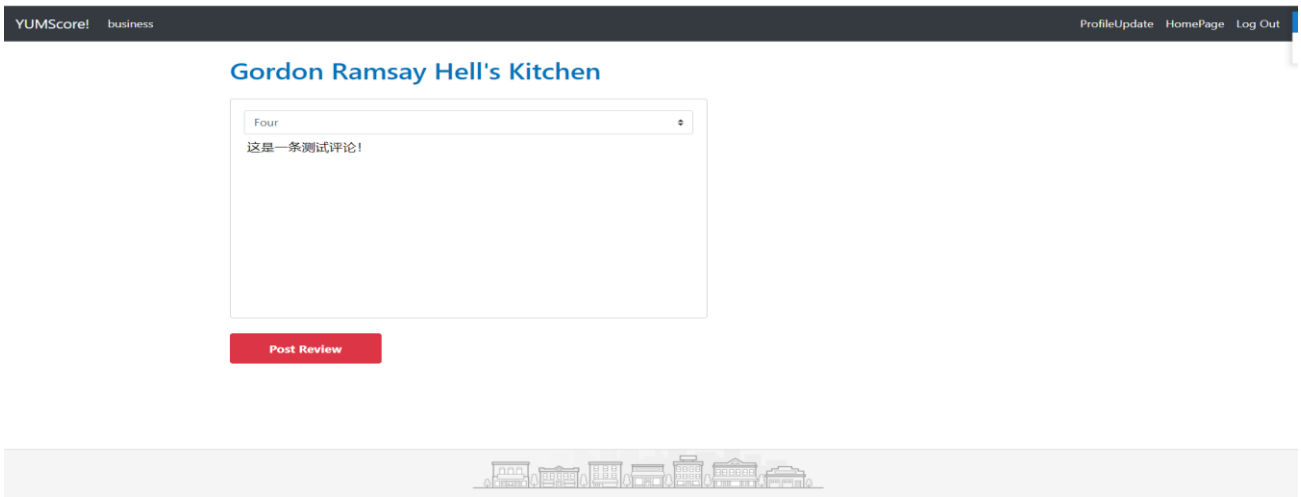




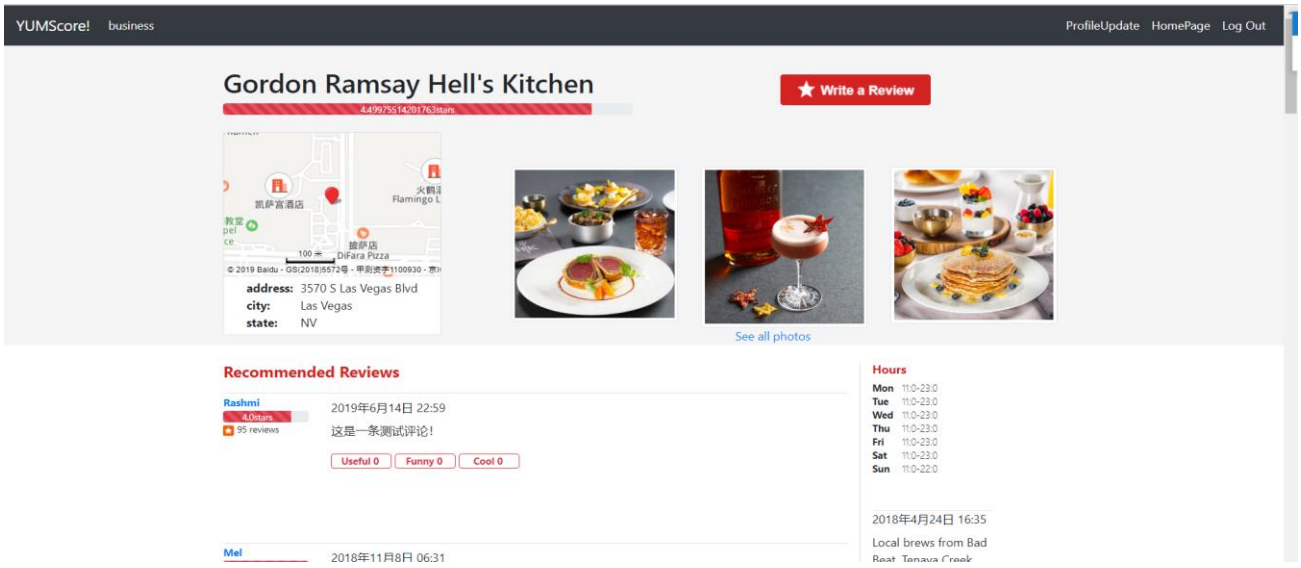
⑤撰写评论

如图，登录用户能够撰写评论，并能够选择评论的打分。系统接收到该评论和评分，将会修改数据库中餐饮公司的相关统计数据，示例如下：

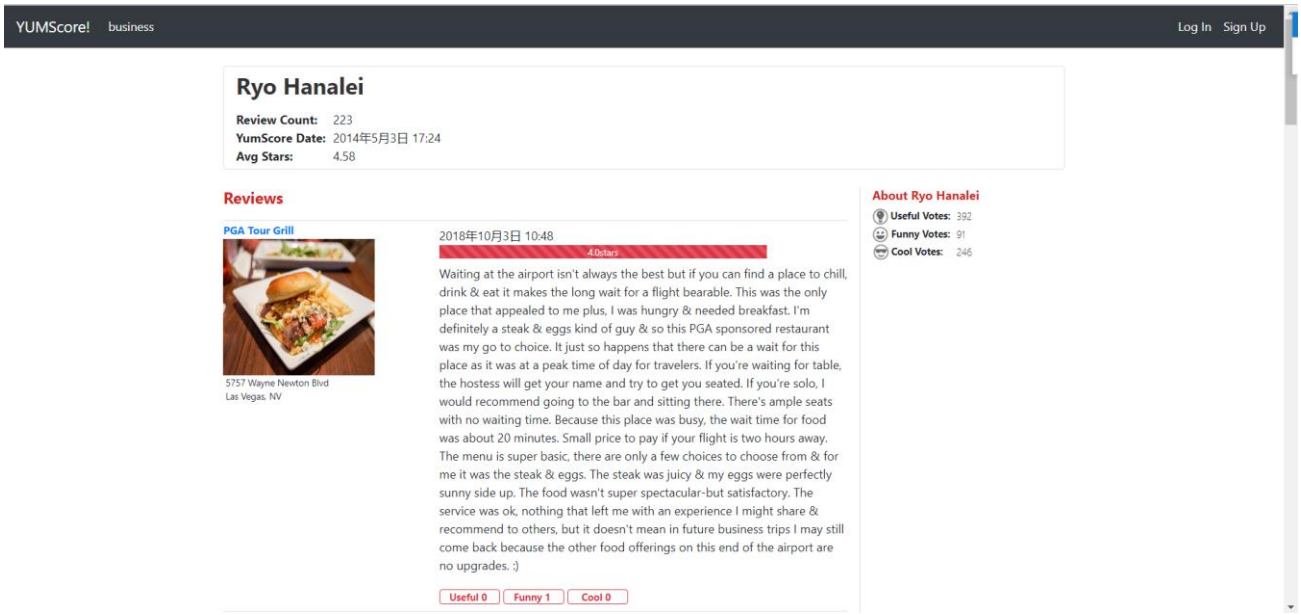
如图，这是评论提交的结果：



⑥用户空间查看

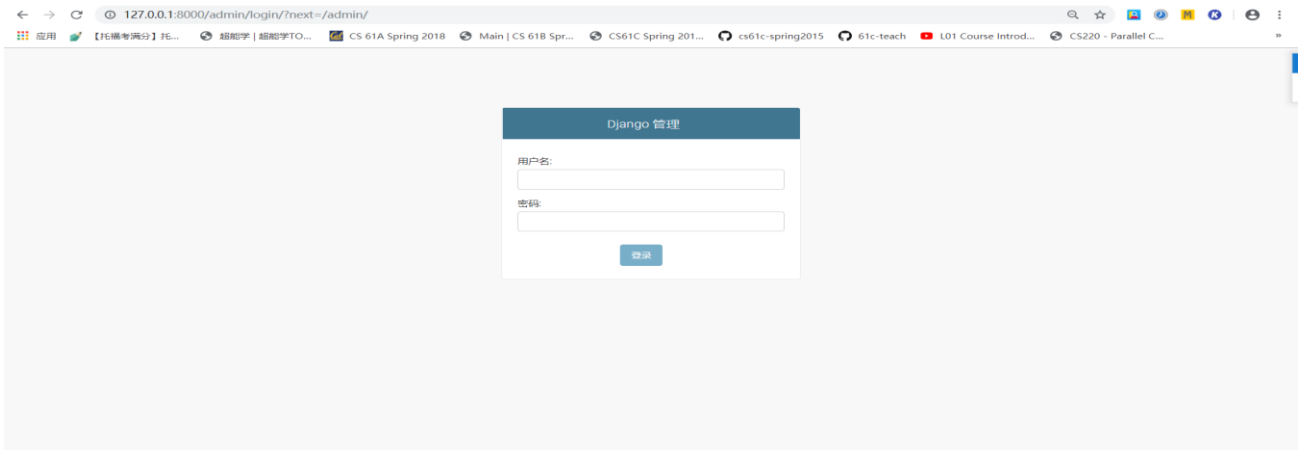


如图，通过将用户 id 传给系统，系统通过 id 在数据库中查找用户相关数据并显示



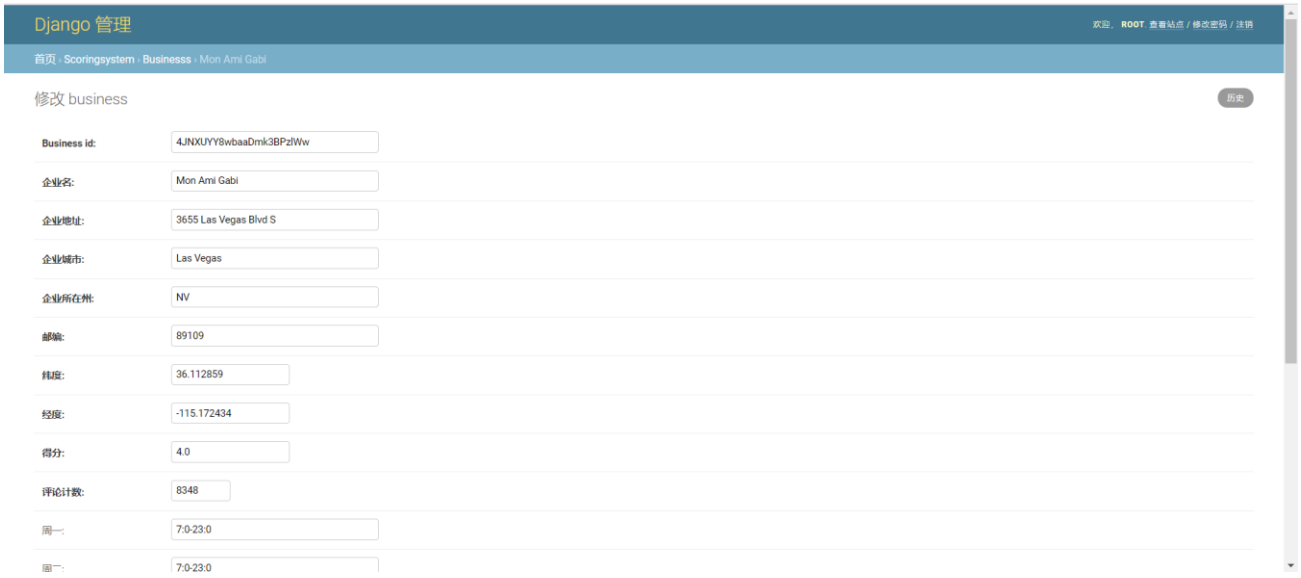
3.4 后台管理系统

如图，这是后台管理系统：



进入后台管理系统，能够对各种数据进行更新、删除等操作：

例如进入 business 的管理页面：



四、对数据库的修改以提高性能

①构建排序方式

```
class Review(models.Model):
    review_user_id = models.ForeignKey(
        UserProfile,
        on_delete=models.SET('用户已注销'),
        db_index=True
    )
    review_business_id = models.ForeignKey(
        Business,
        on_delete=models.CASCADE,
        db_index=True
    )
    review_stars = models.FloatField('评分', default=0)
    review_text = models.TextField('评论')
    review_date = models.DateTimeField('日期', default=timezone.now)
    review_useful_vote = models.IntegerField(default=0)
    review_funny_vote = models.IntegerField(default=0)
    review_cool_vote = models.IntegerField(default=0)

    class Meta:
        ordering = ['-review_date']
```

```
class Business(models.Model):
    business_id = models.CharField(max_length=22, unique=True)
    business_name = models.CharField('企业名', max_length=100, db_index=True)
    business_address = models.CharField('企业地址', max_length=100)
    business_city = models.CharField('企业城市', max_length=100)
    business_state = models.CharField('企业所在州', max_length=100)
    business_postal_code = models.CharField('邮编', max_length=20)
    business_lat = models.FloatField('纬度')
    business_long = models.FloatField('经度')
    business_stars = models.FloatField('得分', default=0.0, db_index=True)
    business_review_count = models.IntegerField('评论计数', default=0)
    business_mon = models.CharField('周一', max_length=20, blank=True)
    business_tue = models.CharField('周二', max_length=20, blank=True)
    business_wed = models.CharField('周三', max_length=20, blank=True)
    business_thu = models.CharField('周四', max_length=20, blank=True)
    business_fri = models.CharField('周五', max_length=20, blank=True)
    business_sat = models.CharField('周六', max_length=20, blank=True)
    business_sun = models.CharField('周日', max_length=20, blank=True)

    def __str__(self):
        return self.business_name

    class Meta:
        ordering = ['-business_review_count']
```

②构建索引

CREATE INDEX `scoringsystem_business_business_name_cfc96227` ON `scoringsystem_business` (`business_name`);

CREATE INDEX `scoringsystem_business_business_stars_13f00cad` ON `scoringsystem_business` (`business_stars`);

五、实验结果总结

在本次项目中，我使用 Django 框架从 0 开始构建了整个 Web 应用，从最初的实体关系模型设计到数据库表项设计，到网页前后端代码的编写，最终到数据集的导入。了解了 Web 开发中数据库的作用，了解了系统如何通过和数据库交互获得数据后，在网页上显示，或是用户通过和网页交互，触发系统对数据库的修改等操作。知道了数据库作为 web 应用性能瓶颈的原因，并能够通过添加索引改善查找性能，收益颇丰。