【卓越软件开发】Project 项目文档

19302010015 许同樵

1、完成情况

1.1 情况概述

所有基本要求全部完成
Bonus 完成：详情页面局部放大、注册与登录验证码、高级模糊搜索、云部署

1.2 网站导航

未登录时导航栏显示 Home，Search，Login 三个选项，登陆时 Login 的位置变为用户的登录名，下拉菜单包含 Photo, Upload, Favourite, Friends, Log out 四个选项，分别可以跳转到对应页面。

1.3 登录注册

注册包含用户名，邮箱，密码，确认密码和验证码。用户名长度 4 至 15 位，包含 4 和 15。密码长度 6 至 12 位，包含 6 和 12。有密码强弱性检测及 UI 醒目显示。密码强弱性根据包含小写字母，大写字母，数字，下划线的种类数量确定。注册成功后变为已登录状态。
登录单独设计页面，包含用户名/邮箱，密码和验证码。若用户名/邮箱或密码错误会在下方显示“Wrong name or pass! ”。登陆后若可以跳转则转到登陆前界面，若不能则返回至首页。
登录和注册功能在前端向后端提交的过程中是用了 CryptoJS 的 aes 加密方式。

1.4 主页页面

即首页。轮播图展示 4 个最多收藏的图片，下方显示最晚上传的图片。
每个最新图片包含作者、主题和发布时间。

1.5 详情页面

显示图片的所有具体信息，包括标题、作者、内容、国家、城市、发布时间、收藏人数和详情介绍。
未登录时无法点击“like”按钮，登录后判断该用户是否已经收藏此图片，显示“like”或“dislike”，点击后执行并刷新页面。
可以进行局部放大。体验较好。

1.6 搜索页面

可选择根据标题或根据主题进行筛选，筛选结果可按时间排序或按热度排序。
实现分页，每页显示 6 张。
搜索实现模糊匹配，并实现了基本的高级模糊搜索。多个关键词可用空格隔开，结果会显示同时包含两个关键词的结果。

1.7 上传页面

选择本机图片上传，上传后显示在合适位置。需要填入标题、作者、主题、简介，选择国家、城市。提交前显示确认信息，数据库中的上传时间为此时的日期和时间，如果有未填入或选择的情况，则拒绝提交。

1.8 我的照片

每张图片有修改和删除两个按钮。
修改按钮可以到达修改界面，其页面事先显示原图片信息，图片不可重新上传，但是其他内容都可以进行改动。
删除按钮点击后提示“Are you sure？”，确认后删除。
实现分页

1.9 我的收藏

显示所有的 like 照片。有取消收藏按钮，点击后可以从收藏中移除
收藏下面显示 footprint，里面不重复地显示了刚刚浏览过的 10 张图片。如果图片浏览过的图片已经被删除，会显示对应的提示。
收藏上方显示“Other can see my favorite images.”的选择框，用户可以在此设置是否允许其他好友看到自己的收藏。

1.10 好友系统

Friends 页面进入好友系统。最上方显示所有好友。下方显示用户搜索系统。用户可以在此找到其他用户并发送好友请求。如果该用户收到请求，会在此页面提示，同意后两者互为好友。
如果已经互为好友且对方设置收藏可见，可点击好友信息下方按钮进入 ta 的收藏界面。如果设置不可见，则按钮不可点击。
每个好友显示名称、邮箱、注册时间和收藏界面按钮（可点击或不可点击）

2、项目架构

2.1 架构概况

本项目前端使用 bootstrap 简单实现，并使用 JavaScript（包括 JQuery）实现部分前端功能。后端使用 JSP+Servlet+JavaBean 的 MVC 架构模式。服务器使用 Mysql 实现。

项目分层情况

2.2 数据库结构

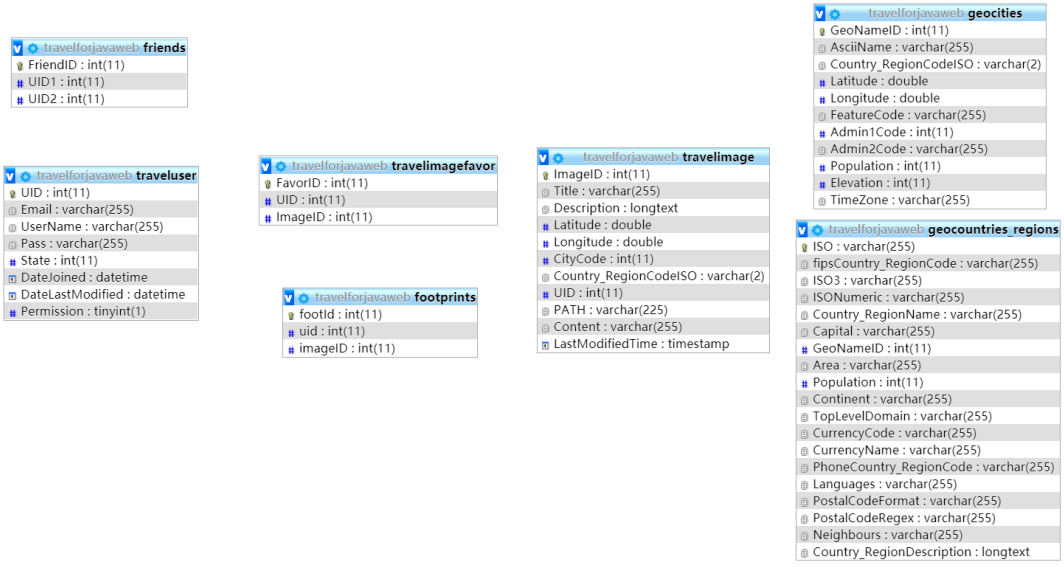
在原数据库的基础上增加的部分：

1、friends 数据表，用于实现好友功能

2、footprints 数据表，用于实现足迹功能

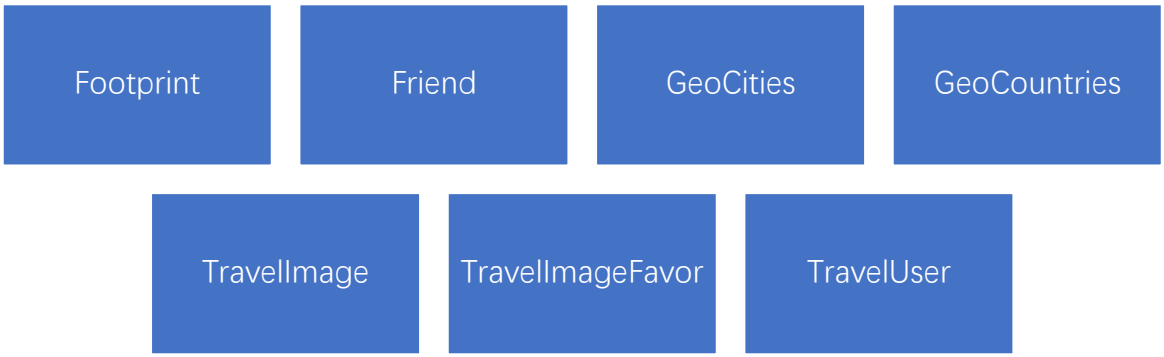
3、traveluser 数据表中新增 permission 列（boolean），用于判断该用户是否允许其他好友访问他的收藏。

4、travelImage 新增 LastModifiedTime 列，用于存储其更新时间。



2.3 实体类

为每一个数据表都各自创建了一个实体类，包含数据库中的每一个必要的私有属性，并创建 get，set 方法和 constructor（无参数和多参数）。



2.4 C3P0 连接池

用于创建和管理缓冲池，提高与数据库连接的性能。本项目中设置了数据库连接池参数：
一次性向数据库请求的连接对象：5
初始化连接数：10
最小连接数：10
最大连接数：50
数据源内加载的 PreparedStatements 数量：20
单个链接所拥有的最大缓存数为：5

2.5 JDBC 工具类

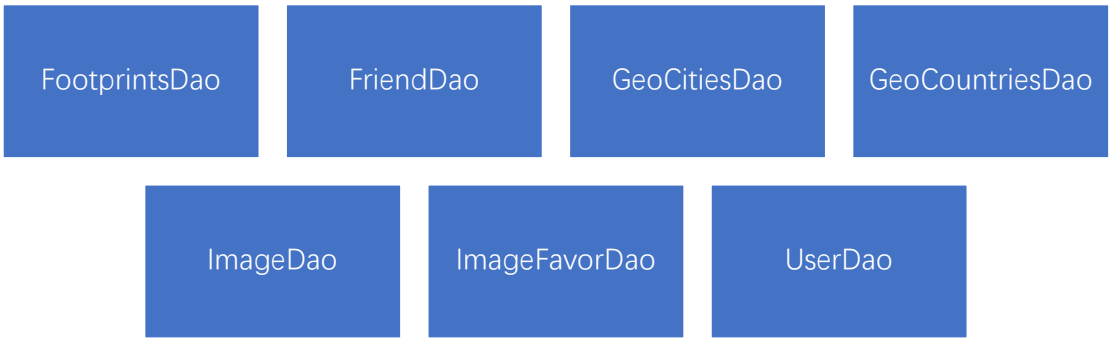
用于控制 c3p0 连接池的连接，包含 getConnection()，和 releaseConnection()两个方法。JDBC 工具类可以进一步降低连接池和 DAO 的耦合性，提高逻辑的健壮性。

2.6 DAO 父类

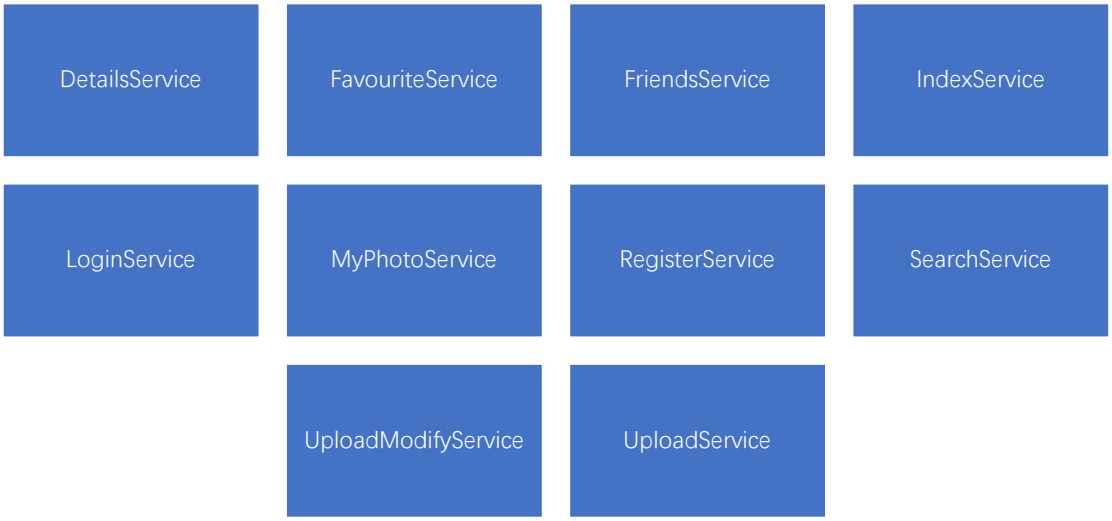
DAO 父类不实现任何 mysql 语句，而是使用泛型创建所有所需要的获取数据的方法，包含获取一个实体对象，获取多个实体对象，获取某个数值以及更新数据库的方法。

2.7 DAO 子类

为每一个实体对象分别创建了一个 DAO 子类，所有 mysql 语句的执行仅在这里实现。DAO 子类继承于 DAO 父类，因此可以使用 DAO 父类的所有方法，即 DAO 子类只需要关注 mysql 语句的逻辑实现即可。
每一个 DAO 子类都为接口，并另有具体的实现类。



2.8 Service 层

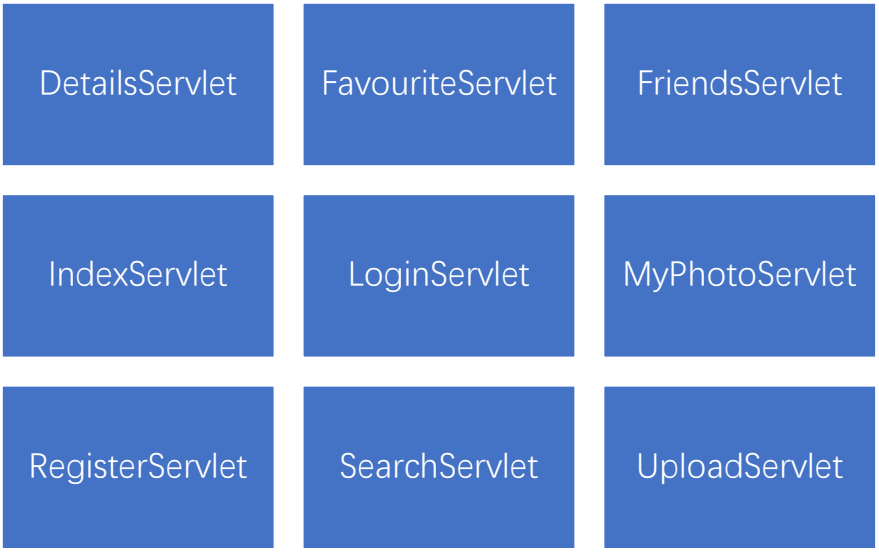


Service 层已经分离至每一个页面，每一个 Service 类都调用 Dao 方法实现相对应页面的业务。Service 层需要调用到 Dao 子类的时候，都会声明一个私有变量，例如：

```
private ImageDao imageDao = new ImageDaoImpl();
```

于是，Service 层不需要写任何 mysql 语句，而只需要关注业务的实现即可。

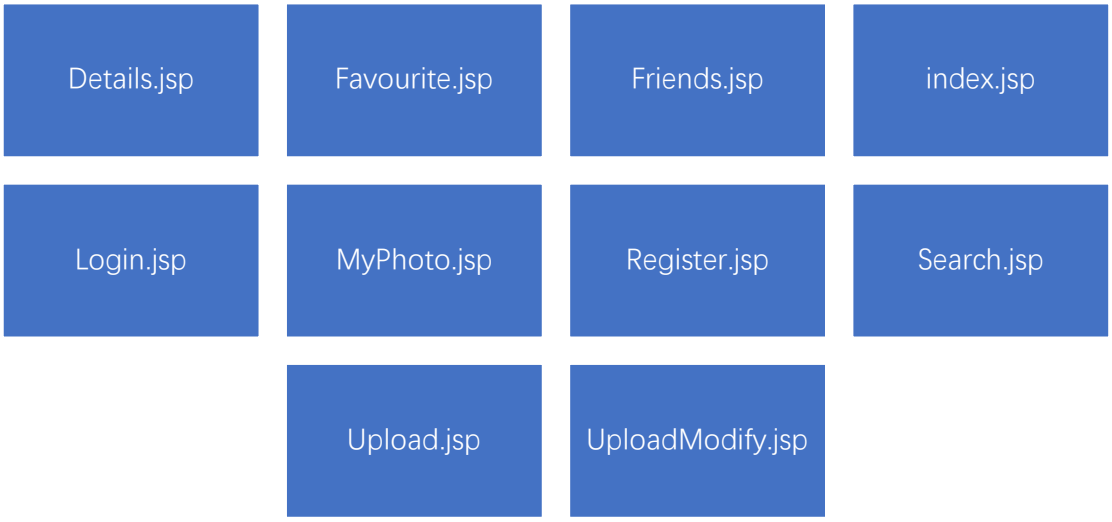
2.9 Servlet



每一个 Servlet 都是通过后缀名匹配访问的。Servlet 包括 post，get 方法和不同业务的实现方法。Servlet 不直接调用 Dao，而是创建对应的 Service 对象并调用其方法来获取数据。在具体访问中，servlet 是进行直接访问的对象。

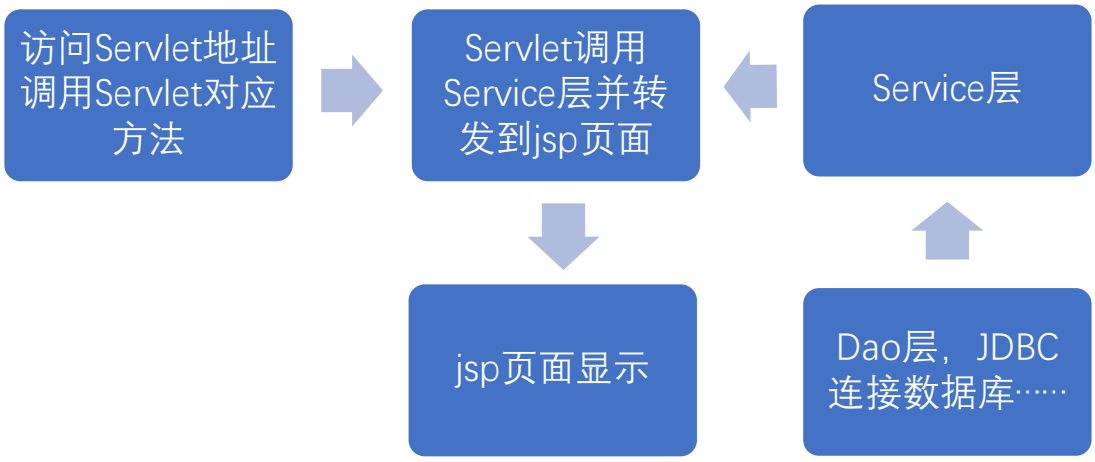
2.10 jsp 页面

Jsp 页面是最终向用户显示的页面，其不被直接访问，而是通过 Servlet 的转发方式访问，以预加载某些数据。

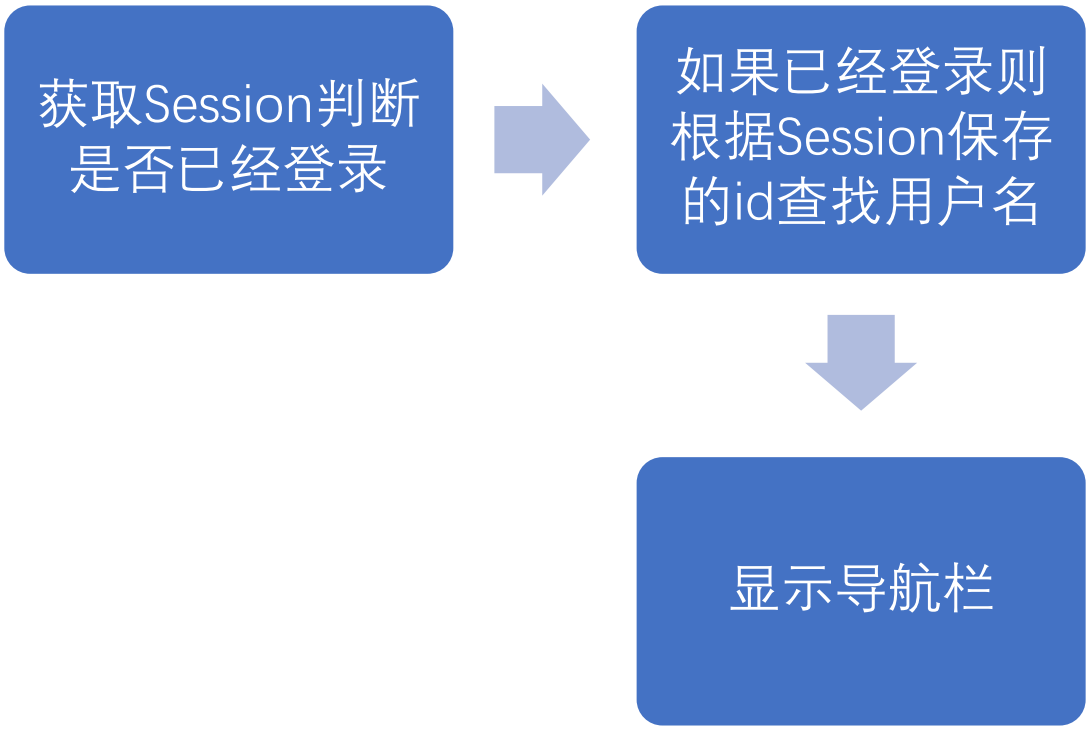


3、业务架构

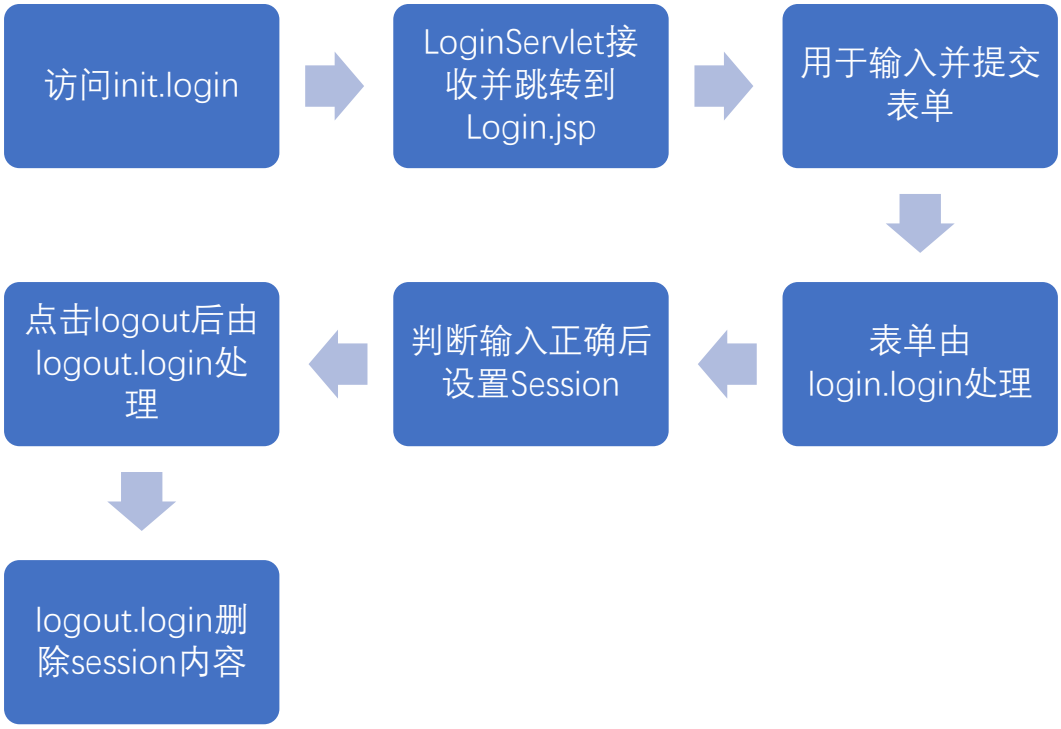
3.1 基本思路



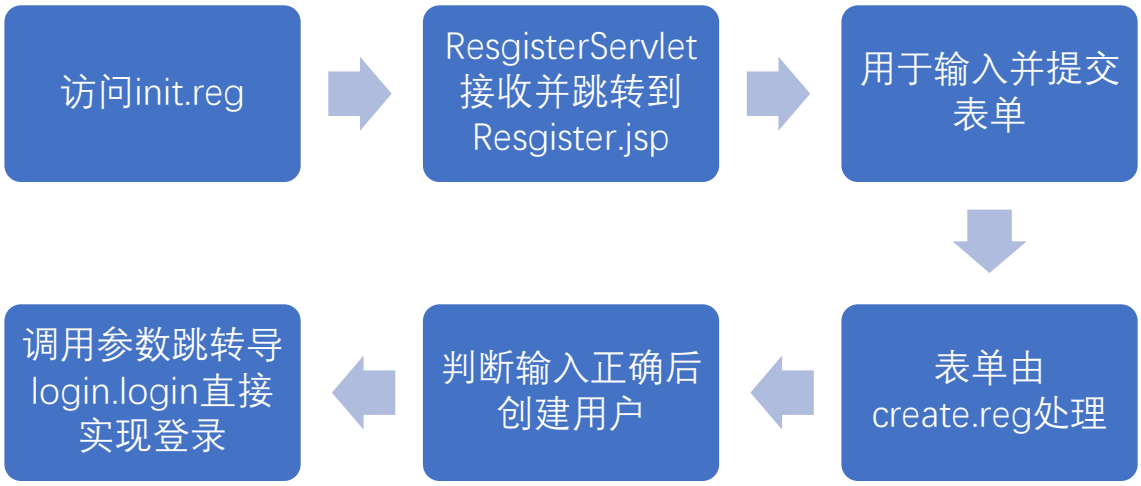
3.2 导航逻辑



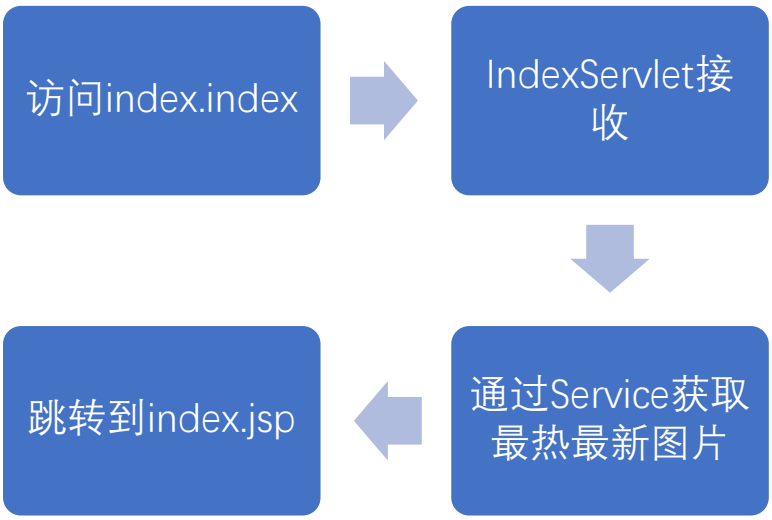
3.3 登录逻辑



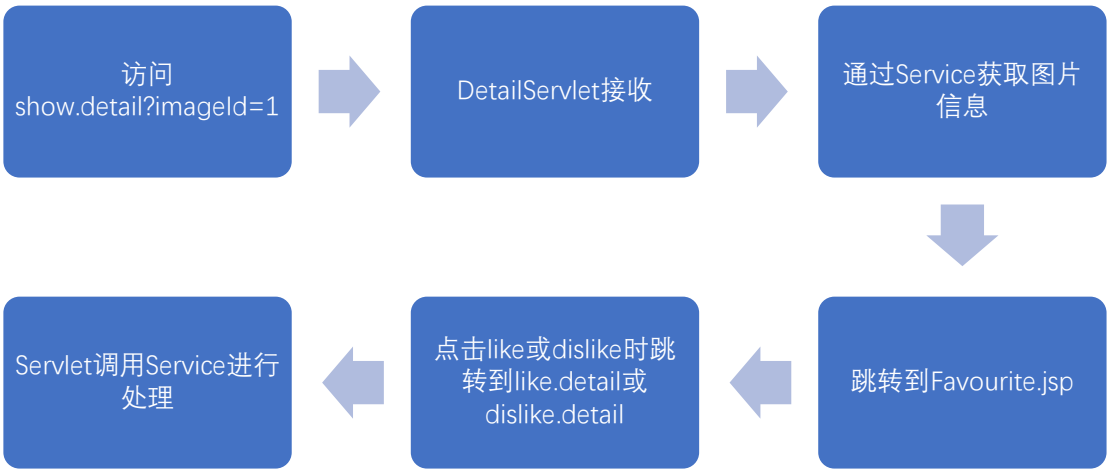
3.4 注册逻辑



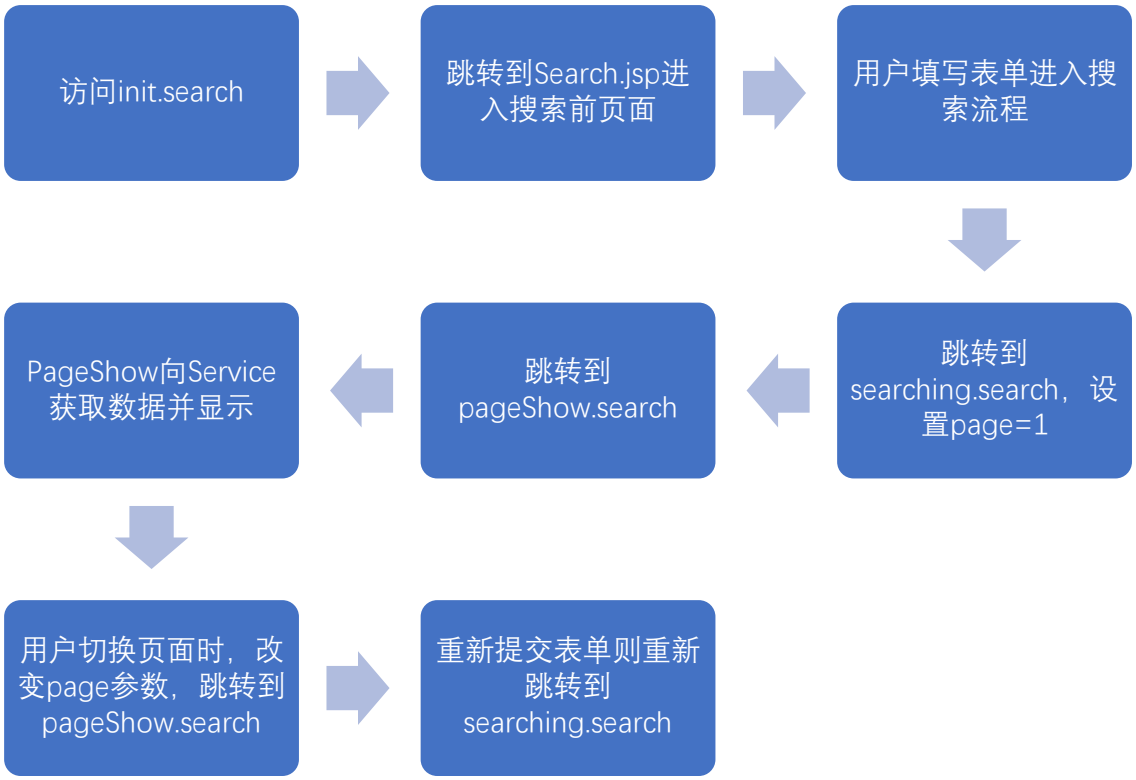
3.5 主页逻辑



3.6 详情逻辑

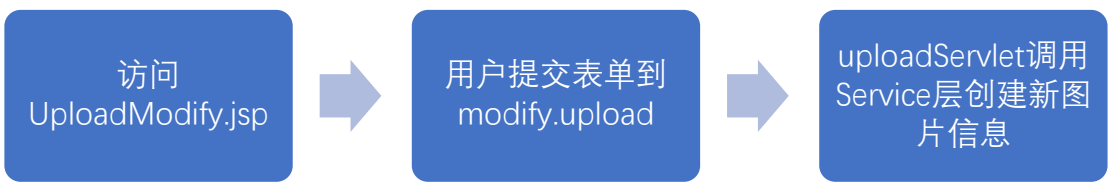


3.7 搜索逻辑

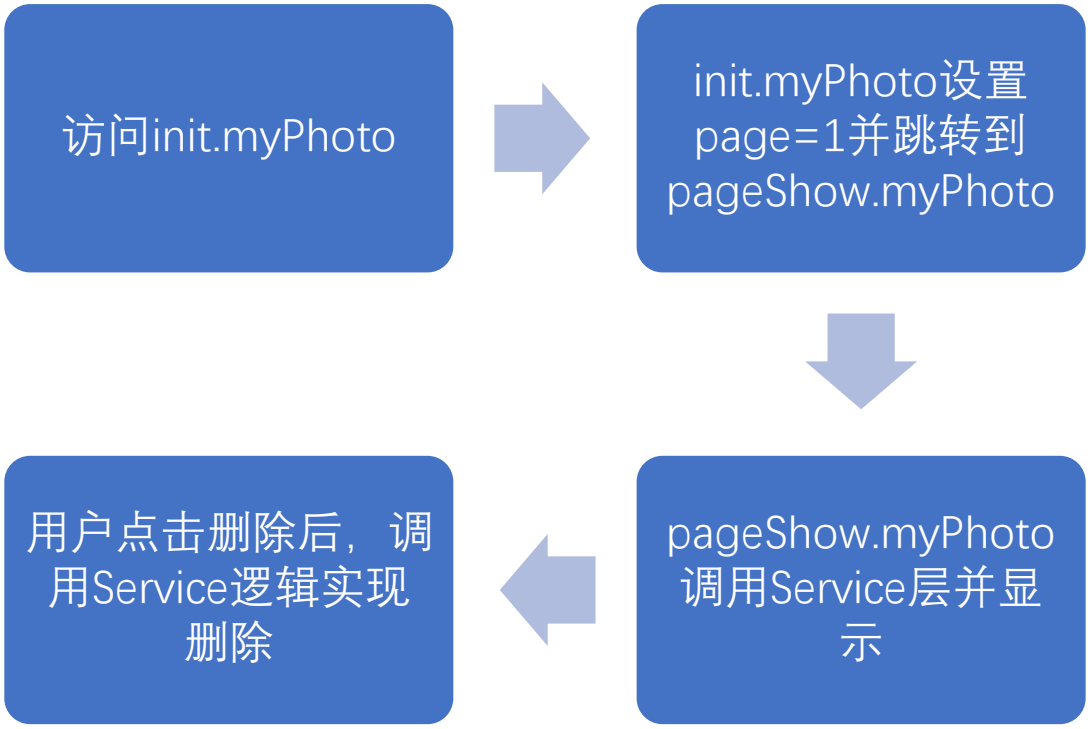


3.7 上传逻辑

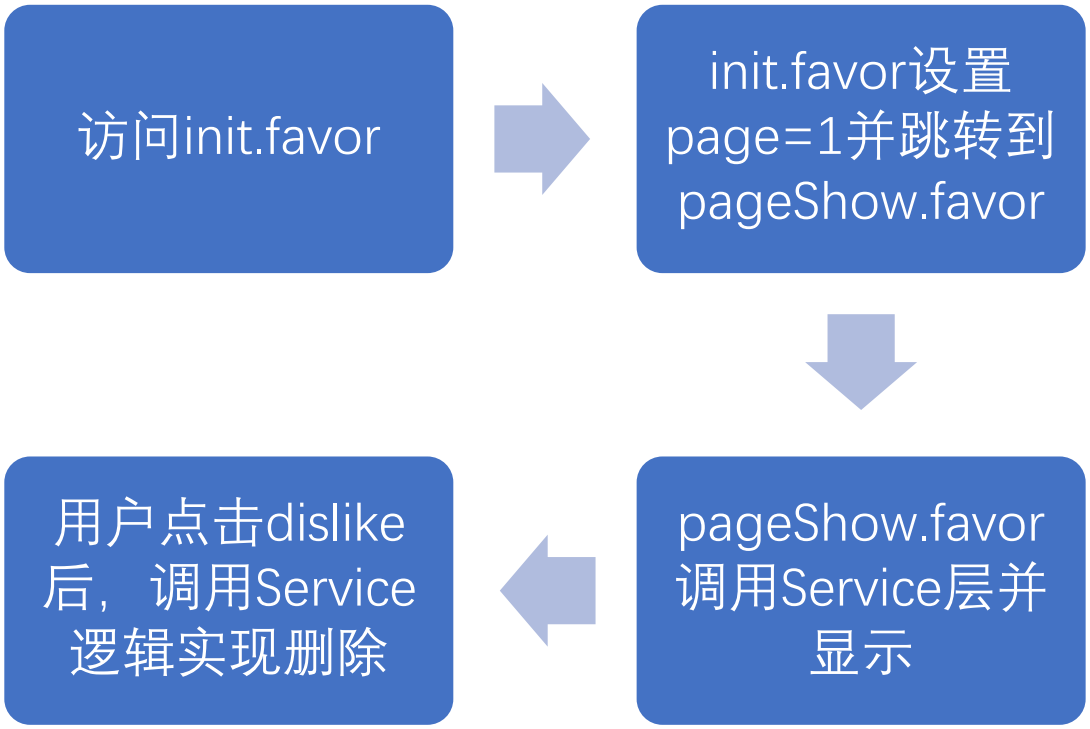
3.8 上传修改



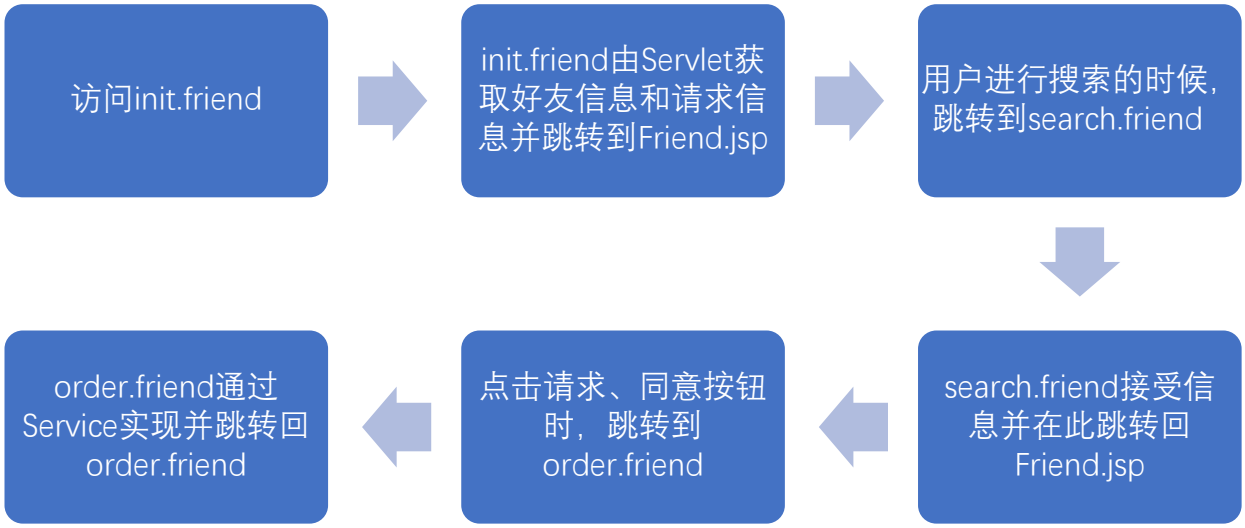
3.9 我的照片



3.10 我的收藏



3.11 好友系统



4、关键实现

4.1 获取图片收藏人数的排序

在 pj 项目中，首页显示、搜索结果常用到获取图片收藏人数排序的情况。在此不仅需要获取每一个图片的收藏人数数据，还需要将其从大到小进行排列。最终，project 使用了**两种方法**，分别实现了首页显示最热图片和搜索排列的功能。

4.1.1 Service 层 实现收藏人数排序

在首页显示最热图片中使用了 Service 层实现收藏人数的排序。方法如下：

1）从 Dao 中获取 ImageFavor 的所有数据

2）创建一个 map 进行计数，如果第一次出现的图片创建新的键值对，否则其键对应的值++

3）用 Sort 对 map 进行排序

4）输出前四个值并查找图片信息，将 4 个 TravelImage 对象输出给前端

4.1.2 使用 mysql 创建临时表 实现收藏人数排序

在搜索页面的排列直接使用了 mysql 语句的方式进行排序。语句在 ImageDao 中执行，语句为：

*CREATE TEMPORARY TABLE likes AS SELECT *, count(*) as likePeople*

FROM travelImageFavor GROUP BY ImageID;

SELECT travelImage.imageId, title(等其他图片信息…) FROM (travelImage LEFT JOIN likes ON likes.ImageID = travelImage.ImageID) WHERE content LIKE '%a%' ORDER BY likePeople DESC LIMIT 0 , 6 ;

DROP TABLE likes;

下面对语句进行具体解释：
1）先创建一个名字为 likes 的临时表（TEMPORARY TABLE），以 ImageID 将 ImageFavor 表进行归类并计数（相当于 4.1.1 中 Service 层使用 map 的操作）
2）使用 likes 数据连接到 travelImage 表充当左表，连接方式就是两者的 ImageID 相同，这样可以做到的是所有的收藏数为 0 的图片也存在于表中。
3）这时需要注意的是，表中有两列为 ImageID，所以需要标明是提取那一列的 ImageID（虽然两个数据一样，但是还是要标明，例如这里写的是 travelImage.imageId）。
4）WHERE 后写模糊搜索内容，注意用正则表达式形式
5）收藏人数从大到小排列，因此加 DESC
6）分页逻辑顺便一起实现了，例如这里是第一页的分页逻辑
7）虽然创建的是临时表，但是为了保险起见还是执行一行删除，防止 connection 未断开的情况出现导致数据错误。（如果不及时删除，可能会出现“已经存在此表“的报错）
8）Service 层不需要额外实现需求，直接调用获取 TravelImage 对象就可以了。

4.1.3 利弊分析

在使用两个方法的时候，可以发现两者各有利弊。

使用 Service 层用 java 代码处理可以让逻辑更为清晰（即 mysql 语句只执行最简单的语句，Service 层完成具体的需求），且实现起来更为简单，但问题在于在 like 的数据量过大的时候，数据库要向 Dao 传送过多的数据，可能造成层与层交流的压力过大。

使用 Mysql 语句相比来说数据库只需要向 Dao 传送六个图片的数据，传送量小，但由于数据处理在 mysql 实现，必定会增加数据库的处理负担。以及临时表的创建需要多行代码，其中包括多种指令（包括增加表，获取数据），因此 Dao 父类中简单的方法并不能直接调用，需要专门为这个问题写一个方法，可能在某种程度上会破坏方法的通用性。

4.2 好友系统的实现

好友系统包括好友的添加，同意添加好友，好友的获取三个功能。本 Project 由于体量并不大，考虑到过多的数据库没有太大的必要，因此只使用了 friend 一个数据库就实现了好友系统。

friends 数据表包括主键 id，Uid1，Uid2 三列。主键 id 用于记录数据的条目，Uid1 被视作是己方，Uid2 被视作是对方。对“朋友“一词的定义是：不仅 A 想成为 B 的朋友，B 也想（至少同意）成为 A 的朋友。因此只有同时存在两列，分别是 Uid1=A Uid2=B 和 Uid1=B Uid2=A，才可算作互为好友。如果只存在一个，那么被视为 Uid1 向 Uid2 发送请求的状态。具体的数据库操作的实现如下：

1）好友发送请求时添加一行 Uid1=当前用户 id Uid2=请求好友的 id

2）访问 friend 页面时获取 Uid2 为自己，Uid1 为另一个用户的数据，如果不存在 Uid1 为自己 Uid2 为他的行，则说明这是一条请求，在 好友请求 一栏中显示。

3）如果存在一对数据满足互为好友的条件，则显示在 我的好友 一栏中。

4.3 表单操作与验证码

本 pj 中的表单验证，密码强度显示，前端加密和验证码都是用 JavaScript 来实现的。

4.3.1 表单验证

4.3.2 密码加密

前后端交换数据的时候密码如果明文传递，则常常会使密码暴露，安全性差。因此本 pj 使用了 aes 加密的方法对密码进行加密传输。Aes 加密是可逆的，且 java 和 js 都有实现方法。这里调用了 CryptoJS 的 aes 加密库进行加密，在 java 和 js 设置了相同的密钥，确保加密解密后的密码不变。

4.3.3 验证码

本 pj 实现了 bonus 项的验证码功能，每次表单加密都需要确认验证码是否正确才会进行提交。验证码的基本思路是用 canvas 进行绘图，添加干扰项并显示在 input 元素后。具体绘图实现如下：

1）创建一个 codes 数组，里面存放 0-9 和 a-z 的字符。

2）进入一个 for 循环，i 从 0 到 3，随机取数组中的一个字符。

3）确定文字的坐标，x 为 10+20*i，确保所有文字不会重合；y 为 20-28 的随机数字，使得坐标可以稍微浮动但又不超出验证码框。

4）使得文字可以略微旋转，产生一个-0.5 到+0.5 的弧度。

5）由于只能将已经输入的内容旋转，因此旋转所有 canvas 绘图环境一个角度，添加文字，再将 canvas 绘图转回原来的角度。

6）循环四次后已经可以显示 4 个字符了，现在循环添加 7 条线和 50 个点，位置为 canvas 绘图区域的随机位置。

7）所有元素的颜色为随机颜色，rgb 都从 50-255 浮动，防止过浅看不见。

8）4 个字符被存到一个数组中，验证时只要验证数组连起来与输入内容是否相符即可。

8）点击 canvas 后重新初始化验证码即可。

4.4 高级搜索

本 pj 实现的高级搜索是：关键词可以用空格隔开，搜索会搜索包含所有关键词的结果。实现方法：

1）将整个输入内容切分空格存入 List。

2）删除空内容（防止输入多个空格的情况），如果最后 list 为空则直接返回空 list 即可，表示未进行输入。

3）mysql 语句在原来的语句中加入一个循环，每次获取到一个关键词则在第一个 WHERE 语句后加入新的条件（例如：AND title LIKE '%park%'）。

4）分页返回结果。

4.5 详情页面放大

本 Pj 实现了详情页面的放大功能，使用 JavaScript 进行实现。具体操作如下：

1）添加一个默认不可见的 div 框 magnifier，z 设置为 9999 使其在最上方，position: absolute。

2）鼠标移入 img 时，magnifier 显示，鼠标移出时隐藏，overflow: hidden。

3）magnifier 中插入了一张图片，地址与详情页的原图片相同，大小也相同。

4）鼠标移入图片内时不断获取其相对于图片的坐标，设置 magnifier 内图片的 margin 的左边和上边为鼠标 x、y 坐标的负值

5）如果鼠标坐标在边缘时条件判断使其不会超出图片边缘

6）将 magnifier 的位置相对鼠标静止，并用 scale（2）将其扩大两倍

用这种扩大 div 的方式可以减少内部图片放大所需要的运算。

4.5 服务器部署及遇到的问题

本 pj 已经部署于服务器中。

地址：http://121.89.211.169:8080/

使用了阿里云 ECS 服务器，并安装了宝塔面板进行管理。这里列举一些服务器部署中遇到的问题。

4.5.1 javaWeb 打包

使用 Tomcat 则需要先把 javaWeb 打包成 war 格式。这项操作可以由 Idea 进行完成。但是 Intellij Idea 需要先在项目设置中在 artifact 中设置打包的方式，需要注意的是所有包都要打包进去。初次打包的时候忘记将 lib 库打包，导致网站无法正常运行。

4.5.2 Tomcat 存放 JavaWeb 的位置

在网上查到使用 Tomcat 一般将 war 文件放入 webapp 文件夹中即可，但是 pj 部署的时候发现并没有作用。

通过查找 Tomcat 的配置文件可以发现，war 文件的默认存放位置是宝塔面板专门用来存放网站的文件夹。可能是宝塔面板将 Tomcat 的默认位置修改过了。最终把存放位置改回了 webapp 即解决问题。

4.5.3 mysql 数据库不存在该数据表的错误

在运行网站的时候发现一旦连接数据库就会出现问题。报错如下：

```
    at com.project.web.servlet.IndexServlet.index(IndexServlet.java:33)
    ... 28 more
java.sql.SQLException: Table 'travelForJavaWeb.travelImageFavor' doesn't exist Query: SELECT FavorID,UID,ImageID FROM travelImageFavor Parameters: []
    at org.apache.commons.dbutils.AbstractQueryRunner.rethrow(AbstractQueryRunner.java:527)
    at org.apache.commons.dbutils.QueryRunner.query(QueryRunner.java:391)
    at org.apache.commons.dbutils.QueryRunner.query(QueryRunner.java:252)
    at com.project.dao.DAO.getForList(DAO.java:71)
    at com.project.dao.impl.ImageFavorDaoImpl.getAll(ImageFavorDaoImpl.java:15)
    at com.project.service.impl.IndexServiceImpl.popularPhotos(IndexServiceImpl.java:19)
```

图中可以看出其实已经连接上 mysql 数据库了，就是无法找到数据表。

后来通过排查发现，本地的 mysql 数据库设置了大小写不敏感，因此数据表全部被改成小写了。但服务器默认了大小写敏感，因此无法访问。最后规范了 Dao 层的语句和数据表名称，即解决问题。