

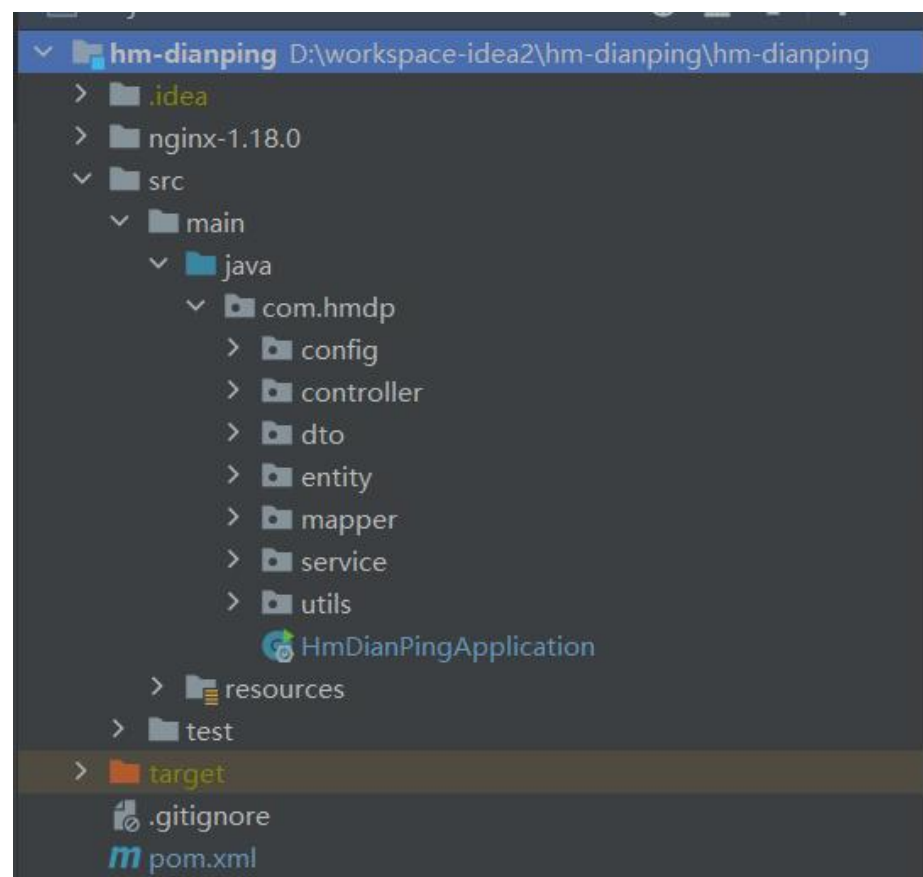
仿大众点评的小项目

一、项目简介

我们的项目是仿大众点评的小型 Java 项目，旨在为用户提供一个点评和分享平台。通过这个平台，用户可以浏览、搜索和比较各种商家的信息，包括美食、KTV 等等。用户可以在平台上创建个人账号、关注他人、发表评论、点赞、上传图片 and 发布自己的博客。

二、技术介绍

目录结构：



后端：

SpringBoot + Redis + MyBatisPlus + Mysql + Elasticsearch

前端:

Html + Axios + Vue + Nginx

开发工具:

JDK	1.8
MYSQL	8.0.25
Redis	6.2.6
Elasticsearch	7.12.1
Kibana	7.12.1
Nginx	1.18.0

三、核心功能

一、短信登录: Redis 的共享 session 应用。

①验证码采用 Redis 存储，时效为两分钟。

②考虑到集群 Session 共享的数据丢失问题，将 User 对象转为 HashMap 存储，并随机生成 Token 作为登录令牌存入 Redis 里，设置过期时间。

session 共享问题: 多台 Tomcat 并不共享 session 存储空间，当请求切换到不同 tomcat 服务时导致数据丢失的问题。

③采用双拦截器实现刷新 token 的有效期，拦截需要登录的路径两个功能。

二、商户查询缓存: 缓存雪崩、缓存穿透等问题解决。

①给查询商铺的缓存添加超时剔除和主动更新的策略。

缓存的作用：①降低后端负载；②提高读写效率，降低响应时间。

②实现商铺缓存与数据库的双写一致。

③采用缓存空对象的方式解决缓存穿透问题。

用户请求的数据在缓存中和数据库中都不存在，不断发起这样的请求，给数据库带来巨大的压力。

④采用设置逻辑过期时间以及互斥锁的方式来解决缓存击穿。

三、优惠券秒杀：涉及 Redis 的计数器、Lua 脚本 Redis、分布式锁、Redis 消息队列等。

①采用全局唯一 id 来解决 id 的规律性太明显且易受单表数据量的限制两大缺点。

②乐观锁解决超卖问题（多线程安全问题）。

③使用 lua 脚本改造分布式锁保证一人一单，且不被误删，但是它不可重入。

基于 Redis 的分布式锁的实现思路：①利用 set nx ex 获取锁，并设置过期时间，保存线程标识；②释放锁时判断线程标识是否与自己一致，一致则删除。

④使用 redission 实现可重入锁。原理：利用 hash 结构，记录线程标示和重入次数；利用 watchdog 延续锁时间；利用信号量控制锁重试等待。

⑤实现优惠券异步秒杀下单功能。采用 lua 脚本方式判断抢单资格并创建订单写入消息队列，获取消息队列中的订单并处理，并且在创建订单失败时再次读取 pending-list，解决了订单丢失的问题。

四、达人探店：基于 List 的点赞列表，基于 SortedSet 的点赞排行榜。

①发布并查看探店笔记：点击首页的探店笔记，会进入详情页面，可以在页面详情中进行点赞、评论和关注。

②点赞：同一个用户只能点赞一次，再次点击则取消点赞。如果当前用户已经点赞，则点赞按钮高亮显示。

③评论：根据博客 ID 查询对应的评论树，即评论及其回复的层次结构数据，通

过使用 `Lambdas` 表达式和 `Stream API`，对评论数据进行了过滤和转换，并将其组织成树形结构的数据，用 `SQL` 乐观锁解决评论数量数据不一致的问题。

④点赞排行榜：在探店笔记的详情页面，选择合适的数据结构将给该笔记点赞的人显示出来，例如将最早点赞的 `TOP5` 形成点赞排行榜。

五、好友关注：基于 `Set` 集合的关注、取关、共同消息推送等功能。

①关注和取关：在探店图文的详情页面中，可以关注发布笔记的作者。用户通过点击进行关注，再次点击则取消关注。

②共同关注：点击博主头像，可以进入博主首页，查看博主的探店笔记以及在博主个人页面展示出当前用户与博主的共同好友。

③关注推送：使用 `Timeline` 的模式里的推模式实现关注推送功能。当新增探店笔记的时候，可以推送到粉丝的收件箱，根据时间戳进行排序。

六、附近的商户：Redis 的 `GeoHash` 的应用。

①实现查看附近商铺的功能。按照商户类型做分组，类型相同的商户作为同一组，根据距离远近进行排序。

②实现筛选商铺的功能，类型相同的商户为同一组，根据距离、人气、评分进行排序展示。

③在搜索框实现自动补全功能：根据关键字搜索商铺，输入拼音等自动补全相关商铺全称，点击即跳转到商铺详情页。通过 `Elasticsearch` 进行搜索，通过输入关键词，返回满足该关键词前缀的补全建议结果。

七、用户签到：Redis 的 `BitMap` 的数据统计功能。

①签到功能：实现签到功能，将当前用户当天签到信息保存到 `Redis` 中。

②签到统计：统计当前用户截止当前时间在本月的连续签到天数。

从最后一次签到开始向前统计，直到遇到第一次未签到为止，计算总的签到次数，就是连续签到天数。

八、UV 统计：Redis 的 HyperLogLog 的统计功能。

①UV：全称 Unique Visitor，也叫独立访客量，是指通过互联网访问、浏览这个网页的自然人。1 天内同一个用户多次访问该网站，只记录 1 次。

②PV：全称 Page View，也叫页面访问量或点击量，用户每访问网站的一个页面，记录 1 次 PV，用户多次打开页面，则记录多次 PV。往往用来衡量网站的流量。

九、个人主页：点赞数量、评论数量、共同关注显示功能，个人信息修改功能。

①在我的笔记页面，改变点赞数量时，要清空缓存，实现数据库与缓存数据一致。

②点击“关注”，“粉丝”，展示我关注的人和关注我的人，点击对应按钮可以跳转到对应详情页。

③点击个人信息页，进行个人信息的编辑，进行即时修改，使用文本框、下拉框、日期选择器来修改个人介绍、昵称、性别、城市和生日的信息。

④上传个人头像：使用了数据库操作和 Redis 缓存更新用户的头像字段和其他信息。返回了上传文件的访问 URL，方便后续展示和访问上传的文件。

十、消息通知：评论信息的显示与移除。

①当“我“的博客出现新评论的时候，将该博客展示在消息通知页上，当点击进入博客详情页后，将该博客评论消息通知移除消息通知页。

四、项目展示

一、用户登录

验证码两分钟以内有效。未登录的时候可以访问主页，登录后可以对个人信息页进行编辑。可以修改昵称、性别、生日、城市等信息，同时也可以对头像进行上传。

手机号码快捷登录

46秒后可重发

618584

未注册的手机号码验证后自动创建账户

登录

验证码登录

我已阅读并同意 [《用户服务协议》](#)、[《隐私政策》](#) 等，接受免除或者限制责任、诉讼管辖约定等粗体标示条款

个人主页

可可今天不吃肉

退出登录

编辑资料

添加个人简介，让大家更好的认识你 [🔗](#)

笔记 关注 粉丝 推送

无尽浪漫的夜晚！在万花丛中摇曳着
红酒杯🍷品战斧牛排🥩

104 2

删除

人均30💰杭州这家港式茶餐厅我疯狂
打call!!

0 2

删除

首页 地图 + 消息 我的

资料编辑

头像

昵称 可可今天不吃肉 >

个人介绍 请介绍自己

性别 选择 ▾

城市 选择 ▾

生日 自

我的积分 查看积分 >

会员等级 成为VIP尊享特权 >

资料编辑

🟢 User information updated successfully

头像

昵称 可可今天不吃肉 >

个人介绍 我是可可哩

性别 女 ▾

城市 上海 ▾

生日 2023-10-02

我的积分 查看积分 >

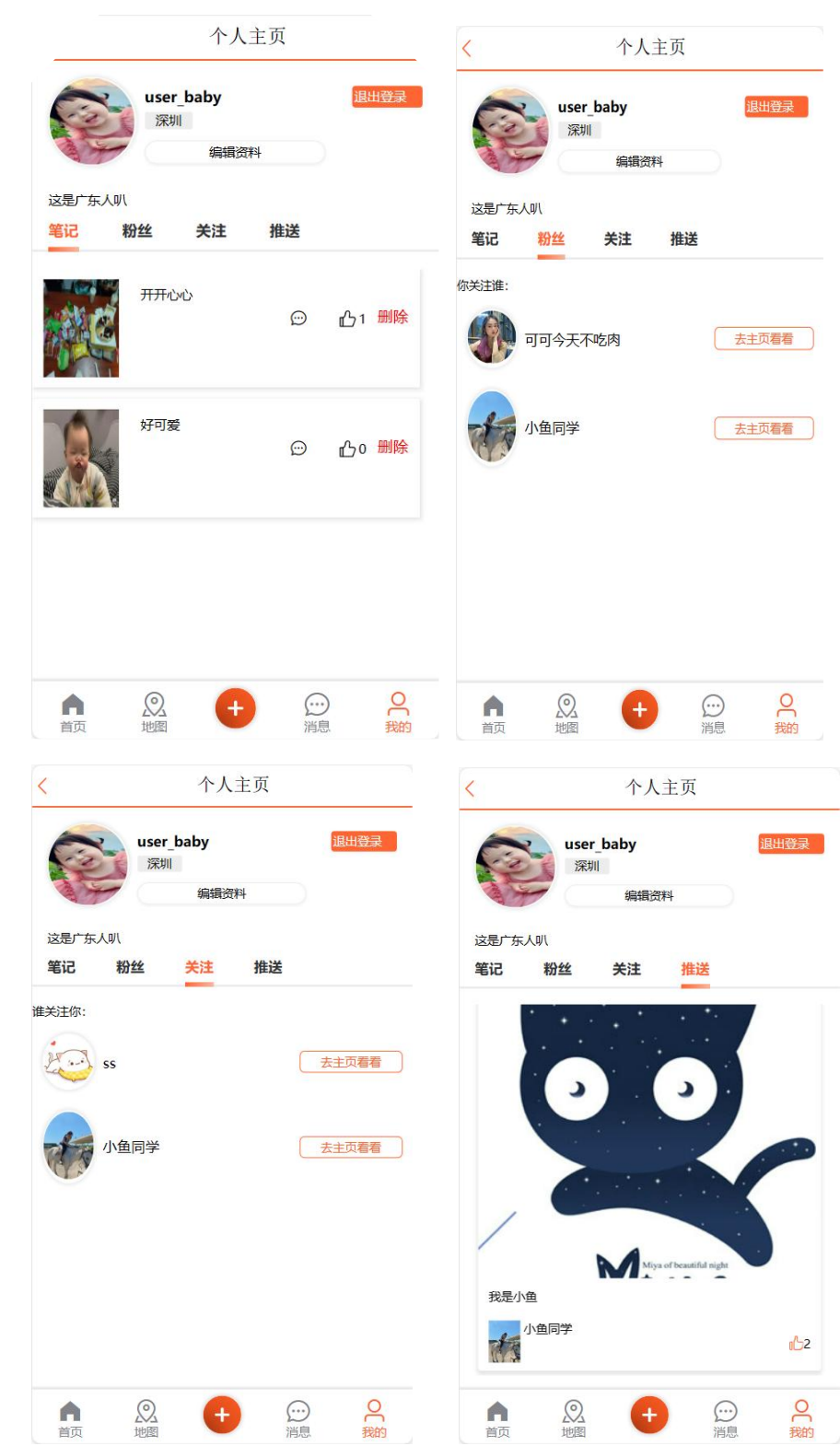
会员等级 成为VIP尊享特权 >

二、粉丝、关注、共同关注

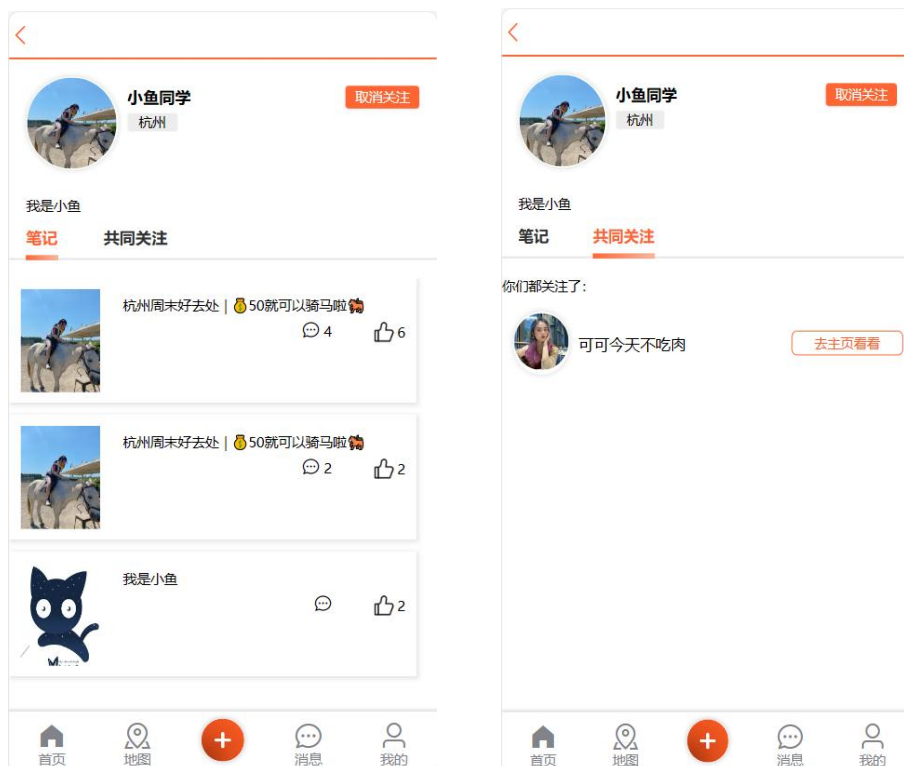
对自己发布的笔记可以进行点赞以及删除。点击“关注”可以查看“我的关

注”，点击“粉丝”可以查看“谁关注我”，再点击相应“去主页看看”可以进行跳转。

当新增探店笔记的时候，可以推送到粉丝的收件箱，根据时间戳进行排序，同时提供点赞功能。

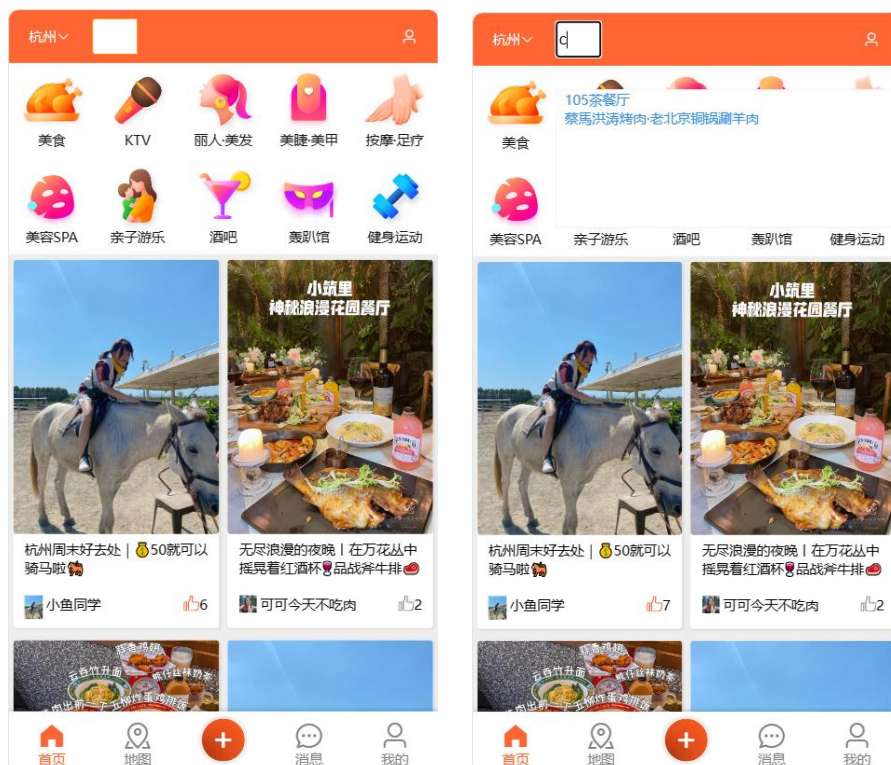


点进去“去主页看看”，可以查看共同关注。

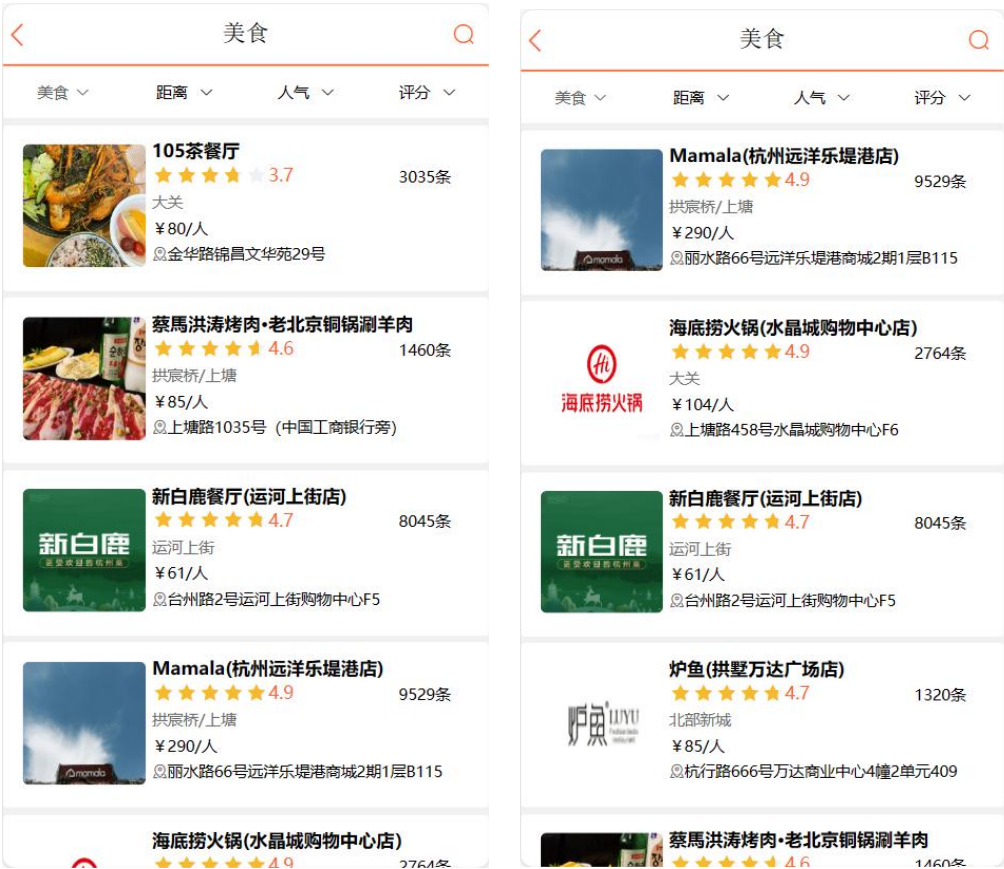


三、在主页上查看商铺和博客

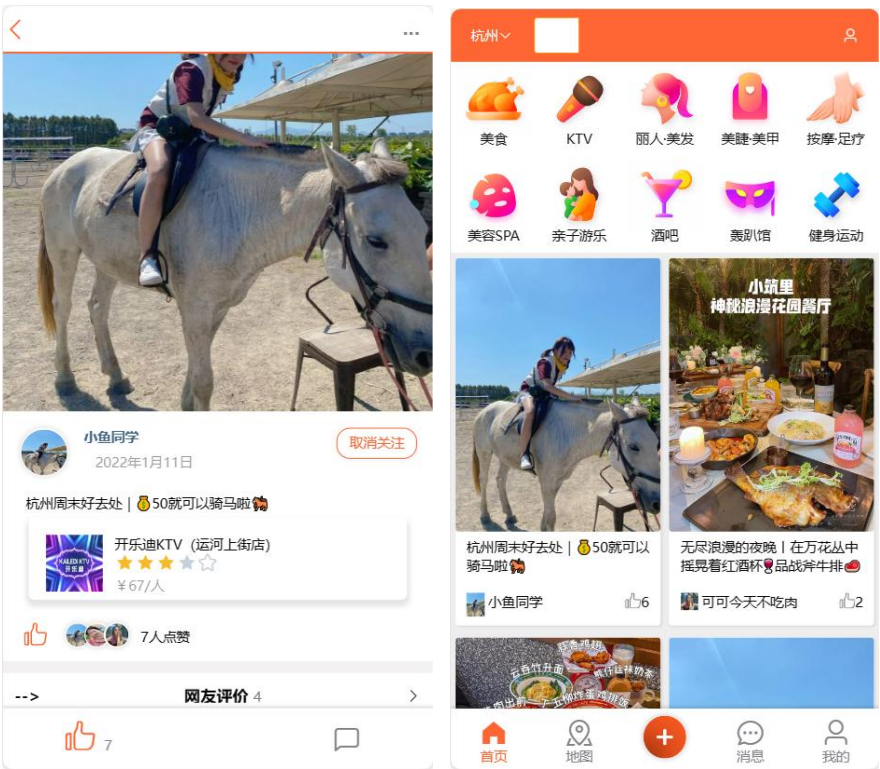
在主页上实现自动补全功能，同时提供点赞功能。



四、按类别为同一组，根据距离、评分、人气等进行排序。

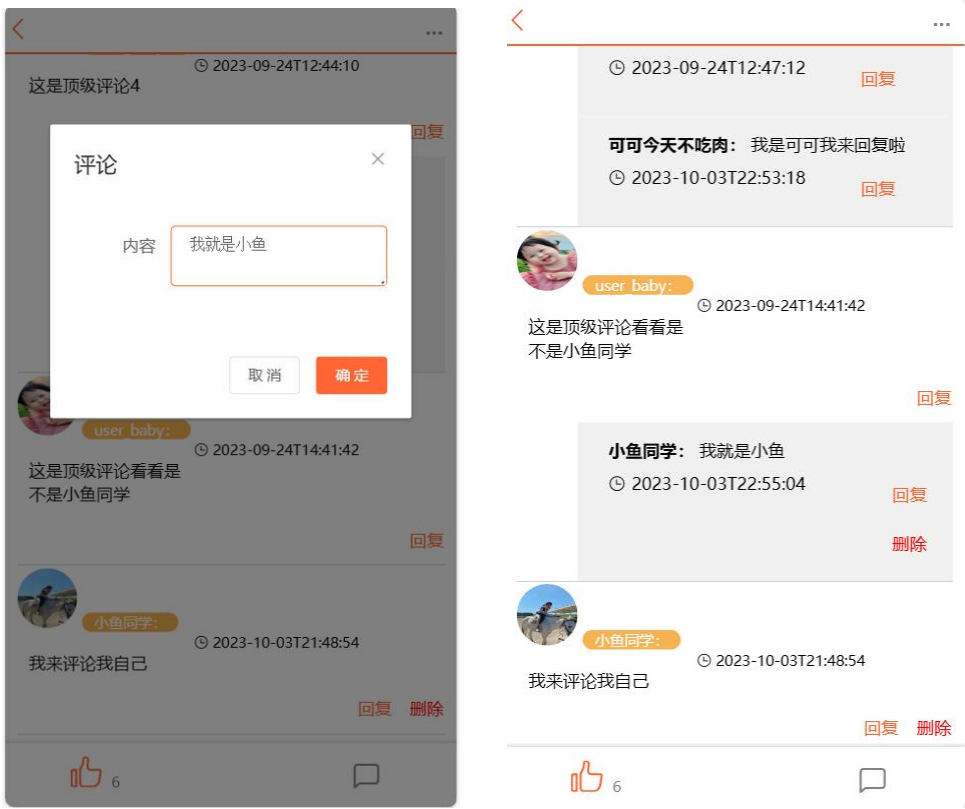


五、点赞及其排行榜



六、评论

评论分顶级评论和回复，当评论为回复他人评论的时候，呈下一级进行展示；
只有是当前用户的评论才显示删除按钮， 他人的评论不显示删除按钮。



七、发布博客及删除博客



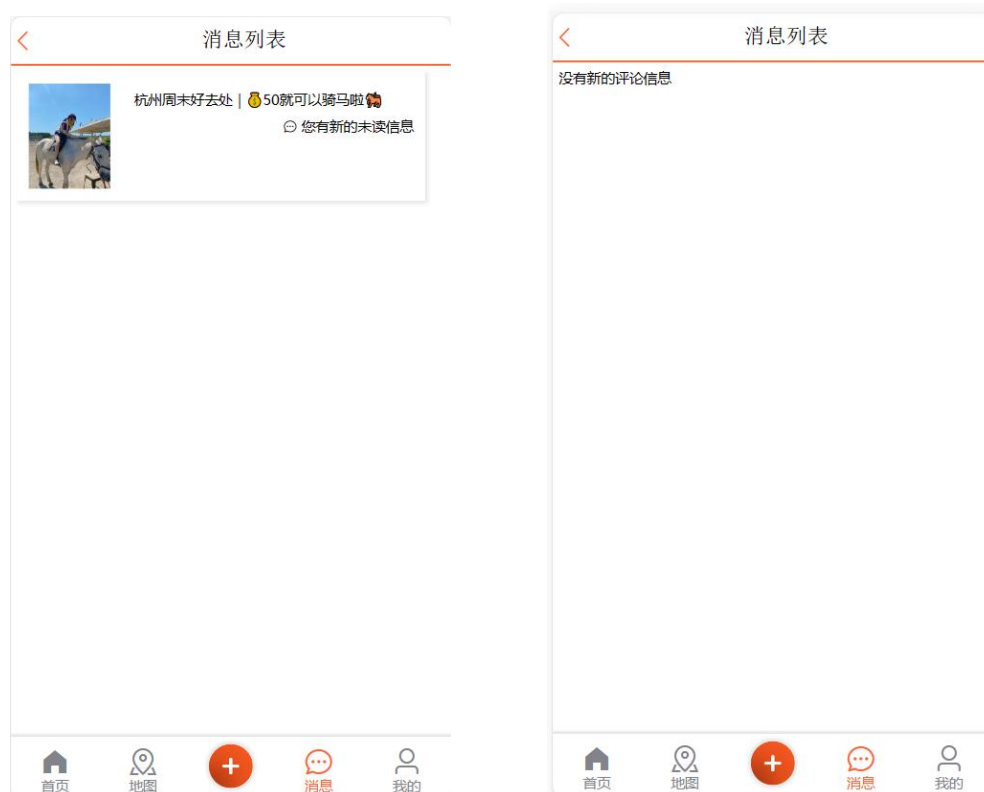


八、抢购优惠券



九、处理未读消息

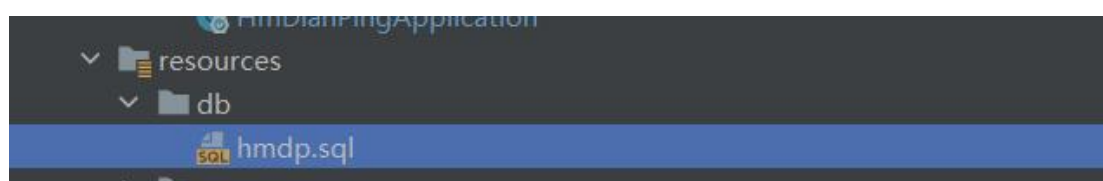
当“我的博客”出现新的评论的时候，则显示在该消息页上。当点击博客进入博客详情页后，就将该博客移除消息页面。

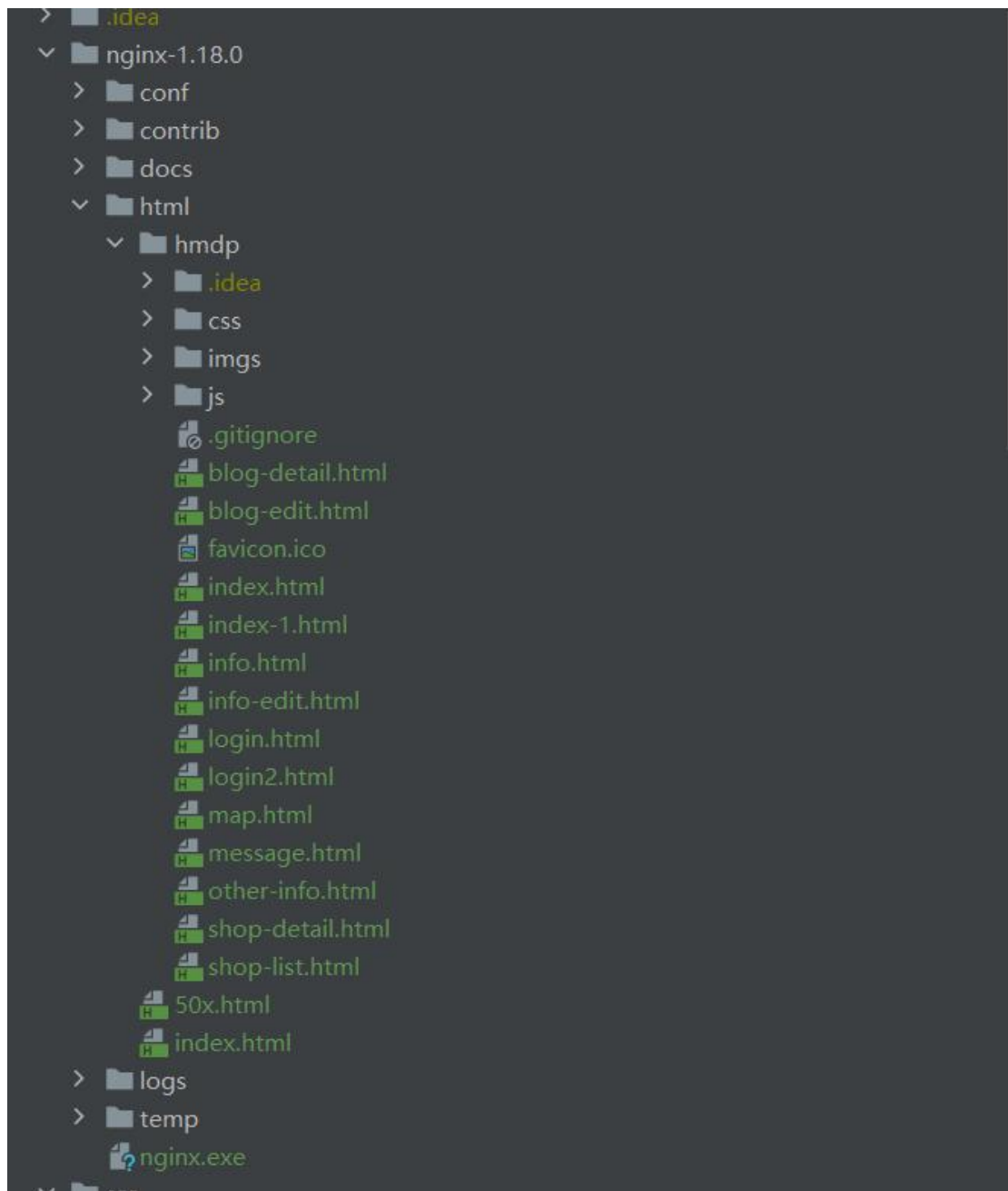


五、项目导入

将该项目用 idea 打开，首先导入后端代码，启动 Redis，运行该 SpringBoot 项目。

而后点击 resources 下的 SQL 文件，导入数据，再双击 nginx.exe，运行 Nginx，访问 <http://localhost:8080> 即可访问到主页。





实现功能前需要提前导入一些缓存数据：

以下是功能全实现中所包含的缓存数据，根据情况可自行先行导入。**Test** 目录下有部分记录，可根据代码进行调整。


```

@Test
void loadShopData() {
    // 1. 查询店铺信息
    List<Shop> list = shopService.list();
    // 2. 把店铺分组, 按照typeId分组, typeId一致的放到一个集合
    Map<Long, List<Shop>> map = list.stream()
        .collect(Collectors.groupingBy(Shop::getTypeId));
    // 3. 分批完成写入Redis
    for (Map.Entry<Long, List<Shop>> entry : map.entrySet()) {
        // 3.1. 获取类型id
        Long typeId = entry.getKey();
        String key = SHOP_GEO_KEY + typeId;
        // 3.2. 获取同类型的店铺的集合
        List<Shop> value = entry.getValue();
        List<RedisGeoCommands.GeoLocation<String>> locations = new ArrayList<>(value.size());
        // 3.3. 写入redis GEOADD key 经度 纬度 member
        for (Shop shop : value) {
            // stringRedisTemplate.opsForGeo().add(key, new Point(shop.getX(), shop.getY(), shop.getId().toString()))
            locations.add(new RedisGeoCommands.GeoLocation<>(
                shop.getId().toString(),
                new Point(shop.getX(), shop.getY())
            ));
        }
        stringRedisTemplate.opsForGeo().add(key, locations);
    }
}

```

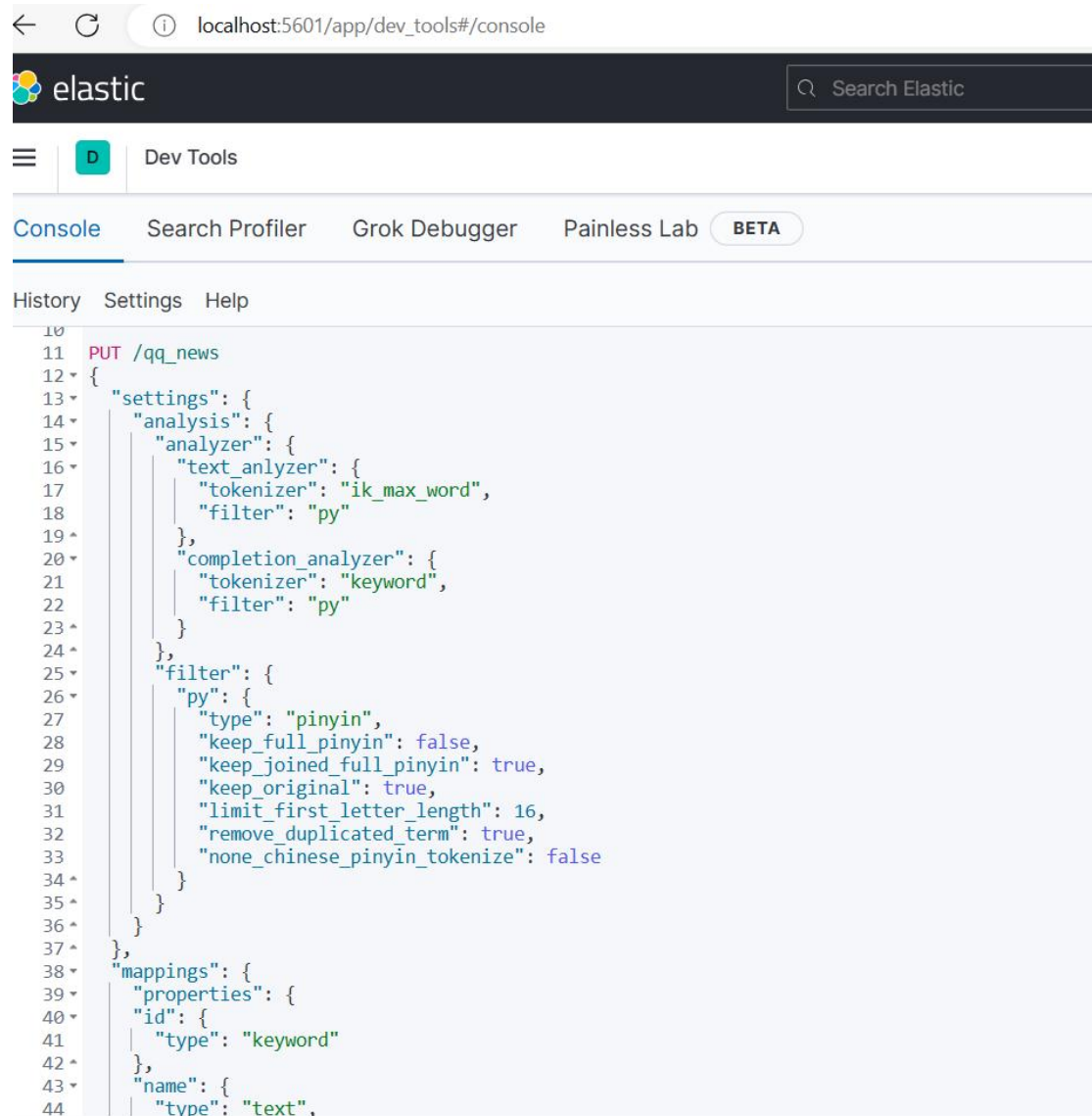
127.0.0.1

- db0 (95/95)
 - FILES_FRONT_ALL
 - blog (21)
 - liked (19)
 - of (2)
 - cache (4)
 - shop (3)
 - cache:shop:1
 - cache:shop:2
 - cache:shop:4
 - type (1)
 - feed (4)
 - followMine (1)
 - followYours (1)
 - follows (3)
 - icr (3)
 - order (3)
 - login (24)
 - token (24)
 - seckill (2)
 - order (1)
 - stock (1)
 - shop (6)
 - comments (2)
 - geo (2)
 - score (2)
 - stream.orders
 - sys_config (6)
 - sys_dict (10)
 - u_c_20230711
 - u_c_888888
 - user (6)

Elasticsearch 自动补全(拼音查询自动补全):



打开 DSL.txt 复制语句并运行。



```
10
11 PUT /qq_news
12 {
13   "settings": {
14     "analysis": {
15       "analyzer": {
16         "text_analyzer": {
17           "tokenizer": "ik_max_word",
18           "filter": "py"
19         },
20         "completion_analyzer": {
21           "tokenizer": "keyword",
22           "filter": "py"
23         }
24       },
25       "filter": {
26         "py": {
27           "type": "pinyin",
28           "keep_full_pinyin": false,
29           "keep_joined_full_pinyin": true,
30           "keep_original": true,
31           "limit_first_letter_length": 16,
32           "remove_duplicated_term": true,
33           "none_chinese_pinyin_tokenize": false
34         }
35       }
36     },
37   },
38   "mappings": {
39     "properties": {
40       "id": {
41         "type": "keyword"
42       },
43       "name": {
44         "type": "text",
```

而后运行 Test 文件夹里面的测试方法导入数据。

```
* 将所有数据存储到名为"qq_news"的索引中。
* @throws IOException
*/
vigeer
@Test
void testBulkRequest() throws IOException {
    // 查询所有的酒店数据
    List<Shop> list = shopService.list();

    // 1.准备Request
    BulkRequest request = new BulkRequest();
    // 2.准备参数
    for (Shop shop : list) {
        // 2.1.转为HotelDoc
        ShopDoc hotelDoc = new ShopDoc(shop);
        // 2.2.转json
        String json = JSON.toJSONString(hotelDoc);
        // 2.3.添加请求
        request.add(new IndexRequest("qq_news").id(shop.getId().toString()).source(json, XContentType.JSON));
    }

    // 3.发送请求
    client.bulk(request, RequestOptions.DEFAULT);
}
```

上传的地址:

```
public class SystemConstants {
    // public static final String IMAGE_UPLOAD_DIR = "D:\\heima_dianping\\nginx-1.18.0\\ht
    4 usages
    public static final String IMAGE_UPLOAD_DIR = System.getProperty("user.dir")+
        "\\nginx-1.18.0\\html\\hmdp\\imgs\\";
    2 usages
    public static final String USER_NICK_NAME_PREFIX = "user_";
    3 usages
    public static final int DEFAULT_PAGE_SIZE = 5;
    9 usages
    public static final int MAX_PAGE_SIZE = 10;
}
```

Resources:

```
files:
  upload:
    path: icons\
```

使用路径的位置有 FileController 和 UploadController:


```

FileController.java x UploadController.java x
37     @Resource
38     private UserMapper userMapper;
39
40     @PostMapping("/upload/{userId}")
41     public String upload(MultipartFile file, @PathVariable("userId") Long userId) {
42         String originalFilename = file.getOriginalFilename();
43         String type = FileUtil.extName(originalFilename);
44         // 定义一个文件唯一的标识码
45         String fileUUID = IdUtil.fastSimpleUUID() + StrUtil.DOT + type;
46         String fileUploadPath = this.getClass().getResource("/").getPath();
47         String fileUploadPath = System.getProperty("user.dir") + "\\nginx-1.18.0\\html\\hmdp" + fileUploadPath;
48         String filePath = SystemConstants.IMAGE_UPLOAD_DIR + fileUploadPath;
49         File uploadFile = new File(pathname: filePath + fileUUID);
50         // 检查目录是否存在，如不存在则创建
51         File parentFile = uploadFile.getParentFile();
52         if (!parentFile.exists()) {
53             parentFile.mkdirs();
54         }
55     }

```

```

FileController.java x UploadController.java x
20     @PostMapping("/blog")
21     public Result uploadImage(@RequestParam("file") MultipartFile image) {
22         try {
23             // 获取原始文件名称
24             String originalFilename = image.getOriginalFilename();
25             // 生成新文件名
26             String fileName = createNewFileName(originalFilename);
27             // 保存文件
28             image.transferTo(new File(SystemConstants.IMAGE_UPLOAD_DIR, fileName));
29             // 返回结果
30             Log.debug("文件上传成功, {}", fileName);
31             return Result.ok(fileName);
32         } catch (IOException e) {
33             throw new RuntimeException("文件上传失败", e);
34         }
35     }
36

```

六、小结

新手小白进一步对该学习项目功能进行完善，在学习 Redis 的相关知识点的同时，对前后端接口联调有了进一步的认识与实践；其他功能的实现也将查询数据库的操作优化为从 Redis 缓存中进行查询，进一步减少数据库的查询压力，实现数据库与 Redis 的数据一致。同时有很多功能思考并不全面，也有很多知识还未涉及，未来会努力的~。