# 目 录

1. [绪 论 1](#_bookmark0)
   1. [研究的背景及意义 1](#_bookmark1)
      1. [选题的背景 1](#_bookmark2)
      2. [国内外研究现状 1](#_bookmark3)
      3. [研究的意义 2](#_bookmark4)
   2. [系统目标 2](#_bookmark5)
2. [需求分析 4](#_bookmark6)
   1. [业务需求 4](#_bookmark7)
   2. [功能需求 8](#_bookmark8)
      1. [角色分析 8](#_bookmark9)
      2. [功能需求 8](#_bookmark10)
   3. [非功能需求 16](#_bookmark11)
      1. [环境需求 16](#_bookmark12)
      2. [性能需求 16](#_bookmark13)
      3. [安全需求 17](#_bookmark14)
3. [总体设计 18](#_bookmark15)
   1. [系统设计的原则 18](#_bookmark16)
   2. [系统体系结构设计 18](#_bookmark17)
   3. [系统功能模块设计 21](#_bookmark18)
4. [数据库设计 23](#_bookmark19)
   1. [概念结构设计 23](#_bookmark20)
      1. [设计思路 23](#_bookmark21)

[4.1.2 E-R 图 23](#_bookmark22)

* 1. [逻辑结构设计 33](#_bookmark23)
     1. [设计思路 33](#_bookmark24)
     2. [逻辑模型 33](#_bookmark25)
  2. [物理结构设计 35](#_bookmark26)
     1. [存取方式 35](#_bookmark27)
     2. [存储结构 35](#_bookmark28)

1. [界面设计 44](#_bookmark29)
   1. [界面关系图 44](#_bookmark30)
   2. [界面设计成果 44](#_bookmark31)
2. [详细设计 49](#_bookmark32)
   1. [系统主要功能模块介绍 49](#_bookmark33)
   2. [用户管理 49](#_bookmark34)
      1. [功能结构设计 49](#_bookmark35)
      2. [类图设计 50](#_bookmark36)
   3. [装修交流平台管理 51](#_bookmark37)
      1. [功能结构设计 51](#_bookmark38)
      2. [类图设计 52](#_bookmark39)
      3. [时序图设计 53](#_bookmark40)
   4. [装修招标管理 54](#_bookmark41)
      1. [功能结构设计 54](#_bookmark42)
      2. [类图设计 54](#_bookmark43)
      3. [时序图设计 55](#_bookmark44)
   5. [装修公司信息管理 56](#_bookmark45)
      1. [功能结构设计 56](#_bookmark46)
      2. [类图设计 56](#_bookmark47)
      3. [时序图设计 57](#_bookmark48)
   6. [设计师风采展示管理 58](#_bookmark49)
      1. [功能结构设计 58](#_bookmark50)
      2. [类图设计 58](#_bookmark51)
      3. [时序图设计 59](#_bookmark52)
   7. [装修服务信息管理 60](#_bookmark53)
      1. [功能结构设计 60](#_bookmark54)
      2. [类图设计 60](#_bookmark55)
      3. [时序图设计 61](#_bookmark56)

[**7** 编码 62](#_bookmark57)

* 1. [代码实现与核心算法 62](#_bookmark58)
  2. [代码优化分析 65](#_bookmark59)

[**8** 测试 68](#_bookmark60)

* 1. [测试方案设计 68](#_bookmark61)
     1. [测试策略 68](#_bookmark62)
     2. [测试进度安排 68](#_bookmark63)
     3. [测试资源 68](#_bookmark64)
     4. [关键测试点 69](#_bookmark65)
  2. [测试用例构建 69](#_bookmark66)
     1. [测试用例编写约定 69](#_bookmark67)
     2. [测试用例设计 69](#_bookmark68)
     3. [关键测试用例 69](#_bookmark69)
     4. [测试用例维护 71](#_bookmark70)

1. [总结与展望 72](#_bookmark71)
   1. [设计工作总结 72](#_bookmark72)
   2. [未来工作展望 73](#_bookmark73)

[参考文献 75](#_bookmark74)

[附录 **A** 外文翻译—原文部分 77](#_bookmark75)

[附录 **B** 外文翻译—译文部分 85](#_bookmark76)

[附录 **C** 软件使用说明书 92](#_bookmark77)

[附录 **D** 主要源代码 96](#_bookmark78)

# 1 绪 论

## 研究的背景及意义

* + 1. 选题的背景

在新时代下网络信息化的快速发展，网络在生活中大面积地覆盖，传统装修服务及相应的装修公司受到了较大的冲击，一些传统的装修公司为迎接新时代下飞速发展的互联网时代的到来，都开始致力于线上交易以及线上服务。线下生活的在线服务也渐渐进入了所有人的生活里，人们对住房装修的要求也越来越高，一方面要求更加优质的装修服务，另一方面要求更加全面适用的装修信息提供和更加公开的装修家居交流平台。伴随林林总总的互联网模式逐渐搭建，网络使用户享有更好的体验，既满足了用户各方面的需求，而且也为企业带来了机遇。

近些年，O2O（Online To Offline）模式的逐步发展，便迅速进入到了人们生活的各方各面，互联网下的各种生态模式错综复杂，人人都能在线上找到适合自己的生存方式。O2O 模式的兴起，让线下传统行业找到了在互联网环境下生存的法宝。就目前的情况来看，整个社会全行业都在融入电子商务，而对于装修行业而言，O2O 模式必将是当下合适的途径。O2O 模式充分发挥出了传统实体店和网上电子商务的双重优势，实体店是线下传统行业的独有优势，电子商务是线下传统行业的必经之路，要更好地运用 O2O 模式，就必须将二者有效地结合。

将来，随着全国人民生活水平的不断提高，人们对装修服务的需求将迅速增长。从整个建筑行业的比重来看，装修行业的总产值将不断上升。在互联网消费快速发展的大环境下，各大装修公司及企业已经开始努力变更营销策略，让自己能够在互联网的浪潮中能够乘风破浪，立足根基。

在如今的互联网+形势下，传统的装修企业要拥有更加强大的竞争力，必然需要有线上服务交易的平台的支撑，网上装修服务显得尤为重要。然而，目前，整个装修行业由于信息及其不对称，很多业主不能在网上及时获取真实适用的有效信息，因此在房屋装修过程中吃过不少亏，走过许多弯路。另外，装修行业具有行业大、企业小、离散性高等特点，各装修公司乱象丛生，目前还没有出现规模化和专业化的大型企业和公司， 整个装修行业缺乏一套统一的服务执行标准，因而容易导致产业混乱。另一方面是价格的不透明化，容易出现装修公司乱开价糊弄业主的情况，业主的合作和消费权益得不到更好的保障。因此，构建一个公开透明、标准统一、信息对称全面的装修家居服务平台就显得如此重要，将装修公司汇集与一体进行标准统一化管理，为用户提供及时丰富的装修信息和优质透明的装修服务。

* + 1. 国内外研究现状

目前，国内的装修市场仍然是及其不透明的，很多的装修公司在与客户交易合作时

有很大的“可操作空间”，这跟装修行业的信息不对称息息相关。这种现状就导致消费者在选择装修服务时遇到重重困难，即便业主在选择装修服务前查阅了大量有关装修的知识和相关资讯信息，装修服务供应商仍然会在里面动手脚，例如低价揽客、增项获利、换劣质材料赚取利益，甚至做完不管，这就使得整个市场缺少可信度。

由于装修行业市场环境过于庞大，在这种环境下，客户的价值观不利于体现，比如： 如果上游产业受政策影响大量缩水，这将导致下游的装修企业竞争客户资源的环境变得恶劣，装修企业便无法没有精力去培养自身的服务理念，再加上企业之间费尽心思争夺人力资源，将会造成工人身价的不断上涨，这导致工人的职业道德培养堪忧。其次，装修行业的门槛较低，这将导致有些企业或者公司的资质较低，水平不够，服务不够专业。

对于国外，如美国，目前美国市场网络装修的主流是搭建平台将设计师、用户、装修公司直接对接，小金额家居装修改造需求相当强烈。

在国内外的市面上或多或少得有一些装修家居有关的网站，提供了基本的信息资讯和预约装修等服务，但功能和信息不是很齐全，操作性不是很强，基本都是一些企业的公司私人网站，很多信息都不能及时查看和更新，数据分散，参差不齐。其次，由于只有个别网站提供了基本服务，对于对比选择的操作体验并不会那么强，不能满足用户对比参考的需求。再者，由于部分公司拥有自己的平台系统，各个公司都有自己的一套标准，规则不明确、不统一的问题仍然不能得到解决。

* + 1. 研究的意义

本平台的设计，既秉承了节约用户的各方面成本、为用户提供及时丰富的装修信息的理念，又给用户提供了良好的用户体验。平台通过汇集各大装修公司和企业于一体， 为用户提供一套统一规范标准，公开透明的装修过程监控。平台为用户提供多方面的装修攻略和资讯，让用户更加了解目前的装修行情。公开的交流、反馈、投诉曝光平台， 让信息更加公开和真实。

由于适合如今的快速生活以及信息网络汇集的多方面应用，满足人们对住房装修的要求，装修家居服务平台也快速发展起来。建立汇集各大装修公司、为用户提供丰富全面的装修信息、提供公开透明的装修交流的装修服务平台，更加重视为公司提供统一的执行操作标准和客户来源，更加重视客户的交流反馈以及信息的透明公开，优化了用户的操作体验，便捷公司和用户之间的合作交流，加强线上线下的联动，塑造企业品牌的价值。在互联网+时代下，将线下的传统装修实体店与网上的专修服务平台的相结合， 更加利于装修企业的发展和创造，也节约了用户的时间，使得装修合作更加得高效和透明，有利于推动整个装修行业的发展。

## 系统目标

平台通过汇集各大装修公司和企业于一体，为用户提供一套统一规范标准，公开透

明的装修过程监控。平台为用户提供多方面的装修攻略和资讯，让用户更加了解目前的装修行情。公开的交流、反馈、投诉曝光平台，让信息更加公开和真实。装修家居服务平台能够实现用户管理、装修公司入驻管理、装修公司信息管理；能够实现装修服务信息管理，为用户提供多方面的装修相关信息和资讯；能够实现装修招标管理、装修交流平台管理，为客户提供更加开放公开的操作体验，解答装修疑惑和发布投诉曝光；能够实现设计师风采展示管理，便于用户自主选择设计师。另外，还能给装修公司提供统一的公司门户，能够实现装修工地展示管理，让业主及时了解各阶段工地的状况；能够实现设计师预约管理，用户选择设计师预约，设计师进行预约受理；能够实现业主装修成果反馈管理，体现装修服务的透明化，便于用户进行交流反馈。

本平台建设包括四个客户端，包括平台管理员端、公司负责人端、设计师端和平台用户端：

1. 平台管理员端是平台管理员登录系统，可以进行公司入驻审批管理、公司基本信息的管理、装修服务信息（包括装修攻略和装修资讯）的发布管理以及用户曝光信息核查等操作。
2. 公司负责人端主要是公司负责人进入系统，可以进行公司入驻申请、公司基本信息的修改和证书等的上传、疑惑解答和装修投标等操作。
3. 设计师端主要是设计师登录系统，可以进行的操作包括上传个人装修案例和预约受理等操作。
4. 平台用户端主要是用户登录平台，可以查看平台页面的展示信息、发布装修疑惑和曝光信息、发布装修招标信息和预约设计师等操作。

# 需求分析

## 业务需求

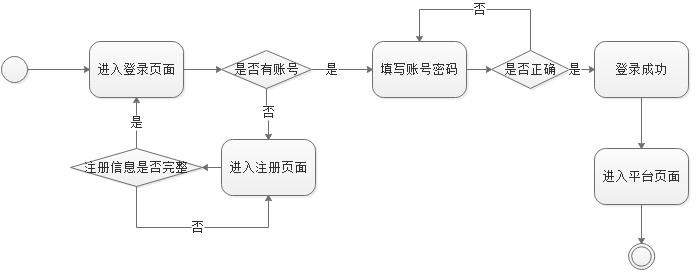
* + 1. 平台用户在进入系统之后，如果要进行特定的操作，需要进行登录操作。
    2. 平台用户可以对一些公司的不良行为或者对公司服务的不满意等进行曝光。平台用户登录平台发布相关的装修曝光信息，平台管理员对曝光进行审核，审核通过则曝光信息在平台的前台装修曝光页面展示，用户都可以查看。
    3. 平台用户可以进行申请装修招标，在平台发布装修招标信息，装修公司进行投标，投标完成后成功招标，双方达成初步合作。
    4. 平台用户可以选择设计师进行预约，填写相关信息，设计师能够对预约进行处理，信息无误后进行预约受理，用户预约设计师成功。
    5. 公司入驻平台需要公司负责人填写相关信息和上传有关文件证明等，上传后平台管理员对其入驻申请表进行审核，审核通过则入驻成功，否则失败。
    6. 平台管理员能够进行装修服务信息的管理，对一些装修资讯和攻略信息的发布，在前台展示以供用户查看了解。
    7. 平台管理员可以对装修公司基本信息进行增删该查的操作，装修公司的基本信息可以在用户客户端进行展示。
    8. 设计师能够在平台上传个人的装修案例，包括效果图和对应的相关信息，上传成功后能够在前台展示，用户可随时查看。
    9. 主要业务流程
       1. 用户登录业务。首先用户进入平台的登录页面，如果还没有账号，点击注册进入注册页面，注册信息完整则注册成功，否则注册失败；注册成功后跳转到登录界面， 填写账号、密码和验证码等，系统判断账号、密码和验证码是否正确，核对正确则登录成功进入平台页面，否则登录失败。相应的活动图如图 2-1 所示：

图 2-1 用户登录活动图

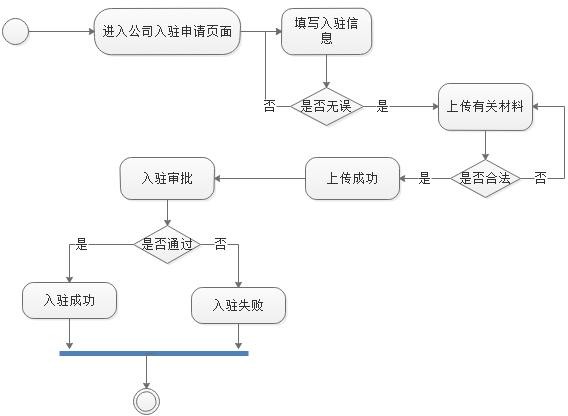
* + - 1. 公司入驻。公司负责人进入公司入驻申请页面填写入驻信息，信息无误后上传相关材料，材料合法则上传成功，完成入驻申请；平台管理员对入驻申请表进行审核， 审核通过则入驻成功，否则入驻失败。相应的活动图如图 2-2 所示：

图 2-2 公司入驻活动图

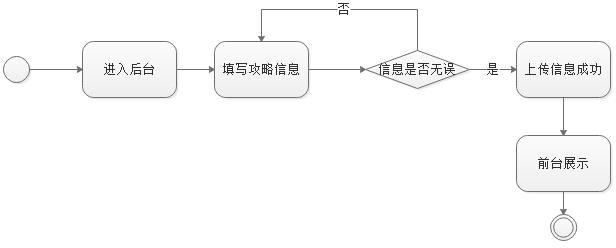
* + - 1. 装修攻略管理。平台管理员登录平台，进入平台管理员后台管理界面，添加装修攻略相关信息，系统判断信息无误后信息上传完成在前台展示。相应的活动图如图2-3所示：

图 2-3 装修攻略管理活动图

* + - 1. 装修案例展示。设计师登录后进入后台，上传效果图及相关信息，图片合法后上传成功，对应的装修案例效果图在前台展示，否则图片上传失败。相应的活动图如图 2-4 所示：

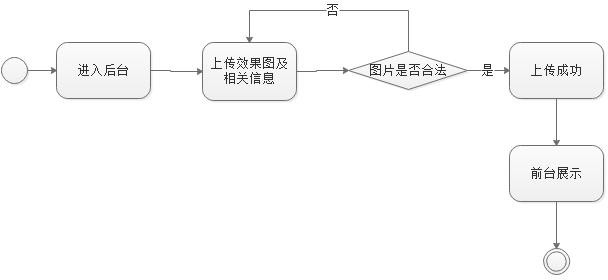


图 2-4 装修案例展示活动图

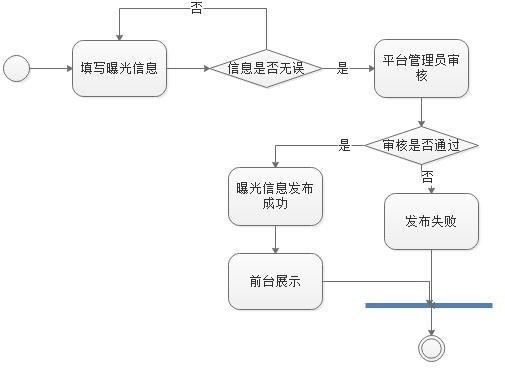
* + - 1. 装修投诉曝光。用户进入平台后填写装修曝光信息，信息无误后上传成功， 平台管理员对装修曝光信息进行真实性审核，审核通过后装修曝光信息发布成功，信息在前台曝光展示。相应的活动图如图 2-5 所示：

图 2-5 装修投诉曝光活动图

* + - 1. 装修招标。平台用户登录平台后填写装修招标信息，信息齐全且正确则发布成功，公司管理员选择相应的招标进行投标。相应的活动图如图 2-6 所示：

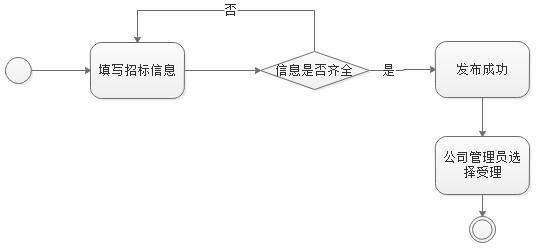


图 2-6 装修招标活动图

* + - 1. 设计师风采展示。设计师登录平台后台，上传信息及案例，无误后上传成功， 在前台进行展示。相应的活动图如图 2-7 所示：

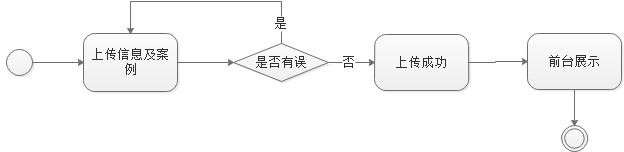


图 2-7 设计师风采展示活动图

* + - 1. 设计师预约。平台用户登录平台填写设计师预约信息，信息判断完整后提交成功，设计师对预约进行受理，受理完成后预约成功。相应的活动图如图 2-8 所示：

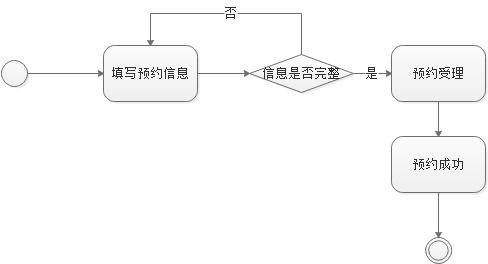


图 2-8 设计师预约活动图

## 功能需求

* + 1. 角色分析

以下从装修家居服务平台的实际需求分析，系统涉及到以下角色，如表 2-1 所示：

表 2-1 系统角色表

|  |  |
| --- | --- |
| 用户 | 功能 |
| 平台管理员 | 对公司入驻进行审批，发布装修服务信息，添加公司基本  信息以及对曝光投诉信息进行核查等。 |
| 公司负责人 | 进行公司入驻申请，上传公司的资质证书，解答平台用户  的装修疑惑，进行装修投标等。 |
| 设计师 | 上传个人装修案例和预约受理等。 |
| 平台用户 | 能够浏览平台前台所展示的各类信息，预约设计师，发布  装修疑惑和曝光信息，进行线上装修招标等。 |

* + 1. 功能需求

以上从业务角度出发，给出了系统的总体用例，包含用户管理、装修服务信息管理、装修交流平台管理、装修招标管理、装修公司信息管理、设计师风采展示管理等用例， 系统总用例图如图 2-9 所示：

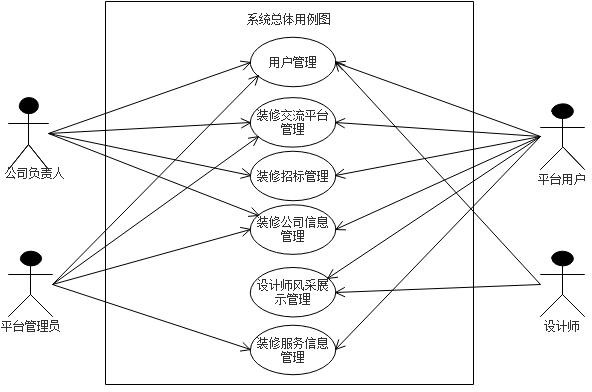


图 2-9 系统总体用例图

* + - 1. 用户管理包括用户注册、用户登录、添加用户和删除用户，用例图如图 2-10

所示：

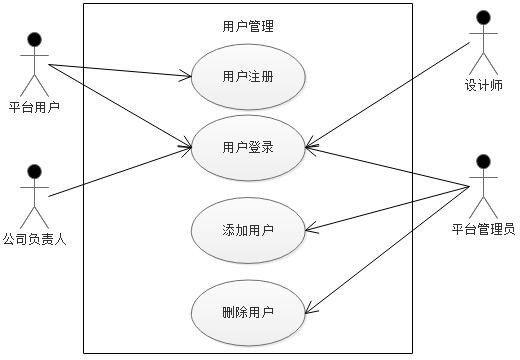


图 2-10 用户管理用例图

添加用户功能的用例描述如表 2-2：

表 2-2 添加用户用例描述表

|  |  |
| --- | --- |
| 用例名称 | 添加用户 |
| 用例标识符 | YL01 |
| 用例描述 | 平台管理员可以进行添加公司负责人和设计师等用户的操作 |
| 参与者 | 平台管理员 |
| 前置条件 | 平台管理员登录成功进入平台管理员后台界面 |
| 基本操作流 | 1. 平台管理员点击左栏用户管理，选择新增公司负责人或者设计 师； 2. 点击新增按钮； 3. 输入相应用户的新增信息； 4. 系统对其输入的信息进行合法验证以及新增用户是否已经存 在。 |
| 后置条件 | 用户添加成功 |

* + - 1. 装修服务信息管理包括装修攻略管理和装修资讯管理，用例图如图 2-11 所示：

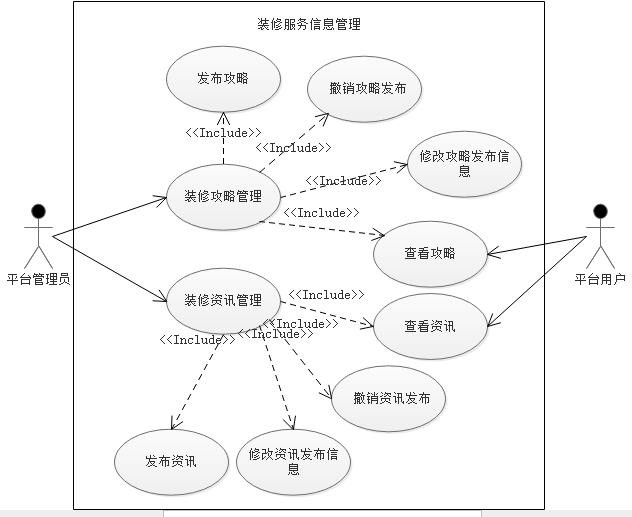


图 2-11 装修服务信息管理用例图

发布攻略用例描述如表 2-3：

表 2-3 发布攻略用例描述表

|  |  |
| --- | --- |
| 用例名称 | 发布攻略 |
| 用例标识符 | YL02 |
| 用例描述 | 平台管理员可以进行装修攻略的发布，以供平台用户浏览查看 |
| 参与者 | 平台管理员 |
| 前置条件 | 平台管理员登录平台成功 |
| 基本操作流 | 1. 平台管理员点击左栏装修攻略管理； 2. 点击新增按钮； 3. 填写相应的装修攻略信息，上传相应的图片； 4. 点击发布按钮； 5. 系统检验填写的信息是否合法，图片是否上传成功。 |
| 后置条件 | 系统提示攻略发布成功 |

查看资讯用例描述如表 2-4：

表 2-4 查看资讯用例描述表

|  |  |
| --- | --- |
| 用例名称 | 查看资讯 |
| 用例标识符 | YL03 |
| 用例描述 | 装修资讯信息发布后，平台管理员和平台用户可以对其进行浏览 |
| 参与者 | 平台管理员和平台用户 |
| 前置条件 | 装修资讯信息成功发布 |
| 基本操作流 | 1. 平台管理员点击左栏装修资讯管理，选择要浏览的资讯，点击 浏览按钮。 2. 平台用户进入平台前台页面，点击装修资讯栏，选择需要浏览   的资讯，点击链接查看资讯信息。 |
| 后置条件 | 装修资讯信息浏览成功 |

* + - 1. 装修交流平台管理用例图如图 2-12 所示：

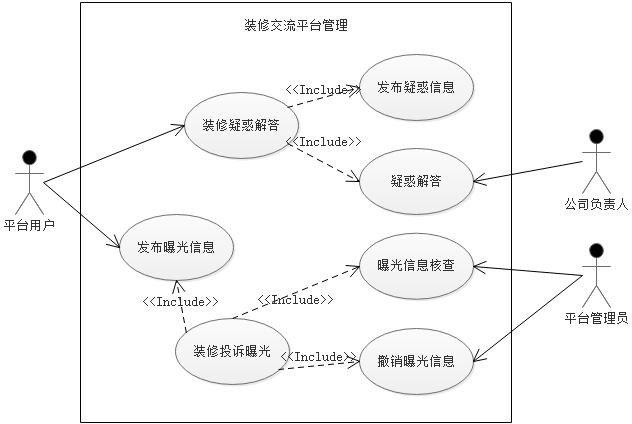


图 2-12 装修交流平台管理用例图

疑惑解答用例描述如表 2-5：

表 2-5 疑惑解答用例描述表

|  |  |
| --- | --- |
| 用例名称 | 疑惑解答 |
| 用例标识符 | YL04 |
| 用例描述 | 装修疑惑内容发布成功后，公司负责人可以对其进行解答 |
| 参与者 | 公司负责人 |
| 前置条件 | 公司负责人用户登录平台成功，且装修疑惑内容发布成功 |
| 基本操作流 | 1. 公司负责人选择点击装修疑惑链接； 2. 填写解答内容； 3. 点击提交； 4. 系统验证信息是否合法且正确。 |
| 后置条件 | 完成装修疑惑解答 |

曝光信息核查用例描述如表 2-6：

表 2-6 曝光信息核查用例描述表

|  |  |
| --- | --- |
| 用例名称 | 曝光信息核查 |
| 用例标识符 | YL05 |
| 用例描述 | 在曝光信息上传成功后，平台管理员需对其进行真实性的核查，  核查通过才能进行曝光展示 |
| 参与者 | 平台管理员 |
| 前置条件 | 平台管理员登录平台成功 |
| 基本操作流 | 1. 平台管理员点击曝光信息管理栏； 2. 选择未核查的曝光信息； 3. 查看内容，进行真实性的核查； 4. 选择核查有效； 5. 点击曝光按钮。 |
| 后置条件 | 核查通过，曝光信息在平台前台显示 |

* + - 1. 装修招标管理用例图如图 2-13 所示：

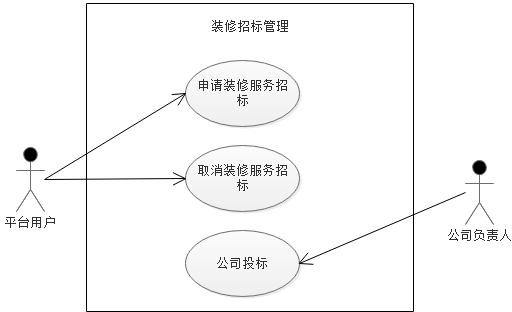


图 2-13 装修招标管理用例图

申请装修服务招标用例描述如表 2-7：

表 2-7 申请装修服务招标用例描述表

|  |  |
| --- | --- |
| 用例名称 | 申请装修服务招标 |
| 用例标识符 | YL06 |
| 用例描述 | 平台用户可以在平台内申请装修招标服务，供装修公司进行投标 |
| 参与者 | 平台用户 |
| 前置条件 | 平台用户登录平台成功 |
| 基本操作流 | 1. 平台用户点击装修招标栏； 2. 填写相应的招标信息； 3. 点击提交按钮； 4. 系统检测填写的数据信息是否合法正确。 |
| 后置条件 | 系统提示装修招标信息发布成功 |

公司投标用例描述如表 2-8：

表 2-8 公司投标用例描述表

|  |  |
| --- | --- |
| 用例名称 | 公司投标 |
| 用例标识符 | YL07 |
| 用例描述 | 在平台用户发布装修服务招标信息成功后，公司负责人可以选择  投标，建立平台用户与装修公司的合作关系 |
| 参与者 | 公司负责人 |
| 前置条件 | 公司负责人登录平台成功，且装修服务招标信息已发布 |
| 基本操作流 | 1. 公司负责人点击装修招标栏； 2. 查看招标信息； 3. 点击投标按钮。 |
| 后置条件 | 系统提示投标成功 |

* + - 1. 装修公司信息管理用例图如图 2-14 所示：

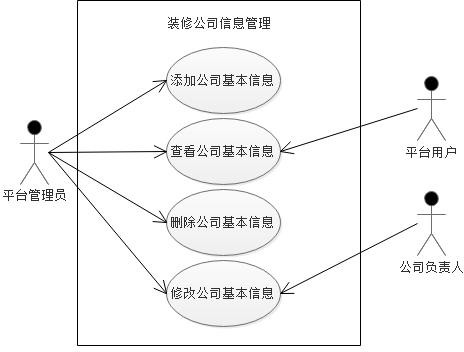


图 2-14 装修公司信息管理用例图

添加装修公司基本信息用例描述如表 2-9：

表 2-9 添加装修公司基本信息用例描述表

|  |  |
| --- | --- |
| 用例名称 | 添加装修公司基本信息 |
| 用例标识符 | YL08 |
| 用例描述 | 装修公司入驻平台后，平台管理员需要进行添加装修公司基本信  息操作 |
| 参与者 | 平台管理员 |
| 前置条件 | 平台管理员登录平台成功 |
| 基本操作流 | 1. 平台管理员点击左栏装修公司信息管理； 2. 点击新增按钮； 3. 填写相应的装修公司基本信息； 4. 点击提交按钮； 5. 系统自动验证填写的信息是否合法。 |
| 后置条件 | 系统提示装修公司基本信息添加成功 |

* + - 1. 设计师风采展示管理用例图如图 2-15 所示：

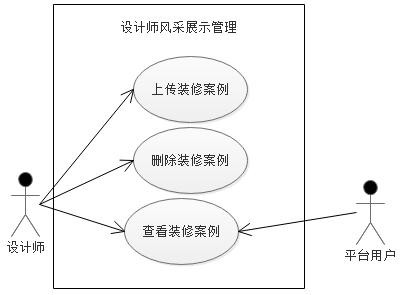


图 2-15 设计师风采展示管理用例图

上传装修案例用例描述如表 2-10：

表 2-10 上传装修案例用例描述表

|  |  |
| --- | --- |
| 用例名称 | 上传装修案例 |
| 用例标识符 | YL09 |
| 用例描述 | 设计师可以上传装修案例，上传完成后将在平台的装修案例栏展  示 |
| 参与者 | 设计师 |
| 前置条件 | 设计师登录平台成功 |
| 基本操作流 | 1. 设计师点击左栏风采展示管理； 2. 点击新增按钮； 3. 填写相应的装修风采信息，上传相应的图片； |
| 基本操作流 | 1. 点击上传按钮； 2. 系统检验填写的信息是否合法，图片是否上传成功。 |
| 后置条件 | 系统提示装修案例上传成功 |

## 非功能需求

* + 1. 环境需求

服务器端环境表如表 3-1：

表 3-1 服务器端环境表

|  |  |
| --- | --- |
| 系统所使用数据库 | MySQL 5.0 |
| 操作系统 | Windows 10 专业版 |
| Web 服务器 | Tomcat 8.0 |
| 开发环境 | IDEA 2017.2+Maven 3.5.4 |

客户端环境表如表 3-2：

表 3-2 客户端环境表

|  |  |
| --- | --- |
| 操作系统 | Windows XP 及其以上版本操作系统 |
| 浏览器 | IE 7.0 及其更高版本，Chrome 浏览器和 Firefox 浏览器 |
| 分辨率 | 最佳效果为 1028×960 像素 |

* + 1. 性能需求
       1. 易操作性：平台应提供简洁美观、实用方便和视觉直观的图形用户操作界面。
       2. 时间特性：用户浏览页面时，页面响应时间要求在 5 秒之内，不允许出现操作

异常而无相应的情况。

* + - 1. 可扩展性：将系统架构和功能设计等方面的功能需求扩展加以考虑，以便能满足今后对系统功能的完善扩充、迭代升级和扩大使用范围的要求。
      2. 开放性：系统的各种数据格式都要符合国家的标准和行业的标准，以确保系统能够具有较好的可移植性。
    1. 安全需求

1. 容错性

当用户进行不当操作而致使不当数据产生时，系统应具有一定的容错校验机制，例如给出友好的信息提示，提示用户重新输入数据或者进行自动校正。

1. 外在环境安全

系统要应充分考虑到系统的硬件和软件故障恢复等问题，确保网络的安全性和操作安全性。还应充分考虑网络的高级别、多层次安全性问题，及时给出安全措施，这包括系统的备份以及防火墙等措施，确保数据的安全。

1. 内部安全

在保证系统的外部安全同时，该系统还应保证用户的合法操作。

# 总体设计

## 系统设计的原则

在充分分析该平台的总体规划和应用环境后，本平台的设计将遵循以下的设计原

则：

* + 1. 实用性原则：平台的实用性会直接影响到公司效益和用户满意程度。因此在设计该平台时，放在首位的必将是平台的实用性。本平台应紧密联系公司和用户的实际业务需求，达到解决传统装修行业服务效率低和不透明问题的目标。
    2. 规范性原则：平台中所有设计都应遵从当前所有的国家和行业标准，应符合软件工程设计规范。系统中采用的编解码协议、传输协议等应符合国家相应的标准，以便能够互联互通和系统移植。
    3. 可扩展性原则：平台设计要考虑到功能业务未来发展的需求，设计应该尽可能地简洁明了，应尽量降低各模块结构的耦合度。除此之外，还应充分考虑到平台的可兼容性和可扩展性原则。

## 系统体系结构设计

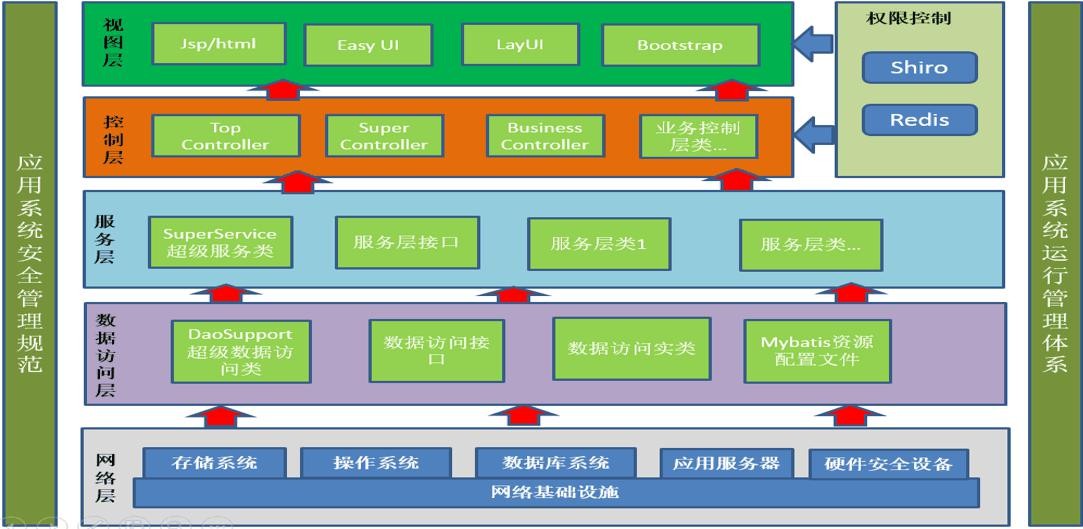
目前项目设计主要的体系结构有 B/S（Browser/Server）结构、C/S(Client/Server) 结构以及 C/S 与 B/S 的混合结构。由于希望系统的维护简单、方便并且根据用户需求和装修家居服务平台本身的特点，决定系统的体系结构采用 B/S 构架。系统结构如图 3-1 所示：

图 3-1 装修家居服务平台系统结构图

本平台采用的是基于 Maven 项目管理框架下的 SSM 架构设计。其中，Maven 是一个用于 Java 相关项目管理的框架工具，使用 Maven 框架搭建的 Web 项目，其项目结构都相同。Maven 项目在 pom.xml 文件中配置项目中使用到的外部 jar 包，框架会直接将所有 jar 包下载到本地“仓库”里，方便项目的迭代和升级。SSM 框架是当前行业比较流行的 Web 项目开发框架，是由 Spring、Spring MVC、MyBatis 三个框架整合而成的完整性框架，其中 Spring MVC 是 Spring 框架的一部分。

其中，Spring 是一个轻量级开源的 JavaWeb 软件开发框架。Spring 使用的是基本的JavaBean 管理来实现往常只可能由 EJB（Enterprise Java Beans）来实现的功能。从简洁性、可测试性和复用性等的的角度考虑，基本上所有的 Java 应用都可以借助于 Spring 并从中收益。简而言之，Spring 是一个轻量级的拥有控制反转（IOC）和面向切面（AOP） 功能的“容器”框架。控制反转也叫做依赖注入，其利用工厂模式将对象交给“容器” 进行管理，因而开发者只需要在 Spring 配置文件配置相应的 JavaBean，设置添加相关的属性，Spring 容器会自动创建类的实例对象。面向切面是对面向对象的补充和完善。在本系统的开发中采用的是动态代理的技术，采用切取消息的方式对消息加以修饰，用于获得原有对象方法的执行。另外采用了 Spring 注解的方式，对 JavaBean 进行单例创建。

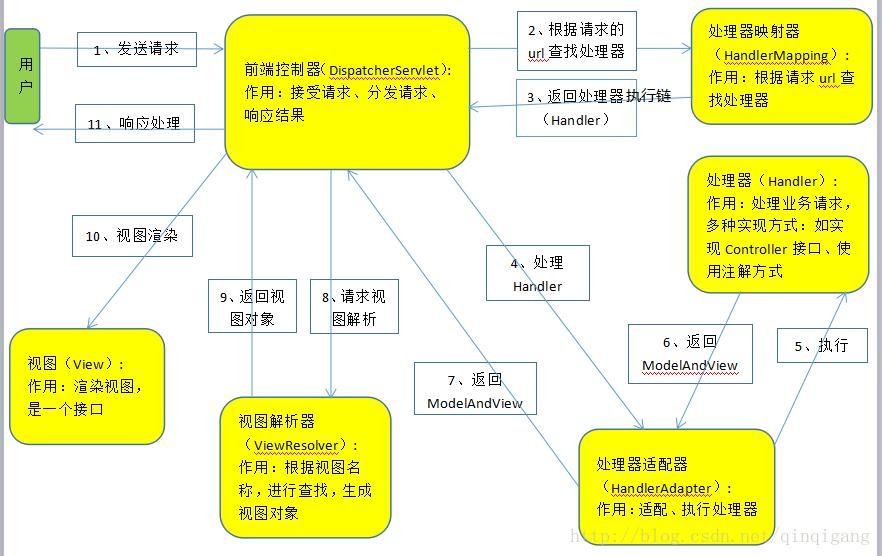
Spring MVC 是Spring 框架中用于控制前台数据和后台数据交互的表现层框架部分， 其具有结构松散特点和松耦合的优点，Spring MVC 使各模块之间进行分离，有利于系统模块的维护和扩充。其具有严格简明的执行流程，Spring MVC 执行流程如图 3-2 所示：

图 3-2 Spring MVC 执行流程图

执行流程说明：

用户首先将请求发送到前端控制器等待分发，然后前端控制器根据请求的地址查找处理器映射器查找相应的处理器执行链，接着处理器映射器返回处理器执行链到前端控制器，接着前端控制器调用处理器适配器去执行处理器执行链，处理器适配器去执行处理器执行链，接着处理器执行链执行完成给适配器返回 ModelAndView 对象，接着处理器适配器向前端控制器返回 ModelAndView 对象，然后前端控制器请求视图解析器去进行视图解析，根据视图名称进行查找，生成视图对象，接着视图解析器向前端控制器返回视图对象，前端控制器再进行视图渲染，视图渲染将模型数据（在 ModelAndView 对象中）填充到请求域中，最后前端控制器向用户响应执行结果。

MyBatis 是一个持久层框架，是 apache 下的顶级项目。使用 Mybatis，开发者只要将主要经理放在 SQL 的编写上，通过 Mybatis 提供的映射方式，自由灵活地生成满足需要的 SQL 语句，其生成方式是半自动化的，大部分需要开发者编写 SQL。Mybatis 可以向PrepareStatement 中输入参数自动进行输入映射，将查询结果集灵活地映射出 Java 对象

（即输出映射）Mybatis 系统结构图如 3-3 所示：

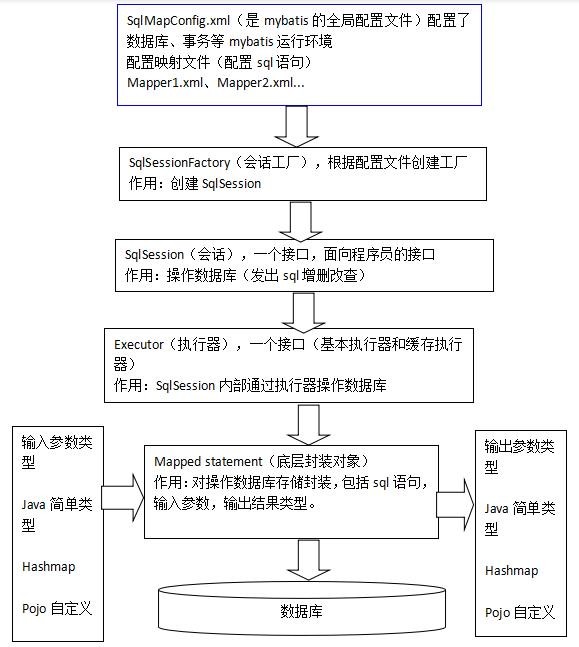


图 3-3 Mybatis 系统结构图

Mybatis 相对于 JDBC 的好处：

1. 使用传统方式的 JDBC 访问数据库：

①使用 JDBC 访问数据库有大量重复代码（比如注册驱动、获取连接、获取传输器、释放资源等）。

②原生 JDBC 自身是没有连接池的，需要不断创建和关闭连接，其效率极低。

③SQL 是写死在程序中，一旦修改 SQL，需要对类重新编译。

④对查询 SQL 执行后返回的 ResultSet 对象，需要手动处理，有时会特别麻烦。

1. 使用 MyBatis 框架访问数据库：

①Mybatis 对 JDBC 进行了封装，可以简化 JDBC 代码。

②Mybatis 支持连接池（也可以配置其他的连接池），因此可以提高程序的效率

③Mybatis 是将 SQL 配置在 mapper 文件中，修改 SQL 只是修改配置文件，类不需要重新编译。

④对查询 SQL 执行后返回的 ResultSet 对象，Mybatis 会帮我们处理，转换成 Java

对象。

## 系统功能模块设计

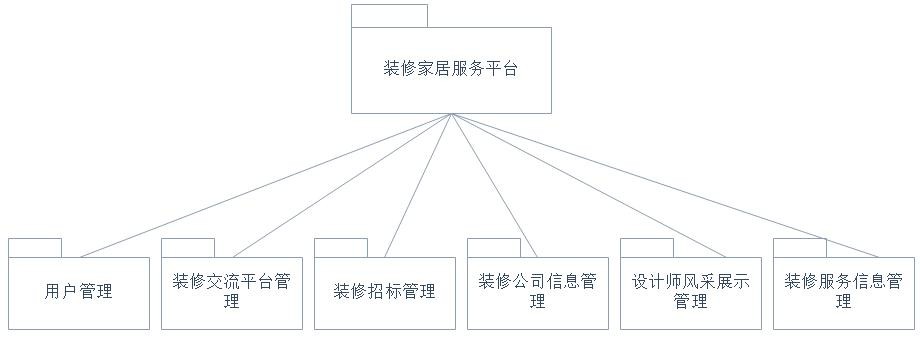
本系统主要包括用户管理、装修交流平台管理、装修招标管理、装修公司信息管理、设计师风采展示管理和装修服务信息管理等功能模块，系统总功能架构包图如图 3-4 所示：

图 3-4 系统总功能架构包图

1. 用户管理具体功能包括平台用户的用户注册、登录，以及平台管理员添加和删除公司负责人、设计师的用户操作。
2. 装修交流平台管理是为用户提供的装修交流的平台，具体功能主要包括装修疑惑解答和装修投诉曝光。客户发布装修相关疑惑问题，用户或者装修人员进行解答交流， 用户发布曝光投诉信息，平台管理员进行曝光信息核查，核查通过公开展示，核查不通过撤销曝光等。
3. 装修招标管理是提供给用户自主发布装修招标信息，供装修公司选择接受服务的功能，具体功能包括用户申请装修服务招标、取消装修服务招标、招标信息公开展示和公司选择接受招标等。
4. 装修公司信息管理是公司入驻后对装修公司的信息进行管理和装修公司对装修案例等的管理。具体功能包括对公司基本信息的添加、修改、删除，基本信息的展示（ 包括基本信息简介展示、资质证书展示、擅长领域信息展示和装修保障条例展示等） 。
5. 设计师风采展示管理主要是提供给设计师进行风采和信息展示的管理，具体功能包括上传信息及案例、设计师信息的展示和设计师装修案例展示等。
6. 装修服务信息管理是平台管理员进行装修攻略的管理和装修资讯的管理，具体功能包括装修前、中、后期各方面的装修攻略的发布展示，资讯的添加、撤销、展示和修改等。

# 数据库设计

## 概念结构设计

* + 1. 设计思路

利用 E-R 结构对数据的概念结构进行分析，总体上可以分 3 步：首先设计出各个模块内部的 E-R 结构，再把各模块每部的 E-R 结构按照各个实体之间的联系整合成全局的E-R 结构，最后对全局的 E-R 结构进行优化和完善。根据平台各个用户实体的各种操作模块，以及模块之间的联系进行设计。平台管理员用户能够进行公司入驻审批、公司基本信息管理、装修服务信息管理和装修曝光信息的核查等操作。公司负责人可以进行公司入驻申请、公司基本信息管理、装修疑惑的解答、装修工地展示管理和装修投标等操作。设计师可以上传装修案例和预约受理等操作。平台用户能够发布装修疑惑信息和曝光信息、进行装修招标管理、装修成果反馈等操作。根据这些基本信息进行本平台的概念设计。

* + 1. E-R 图

E-R 图模型的联系是实体之间关联的描述。依据系统需求分析的结果，考虑局部结构中的各个实体之间存在的一对一、一对多和多对多关系。另外还需要考虑实体内部和实体之间存在的联系等。

1. 本系统的实体关系 ER 图，如图 4-1 所示。

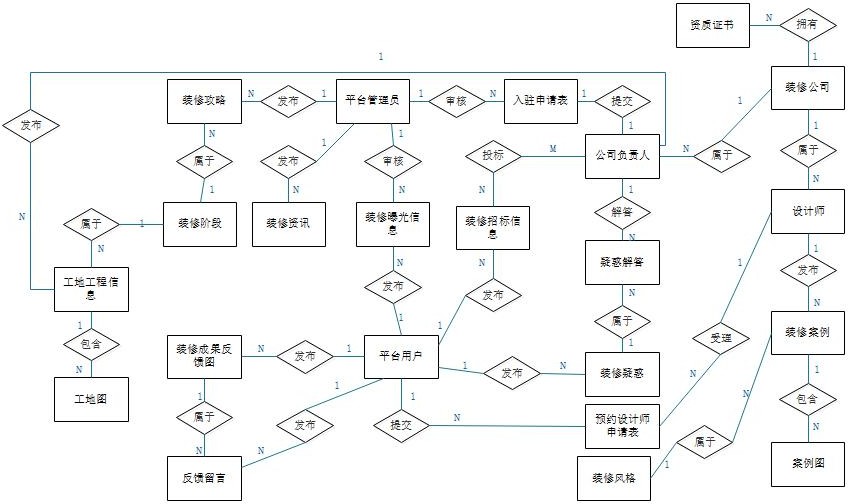


图 4-1 实体关系 ER 图

1. 平台管理员的实体图，如图 4-2 所示： 实体：平台管理员

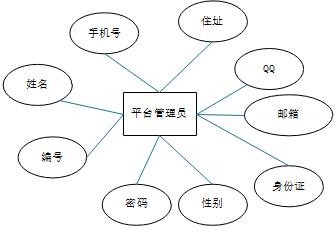
属性：编号，姓名，性别，邮箱，密码，手机号，QQ，身份证，住址。

图 4-2 平台管理员实体图

1. 公司负责人的实体图，如图 4-3 所示： 实体：公司负责人

属性：编号，姓名，住址，电话，QQ，身份证，邮箱，密码，职位，装修公司编号， 招标信息/公司负责人关联编号。

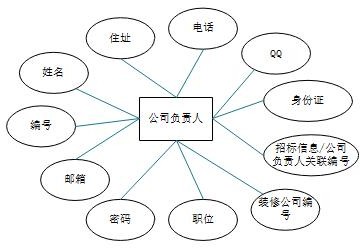


图 4-3 公司负责人实体图

1. 设计师的实体图，如图 4-4 所示： 实体：设计师

属性：编号，姓名，毕业院校，职位，学历，从业年限，设计理念，擅长户型，擅长风格，照片，家庭地址，手机号，邮箱，密码，装修公司编号。

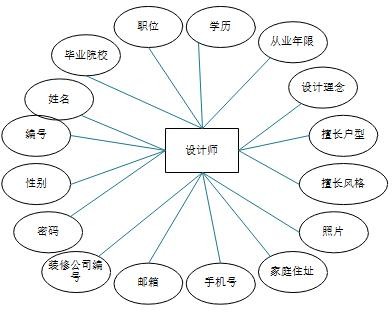


图 4-4 设计师实体图

1. 平台用户的实体图，如图 4-5 所示： 实体：平台用户

属性：编号，用户称呼，性别，头像，邮箱，用户密码，手机号，QQ，通讯地址。

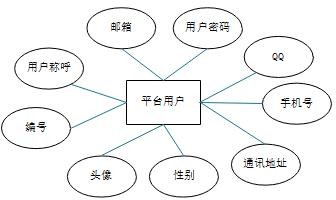


图 4-5 平台用户实体图

1. 装修公司的实体图，如图 4-6 所示： 实体：装修公司

属性：编号，名称，公司地址，所在区域，口碑值，营业执照状态，公司介绍，公司 logo，擅长风格，擅长领域，承接价位，装修保险金，装修保障条例，入驻状态，邮箱，电话，装修模式，注册资本。

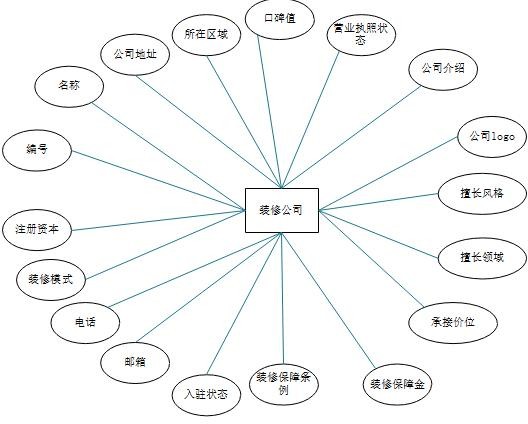


图 4-6 装修公司实体图

1. 入驻申请表的实体图，如图 4-7 所示： 实体：入驻申请表

属性：编号，申请表名，公司名，联系人，电话，QQ，合作意向，受理日期，审批状态，提交日期，公司负责人编号，平台管理员编号。

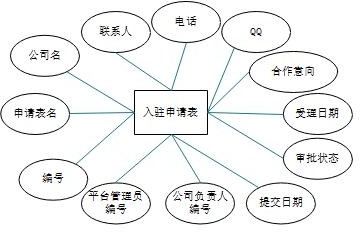


图 4-7 入驻申请表实体图

1. 装修攻略的实体图，如图 4-8 所示： 实体：装修攻略

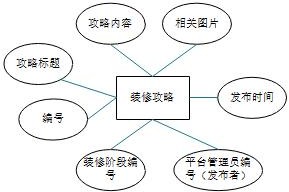
属性：编号，攻略标题，攻略内容，相关图片，发布时间，平台管理员编号，装修阶段编号。

图 4-8 装修攻略实体图

1. 装修资讯的实体图，如图 4-9 所示： 实体：装修资讯

属性：编号，资讯标题，资讯内容，相关图片，发布时间，平台管理员编号。

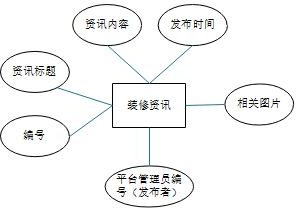


图 4-9 装修资讯实体图

1. 装修案例的实体图，如图 4-10 所示： 实体：装修案例

属性：编号，案例名，房屋面积，房屋类型，装修风格编号，设计师编号。

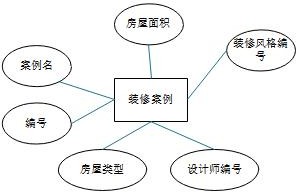


图 4-10 装修案例实体图

1. 案例图的实体图，如图 4-11 所示： 实体：案例图

属性：编号，图片名称，图片地址，图片说明，装修案例编号。

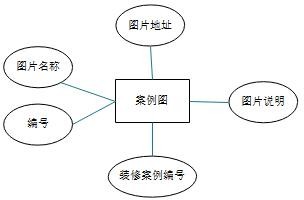


图 4-11 案例图实体图

1. 装修曝光信息的实体图，如图 4-12 所示： 实体：装修曝光信息

属性：编号，曝光标题，曝光公司，所在地区，曝光时间，曝光详情，曝光图片， 审核状态，平台用户编号，平台管理员编号。

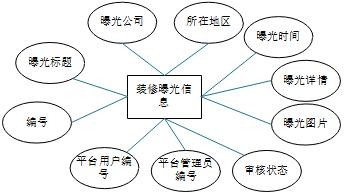


图 4-12 装修曝光信息实体图

1. 装修疑惑的实体图，如图 4-13 所示： 实体：装修疑惑

属性：编号，疑惑标题，疑惑内容，回答数，提问时间，平台用户编号。

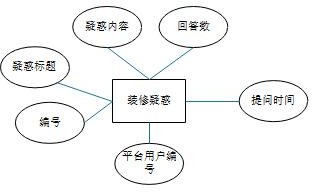


图 4-13 装修疑惑实体图

1. 疑惑解答的实体图，如图 4-14 所示： 实体：疑惑解答

属性：编号，解答内容，解答时间，点赞数，公司负责人编号。

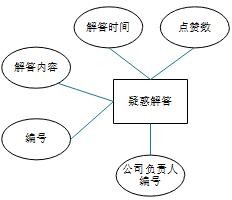


图 4-14 疑惑解答实体图

1. 资质证书的实体图，如图 4-15 所示： 实体：资质证书

属性：编号，图片名称，图片地址，图片说明，装修公司编号。

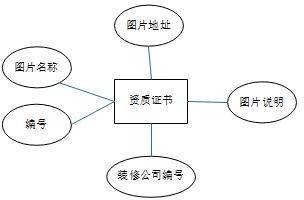


图 4-15 资质证书实体图

1. 装修招标信息的实体图，如图 4-16 所示： 实体：装修招标信息

属性：编号，房屋类型，招标类型，所在区域，小区名称，初步预算，房屋面积， 发布时间，装修时间，量房时间，招标要求，平台用户编号，公司负责人/招标信息关联编号。

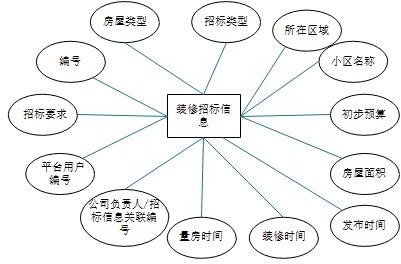


图 4-16 装修招标信息实体图

1. 装修风格的实体图，如图 4-17 所示： 实体：装修风格

属性：编号，风格名称，风格介绍。

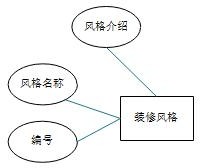


图 4-17 装修风格实体图

1. 装修阶段的实体图，如图 4-18 所示： 实体：装修阶段

属性：编号，阶段名，阶段介绍。

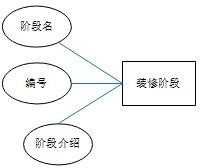


图 4-18 装修阶段实体图

1. 工地工程信息的实体图，如图 4-19 所示： 实体：工地工程信息

属性：编号，工程名称，工地小区，工地地址，工地面积，房屋类型，装修阶段编号，房屋类型。

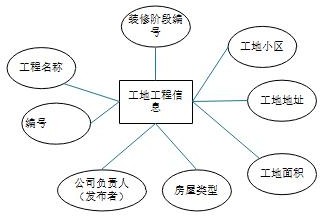


图 4-19 工地工程信息实体图

1. 工地图的实体图，如图 4-20 所示： 实体：工地图

属性：编号，图片名称，图片地址，图片说明，工地工程编号。

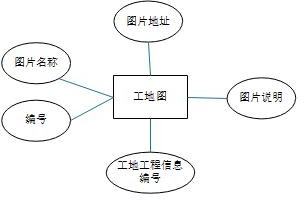


图 4-20 工地图实体图

1. 装修成果反馈图的实体图，如图 4-21 所示： 实体：装修成果反馈图

属性：编号，图片名称，图片地址，图片说明，发布时间，留言数，平台用户编号。

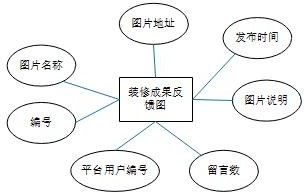


图 4-21 装修成果反馈图实体图

1. 反馈留言的实体图，如图 4-22 所示： 实体：反馈留言

属性：编号，留言内容，留言时间，点赞数，平台用户编号，装修成果反馈图编号。

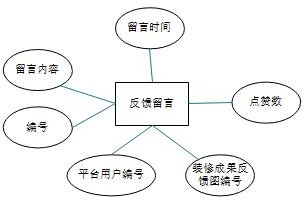


图 4-22 反馈留言实体图

1. 预约设计师申请表的实体图，如图 4-23 所示： 实体：预约设计师申请表

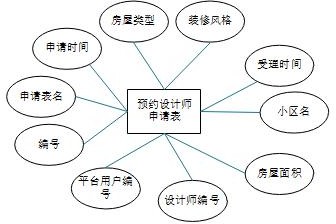
属性：编号，申请表名，申请时间，房屋类型，装修风格，受理时间，小区名，房屋面积，设计师编号，平台用户编号。

图 4-23 预约设计师申请表实体图

## 逻辑结构设计

* + 1. 设计思路

数据库逻辑结构的设计首先将系统需求转化为数据库的逻辑模型，然后将转换来的实体关系、网状结构以及层次模型向特定数据库管理系统（DBMS）支持下的数据模型转换，最后对数据模型进行优化便得出数据库的逻辑结构。

* + 1. 逻辑模型
       1. 平台管理员表：（编号，名称，性别，邮箱，密码，手机号，QQ，身份证，住址）
       2. 公司负责人表：（编号，名称，住址，电话，QQ，身份证，邮箱，密码，职位， 装修公司编号，招标信息/公司负责人关联编号）
       3. 设计师表：（编号，名称，毕业院校，职位，学历，从业年限，设计理念，擅长户型，擅长风格，照片，家庭地址，手机号，邮箱，密码，装修公司编号）
       4. 平台用户表：（编号，用户称呼，性别，头像，邮箱，用户密码，手机号，QQ，通讯地址）
       5. 装修公司表：（编号，名称，公司地址，所在区域，口碑值，营业执照状态， 公司介绍，公司 logo，擅长风格，擅长领域，承接价位，装修保险金，装修保障条例， 入驻状态，邮箱，电话，装修模式，注册资本）
       6. 入驻申请表：（编号，申请表名，公司名，联系人，电话，QQ，合作意向，受理日期，审批状态，提交日期，公司负责人编号，平台管理员编号）
       7. 装修攻略表：（编号，攻略标题，攻略内容，相关图片，发布时间，平台管理员编号，装修阶段编号）
       8. 装修资讯表：（编号，资讯标题，资讯内容，相关图片，发布时间，平台管理员编号）
       9. 装修案例表：（编号，案例名，房屋面积，房屋类型，装修风格编号，设计师编号）
       10. 案例图表：（编号，图片名称，图片地址，图片说明，装修案例编号）
       11. 装修曝光信息表：（编号，曝光标题，曝光公司，所在地区，曝光时间，曝光详情，曝光图片，审核状态，平台用户编号，平台管理员编号）
       12. 装修疑惑表：（编号，疑惑标题，疑惑内容，回答数，提问时间，平台用户编号）
       13. 疑惑解答表：（编号，解答内容，解答时间，点赞数，公司负责人编号）
       14. 资质证书表：（编号，图片名称，图片地址，图片说明，装修公司编号）
       15. 装修招标信息表：（编号，房屋类型，招标类型，所在区域，小区名称，初步预算，房屋面积，发布时间，装修时间，量房时间，招标要求，平台用户编号，公司负责人/招标信息关联编号）
       16. 装修风格表：（编号，风格名称，风格介绍）
       17. 装修阶段表：（编号，阶段名，阶段介绍）
       18. 工地工程信息表：（编号，工程名称，工地小区，工地地址，工地面积，房屋类型，装修阶段编号，房屋类型）
       19. 工地图表：（编号，图片名称，图片地址，图片说明，工地工程编号）
       20. 装修成果反馈图表：（编号，图片名称，图片地址，图片说明，发布时间， 留言数，平台用户编号）
       21. 反馈留言表：（编号，留言内容，留言时间，点赞数，平台用户编号，装修成果反馈图编号）
       22. 预约设计师申请表：（编号，申请表名，申请时间，房屋类型，装修风格， 受理时间，小区名，房屋面积，设计师编号，平台用户编号）

注：以上表中加有下划线的属性为该表的主键。

## 物理结构设计

* + 1. 存取方式

本平台采用的是 MYSQL 结构型数据库进行数据的存储。当前流行的数据库管理系统

（DBMS）一般都提供了多种存取方法，如索引法、HASH 法等。本平台中使用的是索引法。在数据库中，索引法可以实现应用程序快速地找到表中的数据，而没必要取扫描整个数据库。这是因为索引其实也就是表中数据与对应的存储位置的列表，使用索引法能够极大程度上减少数据的查询时间。

* + 1. 存储结构

1. 平台管理员表 administrator，如表 4-1 所示：

表 4-1 平台管理员表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 名称 | varchar | 20 |  |
| 3 | gender | 性别 | varchar | 5 |  |
| 4 | email | 邮箱 | varchar | 35 |  |
| 5 | password | 密码 | varchar | 100 |  |
| 6 | telephone | 手机号 | varchar | 35 |  |
| 7 | address | 住址 | varchar | 100 |  |
| 8 | idnum | 身份证号 | varchar | 35 |  |

1. 公司负责人表 manager，如表 4-2 所示：

表 4-2 公司负责人表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 名称 | varchar | 20 |  |
| 3 | email | 邮箱 | varchar | 35 |  |
| 4 | password | 密码 | varchar | 100 |  |
| 5 | idnum | 身份证号 | varchar | 35 |  |
| 6 | telephone | 手机号 | varchar | 35 |  |

续表 4-2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 7 | qq | QQ | varchar | 35 |  |
| 8 | post | 职位 | varchar | 10 |  |
| 9 | address | 住址 | varchar | 100 |  |
| 10 | companyid | 装修公司编号 | varchar | 35 | 外键 |

1. 设计师表 desiner，如表 4-3 所示：

表 4-3 设计师表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 名称 | varchar | 20 |  |
| 3 | gender | 性别 | varchar | 5 |  |
| 4 | email | 邮箱 | varchar | 35 |  |
| 5 | password | 密码 | varchar | 100 |  |
| 6 | image | 个人照片 | varchar | 100 |  |
| 7 | period | 从业年限 | int | 5 |  |
| 8 | school | 毕业院校 | varchar | 50 |  |
| 9 | post | 职位 | varchar | 10 |  |
| 10 | degree | 学历 | varchar | 50 |  |
| 11 | philosophy | 设计理念 | varchar | 200 |  |
| 12 | housetype | 擅长户型 | varchar | 35 |  |
| 13 | housestyle | 擅长风格 | varchar | 100 |  |
| 14 | address | 家庭住址 | varchar | 100 |  |
| 15 | telephone | 手机号 | varchar | 35 |  |
| 16 | companyid | 装修公司编号 | varchar | 35 | 外键 |

1. 平台用户表 userinfo，如表 4-4 所示：

表 4-4 平台用户表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 11 | 主键 |
| 2 | name | 用户称呼 | varchar | 20 |  |
| 3 | password | 密码 | varchar | 200 |  |
| 4 | email | 邮箱 | varchar | 35 |  |
| 5 | gender | 性别 | varchar | 5 |  |

续表 4-4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 6 | qq | QQ | varchar | 35 |  |
| 7 | telephone | 手机号 | varchar | 35 |  |
| 8 | address | 通讯地址 | varchar | 100 |  |
| 9 | image | 头像 | varchar | 100 |  |

1. 装修公司表 company，如表 4-5 所示：

表 4-5 装修公司表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 公司名称 | varchar | 100 |  |
| 3 | address | 地址 | varchar | 100 |  |
| 4 | district | 区域 | varchar | 20 |  |
| 5 | kbnum | 口碑值 | int | 10 |  |
| 6 | licstate | 营业执照状态 | varchar | 10 |  |
| 7 | logo | 公司 logo | varchar | 100 |  |
| 8 | housestyle | 擅长风格 | varchar | 100 |  |
| 9 | field | 擅长领域 | varchar | 255 |  |
| 10 | cjprice | 承接价位 | double | 10 | 单位：万元 |
| 11 | bzmoney | 装修保障金 | double | 10 | 单位：元 |
| 12 | regulations | 装修保障条例 | varchar | 500 |  |
| 13 | email | 邮箱 | varchar | 35 |  |
| 14 | telephone | 电话 | varchar | 35 |  |
| 15 | pattern | 装修模式 | varchar | 255 |  |
| 16 | zcmoney | 注册资本 | double | 10 | 单位：万元 |
| 17 | introduction | 公司介绍 | varchar | 1000 |  |
| 18 | rzstate | 入驻状态 | varchar | 10 |  |

1. 入驻申请表 rzapplication，如表 4-6 所示：

表 4-6 入驻申请表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 申请表名 | varchar | 100 |  |
| 3 | comname | 公司名称 | varchar | 100 |  |

续表 4-6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 4 | contactname | 联系人姓名 | varchar | 100 |  |
| 5 | telephone | 电话 | varchar | 35 |  |
| 6 | qq | QQ | varchar | 35 |  |
| 7 | purpose | 合作意向 | varchar | 255 |  |
| 8 | subtime | 提交日期 | date | 20 |  |
| 9 | actime | 受理日期 | date | 20 |  |
| 10 | state | 审批状态 | varchar | 10 |  |
| 11 | managerid | 公司负责人编  号 | varchar | 35 | 外键 |
| 12 | adminid | 平台管理员编  号 | varchar | 35 | 外键 |

1. 装修攻略表 strategy，如表 4-7 所示：

表 4-7 装修攻略表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 攻略标题 | varchar | 100 |  |
| 3 | content | 攻略内容 | varchar | 1000 |  |
| 4 | image | 相关图片 | varchar | 100 |  |
| 5 | subtime | 发布时间 | date | 10 |  |
| 6 | stageid | 装修阶段编号 | varchar | 35 |  |
| 7 | adminid | 平台管理员编  号 | varchar | 35 | 外键 |

1. 装修资讯表 information，如表 4-8 所示：

表 4-8 装修资讯表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 资讯标题 | varchar | 100 |  |
| 3 | content | 资讯内容 | varchar | 1000 |  |
| 4 | image | 相关图片 | varchar | 100 |  |
| 5 | subtime | 发布时间 | date | 10 |  |
| 6 | adminid | 平台管理员编号 | varchar | 35 | 外键 |

1. 装修案例表 case，如表 4-9 所示：

表 4-9 装修案例表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 案例名 | varchar | 100 |  |
| 3 | housetype | 房屋类型 | varchar | 35 |  |
| 4 | area | 房屋面积 | int | 10 | 单位：平方米 |
| 5 | desinerid | 设计师编号 | varchar | 35 | 外键 |
| 6 | housestyleid | 装修风格编号 | varchar | 35 | 外键 |

1. 案例图表 casephoto，如表 4-10 所示：

表 4-10 案例图表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 图片名称 | varchar | 100 |  |
| 3 | address | 图片地址 | varchar | 100 |  |
| 4 | description | 图片说明 | varchar | 255 |  |
| 5 | caseid | 装修案例编号 | varchar | 35 | 外键 |

1. 装修曝光信息表 bgmessage，如表 4-11 所示：

表 4-11 装修曝光信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 曝光标题 | varchar | 100 |  |
| 3 | company | 曝光公司名 | varchar | 100 |  |
| 4 | plotname | 所在地区名 | varchar | 255 |  |
| 5 | time | 曝光时间 | date | 10 |  |
| 6 | content | 曝光详情 | varchar | 500 |  |
| 7 | image | 曝光图片 | varchar | 100 |  |
| 8 | state | 审核状态 | varchar | 10 |  |
| 9 | userid | 平台用户编号 | varchar | 35 | 外键 |
| 10 | adminid | 平台管理员编  号 | varchar | 35 | 外键 |

1. 装修疑惑表 doubt，如表 4-12 所示：

表 4-12 装修疑惑表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 疑惑标题 | varchar | 100 |  |
| 3 | content | 疑惑内容 | varchar | 500 |  |
| 4 | answernum | 回答数 | int | 10 |  |
| 5 | subtime | 提问时间 | varchar | 10 |  |
| 6 | userid | 平台用户 id | varchar | 35 | 外键 |

1. 疑惑解答表 answer，如表 4-13 所示：

表 4-13 疑惑解答表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | content | 解答内容 | varchar | 255 |  |
| 3 | date | 解答时间 | varchar | 10 |  |
| 4 | dznum | 点赞数 | int | 10 |  |
| 5 | managerid | 公司负责人编  号 | varchar | 35 | 外键 |

1. 资质证书表 zzzsphoto，如表 4-14 所示：

表 4-14 资质证书表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 图片名称 | varchar | 100 |  |
| 3 | address | 图片地址 | varchar | 100 |  |
| 4 | description | 图片说明 | varchar | 255 |  |
| 5 | companyid | 装修公司编号 | varchar | 35 | 外键 |

1. 装修招标信息表 zxzb，如表 4-15 所示：

表 4-15 装修招标信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | housetype | 房屋类型 | varchar | 35 |  |
| 3 | zbtype | 招标类型 | varchar | 10 |  |
| 4 | district | 所在区域 | varchar | 100 |  |
| 5 | plotname | 小区名称 | varchar | 255 |  |

续表 4-15

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 6 | budget | 初步预算 | double | 10 | 单位：万元 |
| 7 | area | 房屋面积 | int | 10 | 单位：平方米 |
| 8 | subtime | 发布时间 | date | 10 |  |
| 9 | zxtime | 预计装修时间 | date | 10 |  |
| 10 | lftime | 量房时间 | date | 10 |  |
| 11 | requirement | 招标要求 | varchar | 255 |  |
| 12 | userid | 平台用户编号 | varchar | 35 | 外键 |

1. 装修风格表 housestyle，如表 4-16 所示：

表 4-16 装修风格表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 11 | 主键 |
| 2 | name | 风格名称 | varchar | 100 |  |
| 3 | introduction | 风格介绍 | varchar | 255 |  |

1. 装修阶段表 stage，如表 4-17 所示：

表 4-17 装修阶段表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 阶段名 | varchar | 100 |  |
| 3 | introduction | 阶段介绍 | varchar | 255 |  |

1. 工地工程信息表 project，如表 4-18 所示：

表 4-18 工地工程信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 工程名称 | varchar | 100 |  |
| 3 | plotname | 工地小区 | varchar | 100 |  |
| 4 | address | 工地地址 | varchar | 255 |  |
| 5 | area | 工地面积 | int | 10 | 单位：平方米 |
| 6 | housetype | 房屋类型 | varchar | 10 |  |
| 7 | stageid | 装修阶段编号 | varchar | 35 | 外键 |
| 8 | managerid | 公司负责人编  号 | varchar | 35 | 外键 |

1. 工地图表 gdphoto，如表 4-19 所示：

表 4-19 工地图表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 图片名称 | varchar | 100 |  |
| 3 | address | 图片地址 | varchar | 100 |  |
| 4 | description | 图片说明 | varchar | 255 |  |
| 5 | projectid | 工地工程信息  编号 | varchar | 35 | 外键 |

1. 装修成果反馈图表 fkphoto，如表 4-20 所示：

表 4-20 装修成果反馈图表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 图片名称 | varchar | 100 |  |
| 3 | address | 图片地址 | varchar | 100 |  |
| 4 | description | 图片说明 | varchar | 255 |  |
| 5 | subtime | 发布时间 | date | 10 |  |
| 6 | msgnum | 留言数 | int | 10 |  |
| 7 | userid | 平台用户编号 | varchar | 35 | 外键 |

1. 反馈留言表 fkmessage，如表 4-21 所示

表 4-21 反馈留言表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | content | 留言内容 | varchar | 500 |  |
| 3 | time | 留言时间 | date | 10 |  |
| 4 | dznum | 点赞数 | int | 10 |  |
| 5 | userid | 平台用户编号 | varchar | 35 | 外键 |
| 6 | fkphotoid | 装修成果反馈  图编号 | varchar | 35 | 外键 |

1. 预约设计师申请表 yysjsapplication，如表 4-22 所示

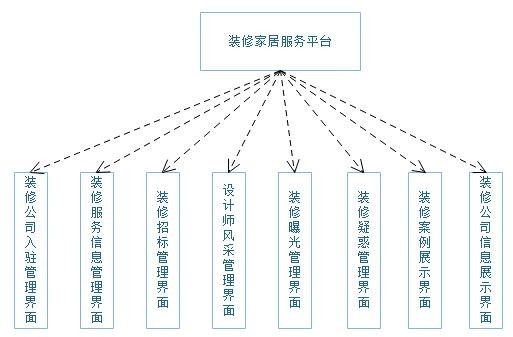
表 4-22 预约设计师申请表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 字段含义 | 类型 | 长度 | 备注 |
| 1 | id | 编号 | varchar | 35 | 主键 |
| 2 | name | 申请表名 | varchar | 100 |  |
| 3 | subtime | 申请时间 | date | 10 |  |
| 4 | housetype | 房屋类型 | varchar | 35 |  |
| 5 | housestyle | 装修风格 | varchar | 35 |  |
| 6 | actime | 受理时间 | date | 0 |  |
| 7 | plotname | 小区名称 | varchar | 100 |  |
| 8 | area | 房屋面积 | int | 10 | 单位：平方米 |
| 9 | state | 申请表状态 | varchar | 10 |  |
| 10 | userid | 平台用户编号 | varchar | 35 | 外键 |
| 11 | desinerid | 设计师编号 | varchar | 35 | 外键 |

# 界面设计

## 界面关系图

本系统主要界面包括装修公司入驻管理界面、装修服务信息管理界面、装修招标管理界面、设计师风采展示管理界面、装修曝光管理界面、装修疑惑管理界面、装修案例展示界面和装修公司信息展示界面，系统界面关系图如图 5-1 所示：



## 界面设计成果

图 5-1 系统界面关系图

* + 1. 装修公司入驻管理界面为平台管理员后台的一个功能界面，左侧为装修公司入驻管理的导航栏，右侧是管理界面的具体内容，数据以表格的方式进行展示，在数据上方设计一个搜索框，设计成果如图 5-2 所示：

图 5-2 装修公司入驻管理界面

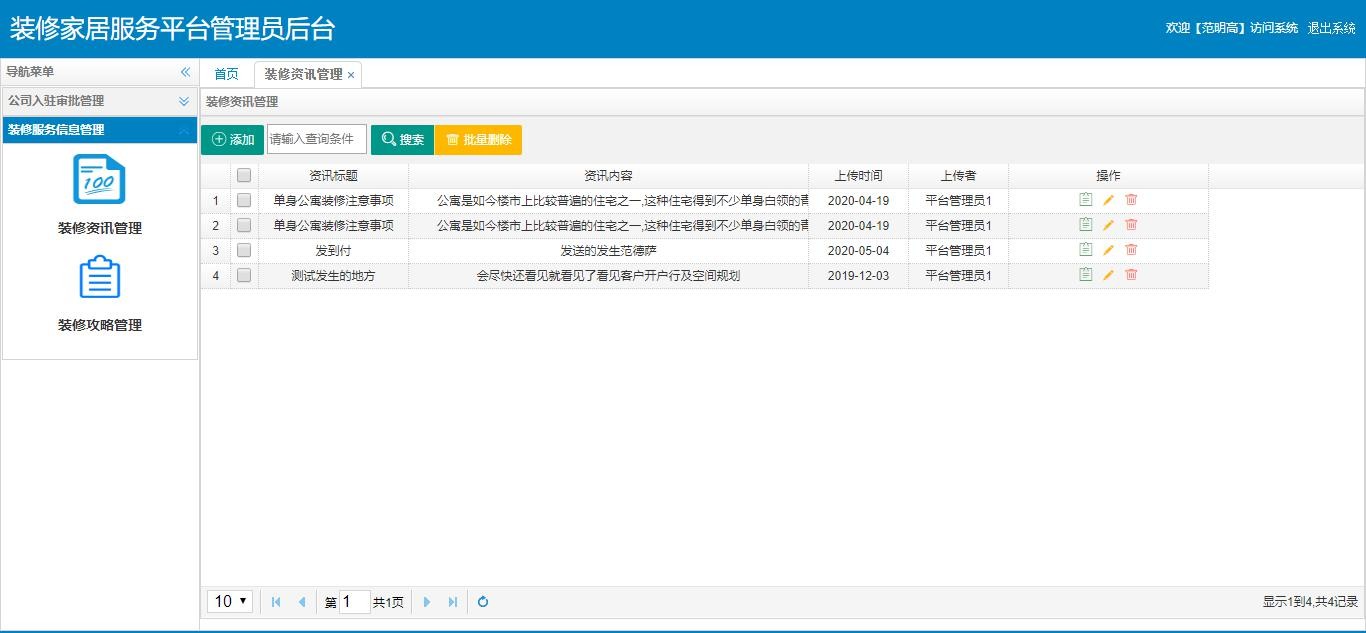
* + 1. 装修服务信息管理界面为平台管理员后台的一个功能界面，左侧为装修服务信息管理的导航栏，右侧是管理界面的具体内容，数据以表格的方式进行展示，在数据上方设计添加按钮、搜索框以及批量删除按钮，设计成果如图 5-3 所示：

图 5-3 装修服务信息管理界面

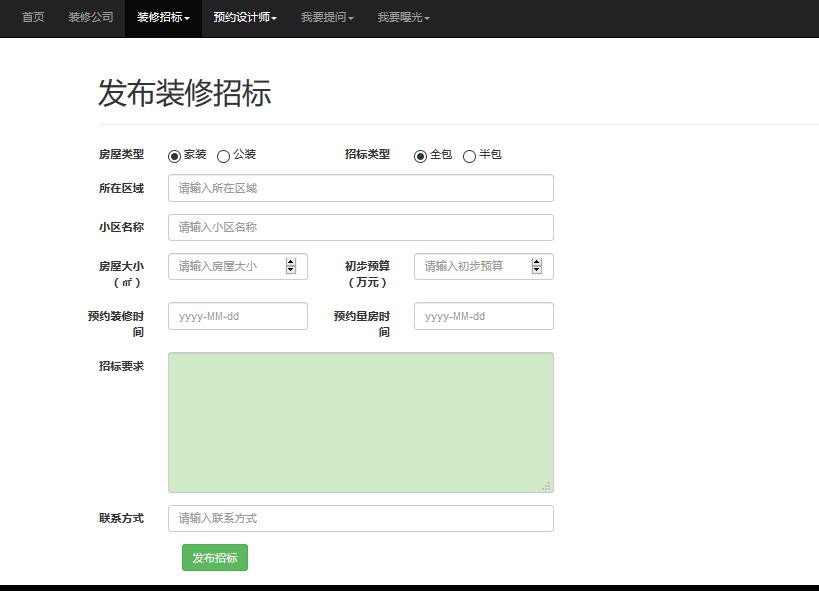
* + 1. 装修招标管理界面主要是以数据表单为主，顶部为前台页面的导航栏，中间为数据表单，设计成果如图 5-4 所示：

图 5-4 装修招标管理界面

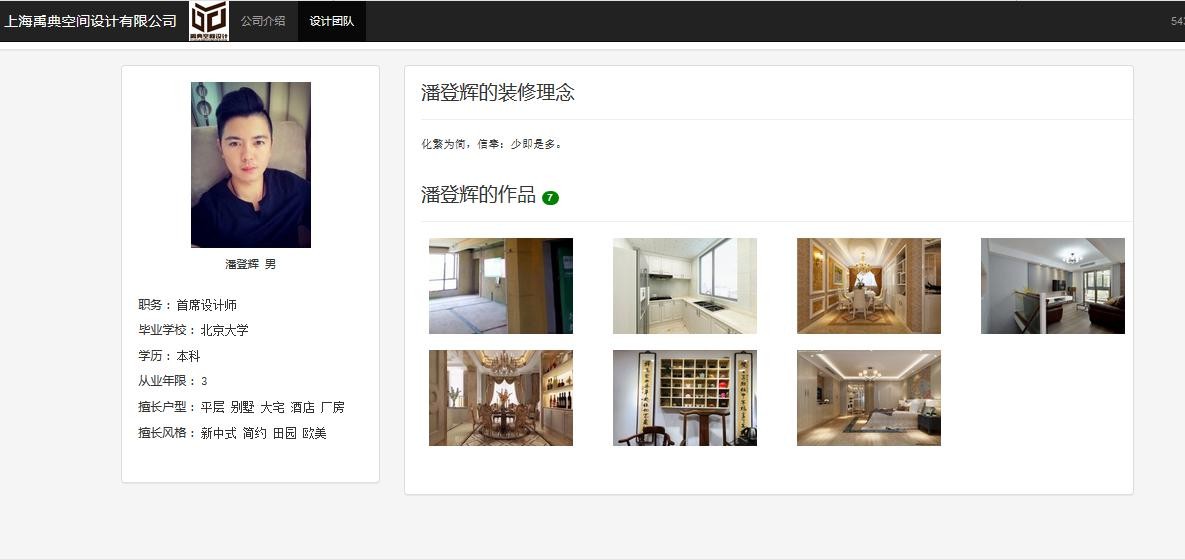
* + 1. 设计师风采管理界面顶部为页面的导航栏，中间部分为设计师信息的展示， 其中中间部分的右侧为设计师的照片和基本信息展示，左侧有设计师的装修理念和设计师的作品列表展示，设计成果如图 5-5 所示：

图 5-5 设计师风采管理界面

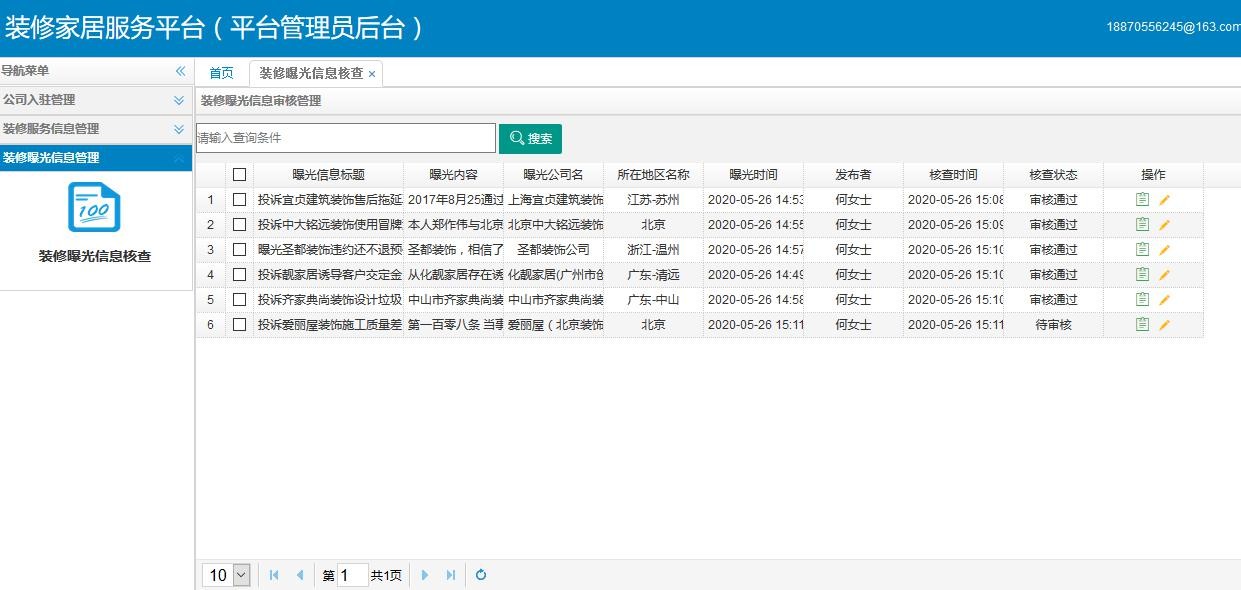
* + 1. 装修曝光管理界面为平台管理员后台的一个功能界面，左侧为装修曝光管理的导航栏，右侧是管理界面的具体内容，数据以表格的方式进行展示，在数据上方设计一个搜索框，设计成果如图 5-6 所示：

图 5-6 装修曝光管理界面

* + 1. 装修疑惑管理界面顶部为页面的导航栏，中间部分为装修疑惑信息、发布者的基本信息和疑惑的解答内容列表以及进行解答的装修公司名称和 logo，设计成果如图 5-7 所示：



图 5-7 装修疑惑管理界面

* + 1. 装修案例展示界面主要包括装修案例图片等信息的栅格化展示布局，每一个部分又包括一张案例图、案例名称和装修公司名称等信息，设计成果如图 5-8 所示：

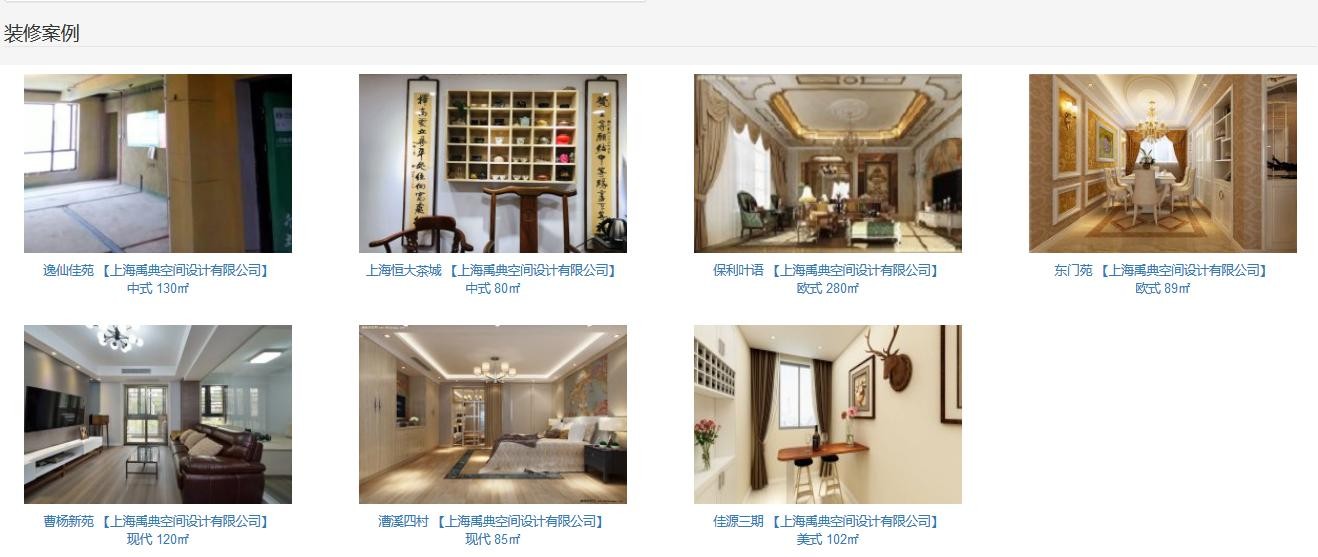


图 5-8 装修案例展示界面

* + 1. 装修公司信息展示界面主要就是装修公司信息列表列表块的展示，其中每个部分为某个装修公司的基本信息展示，以栅格化分为四部分，依次展示装修公司 logo、装修公司基本信息、装修保障金等和详细信息按钮，设计成果如图 5-9 所示：



图 5-9 装修公司信息展示界面

# 详细设计

## 系统主要功能模块介绍

本子系统的功能模块主要包括:

* + 1. 用户管理：

用户管理具体功能包括平台用户的用户注册、登录，以及平台管理员添加和删除公司负责人、设计师的用户操作。

* + 1. 装修交流平台管理：

装修交流平台管理是为用户提供的装修交流的平台，具体功能主要包括装修疑惑解答和装修投诉曝光。客户发布装修相关疑惑问题，用户或者装修人员进行解答交流，用户发布曝光投诉信息，平台管理员进行曝光信息核查，核查通过公开展示， 核查不通过撤销曝光等。

* + 1. 装修招标管理：

装修招标管理是提供给用户自主发布装修招标信息，供装修公司选择接受服务的功能，具体功能包括用户申请装修服务招标、取消装修服务招标、招标信息公开展示和公司选择接受招标等。

* + 1. 装修公司信息管理：

装修公司信息管理是公司入驻后对装修公司的信息进行管理和装修公司对装修案例等的管理。具体功能包括对公司基本信息的添加、修改、删除，基本信息的展示（包括基本信息简介展示、资质证书展示、擅长领域信息展示和装修保障条例展示等），公司上传公司的最新装修案例和装修案例的展示等。

* + 1. 设计师风采展示管理：

设计师风采展示管理主要是提供给设计师进行风采和信息展示的管理，具体功能包括上传信息及案例、设计师信息的展示和设计师装修案例展示等。

* + 1. 装修服务信息管理：

装修服务信息管理是平台管理员进行装修攻略的管理和装修资讯的管理，具体功能包括装修前、中、后期各方面的装修攻略的发布展示，资讯的添加、撤销、展示和修改等。

## 用户管理

* + 1. 功能结构设计

用户管理模块功能结构包图如图 6-1 所示：

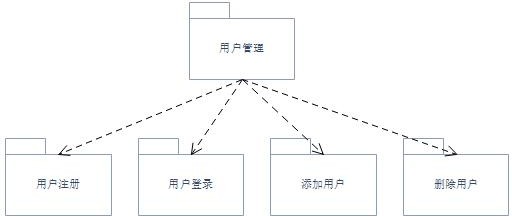


图 6-1 用户管理包图

用户管理具体功能包括平台用户的用户注册、登录，以及平台管理员添加和删除公司负责人、设计师的用户操作。

* + 1. 类图设计

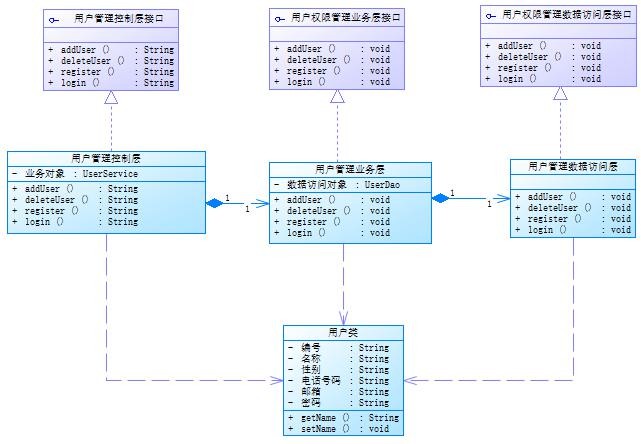
用户管理类图如图 6-2 所示：

图 6-2 用户管理类图

表 6-1 用户管理控制类描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 类代号（code） | UserControllerImpl | 类名称（中文） | 用户管理控制类 |
| 类功能描述： | 此类中包含用户管理各功能的方法，用于实现前台数据跟后台进行交互。 | | |
| 所在包名称： | controller.impl（控制层实现包） | | |
| 继承类： | KesunBusinessController | | |

续表 6-1

|  |  |  |  |
| --- | --- | --- | --- |
| 实现接口： | UserController | 实现接口： | UserController |
| 类属性 | 属性类型 | 属性名称 | 属性描述 |
| userService | UserService | 用户管理业  务层对象 | 业务层对象，用于调用业务层相应的方法 |
| 主要实现方法 | addUser()、deleteUser()、register()、login() | | |

* + 1. 时序图设计

用户管理时序图如图 6-3 所示：



图 6-3 用户管理时序图

平台管理员在用户管理界面请求添加用户信息，控制层收到请求后向业务层传输用户信息，业务层再把数据传输到数据访问层，数据访问层对数据库进行交互，向业务层返回添加结果，然后业务层将添加结果返回控制层，最后控制层将结果返回至页面，页面响应相应的添加结果。

## 装修交流平台管理

* + 1. 功能结构设计

装修交流平台管理模块功能结构包图如图 6-4 所示：

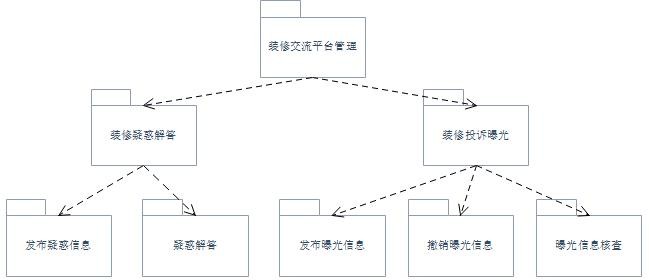


图 6-4 装修交流平台管理包图

装修交流平台管理是为用户提供的装修交流的平台，具体功能主要包括装修疑惑解答和装修投诉曝光。客户发布装修相关疑惑问题，公司负责人进行解答交流，用户发布曝光投诉信息，平台管理员进行曝光信息核查，核查通过公开展示，否则撤销曝光等。

* + 1. 类图设计

装修交流平台管理类图如图 6-5 所示：

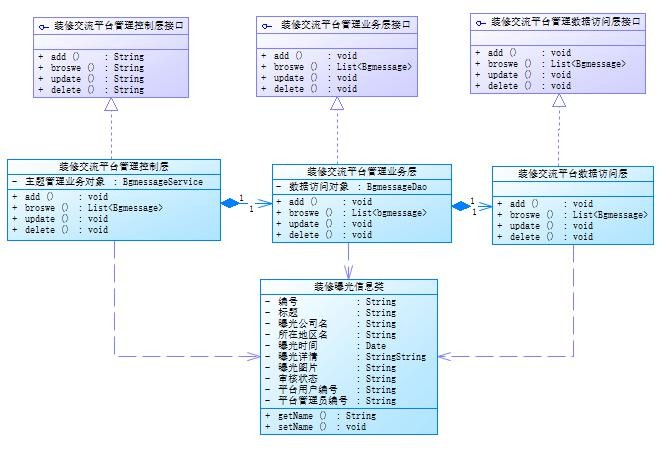


图 6-5 装修交流平台管理类图

表 6-2 装修交流平台管理控制类描述表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 类代号（code） | BgmessageControllerImpl | | 类名称（中文） | | 装修交流平台管理控制类 |
| 类功能描述： | 此类中包含装修交流平台管理各功能的方法，用于实现前台数据跟后台  进行交互。 | | | | |
| 所在包名称： | controller.impl（控制层实现包） | | | | |
| 继承类： | KesunBusinessController | | | | |
| 实现接口： | BgmessageController | | | | |
| 类属性 | 属性类型 | 属性名称 | | 属性描述 | |
| bgmessageService | BgmessageService | 装修交流平台管理业务  层对象 | | 业务层对象，用于调用业务层相应的方法 | |
| 主要实现方法 | add()、delete()、browse()、update() | | | | |

* + 1. 时序图设计

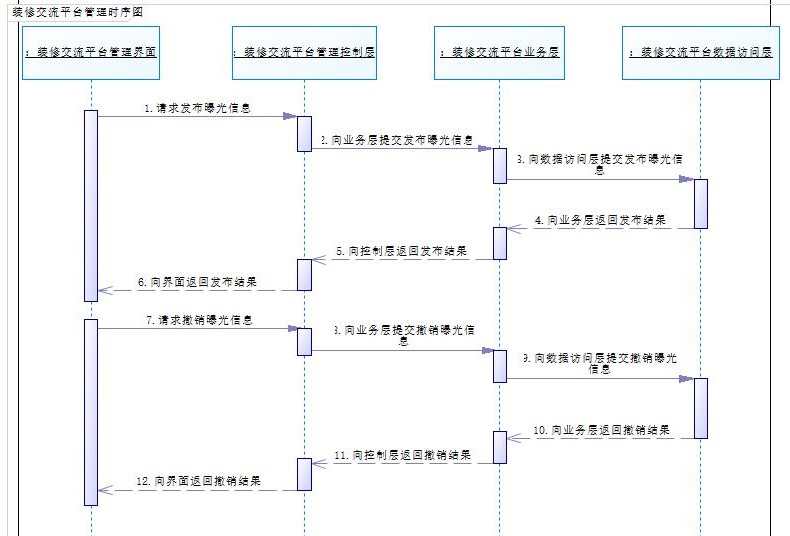
装修交流平台管理时序图如图 6-6 所示：

图 6-6 装修交流平台管理时序图

平台管理员在装修交流平台管理界面请求发布曝光信息，控制层收到请求后向业务层提交发布曝光信息，业务层再把数据传输到数据访问层，数据访问层对数据库进行交互，向业务层返回发布结果，然后业务层将发布结果返回控制层，最后控制层将结果返回至页面，页面响应相应的发布结果。

## 装修招标管理

* + 1. 功能结构设计

装修招标管理模块功能结构包图如图 6-7 所示：

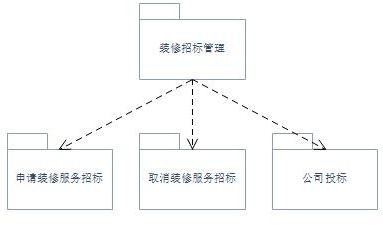


图 6-7 装修招标管理包图

装修招标管理是提供给用户自主发布装修招标信息，供装修公司选择接受服务的功能，具体功能包括用户申请装修服务招标、取消装修服务招标和公司选择投标等。

* + 1. 类图设计

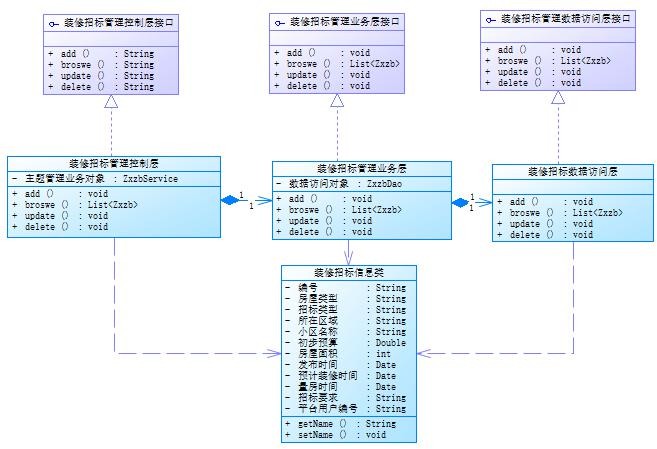
装修招标管理类图如图 6-8 所示：

图 6-8 装修招标管理类图

表 6-3 装修招标管理控制类描述表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 类代号（code） | ZxzbControllerImpl | | 类名称（中文） | | 装修招标管理控制类 |
| 类功能描述： | 此类中包含装修招标管理各功能的方法，用于实现前台数据跟后台进行交  互 | | | | |
| 所在包名称： | controller.impl（控制层实现包） | | | | |
| 继承类： | KesunBusinessController | | | | |
| 实现接口： | CompanyController | | | | |
| 类属性 | 属性类型 | 属性名称 | | 属性描述 | |
| zxzbService | ZxzbService | 装修招标管理业务层对  象 | | 业务层对象，用于调用业务层相应的方法 | |
| 主要实现方法 | add()、delete()、browse()、update() | | | | |

* + 1. 时序图设计

装修招标管理时序图如图 6-9 所示：

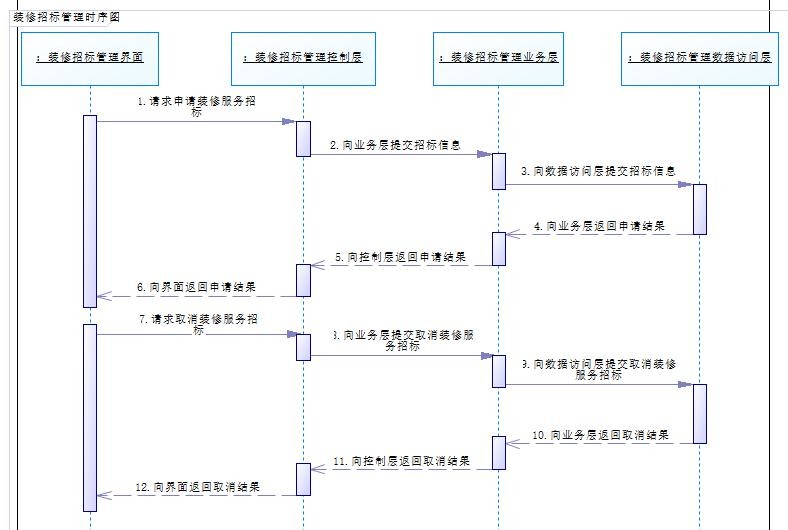


图 6-9 装修招标管理时序图

平台用户在装修招标管理界面请求申请装修服务招标，控制层收到请求后向业务层提交发布招标信息，业务层再把数据传输到数据访问层，数据访问层对数据库进行交互， 向业务层返回发布结果，然后业务层将发布结果返回控制层，最后控制层将结果返回至页面，页面响应相应的发布结果。

## 装修公司信息管理

* + 1. 功能结构设计

装修公司信息管理功能结构包图如图 6-10 所示：

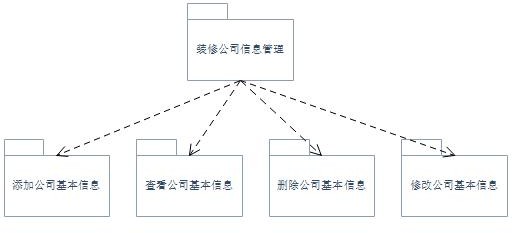


图 6-10 装修公司信息管理包图

装修公司信息管理是公司入驻后对装修公司的信息进行管理和装修公司对装修案例等的管理。具体功能包括对公司基本信息的添加、修改、删除和查看等。

* + 1. 类图设计

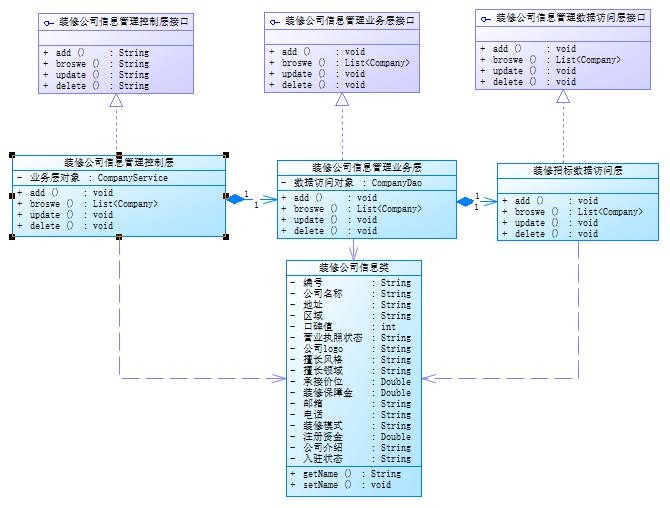
装修公司信息管理类图如图 6-11 所示：

图 6-11 装修公司信息管理类图

表 6-4 装修公司信息管理控制类描述表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 类代号（code） | CompanyControllerImpl | | 类名称（中文） | | 装修公司信息管理控制类 |
| 类功能描述： | 此类中包含装修公司信息管理各功能的方法，用于实现前台数据跟后台进  行交互 | | | | |
| 所在包名称： | controller.impl（控制层实现包） | | | | |
| 继承类： | KesunBusinessController | | | | |
| 实现接口： | CompanyController | | | | |
| 类属性 | 属性类型 | 属性名称 | | 属性描述 | |
| companyService | CompanyService | 装修公司信息管理业务  层对象 | | 业务层对象，用于调用业务层相应的方法 | |
| 主要实现方法 | add()、delete()、browse()、update() | | | | |

* + 1. 时序图设计

装修公司信息管理时序图如图 6-12 所示：

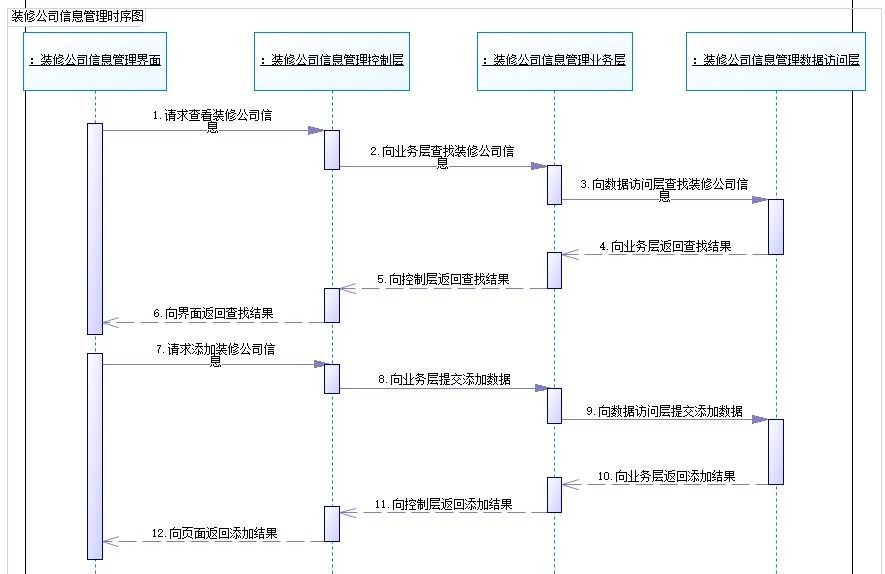


图 6-12 装修公司信息管理时序图

平台管理员在装修公司信息管理界面请求查看专修公司信息，控制层收到请求后向业务层查找装修公司信息，业务层再把数据传输到数据访问层，数据访问层对数据库进行交互，向业务层返回查找结果，然后业务层将查找结果返回控制层，最后控制层将结果返回至页面，页面响应相应的查找结果。

## 设计师风采展示管理

* + 1. 功能结构设计

设计师风采展示管理模块功能结构包图如图 6-13 所示：

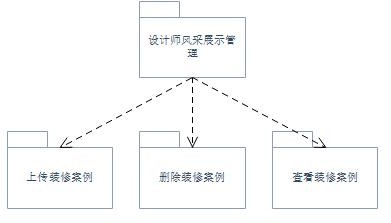


图 6-13 设计师风采展示管理包图

设计师风采展示管理主要是提供给设计师进行风采和信息展示的管理，具体功能包括上传信息及案例、删除装修案例和查看装修案例等。

* + 1. 类图设计

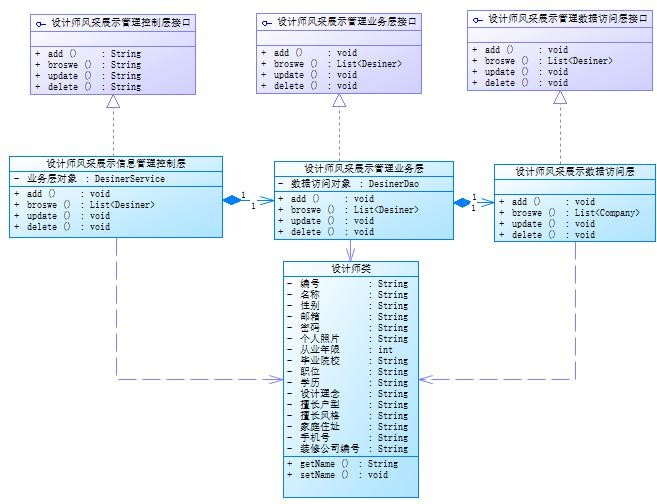
设计师风采展示管理类图如图 6-14 所示：

图 6-14 设计师风采展示管理类图

表 6-5 设计师风采展示管理控制类描述表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 类代号（code） | DesinerControllerImpl | | 类名称（中文） | | 设计师风采展示管理控制类 |
| 类功能描述： | 此类中包含设计师风采展示管理各功能的方法，用于实现前台数据跟后台  进行交互 | | | | |
| 所在包名称： | controller.impl（控制层实现包） | | | | |
| 继承类： | KesunBusinessController | | | | |
| 实现接口： | DesinerController | | | | |
| 类属性 | 属性类型 | 属性名称 | | 属性描述 | |
| desinerService | DesinerService | 设计师风采展示管理业  务层对象 | | 业务层对象，用于调用业务层相应的方法 | |
| 主要实现方法 | add()、delete()、browse()、update() | | | | |

* + 1. 时序图设计

设计师风采展示管理时序图如图 6-15 所示：

图 6-15 设计师风采展示管理时序图

设计师在设计师风采展示管理界面请求上传装修案例，控制层收到请求后向业务层提交上传装修案例，业务层再向数据访问层提交上传装修案例，数据访问层对数据库进行交互，向业务层返回上传结果，然后业务层将上传结果返回控制层，最后控制层将结果返回至页面，页面响应相应的上传结果。

## 装修服务信息管理

* + 1. 功能结构设计

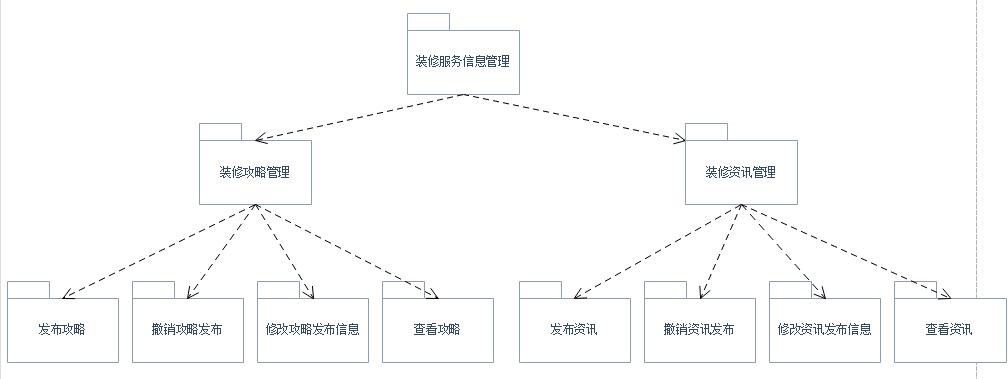
装修服务信息管理模块功能结构包图如图 6-16 所示：

图 6-16 装修服务信息管理包图

装修服务信息管理是平台管理员进行装修攻略的管理和装修资讯的管理，具体功能包括装修前、中、后期各方面的装修攻略的发布，资讯的添加、撤销、展示和修改等。

* + 1. 类图设计

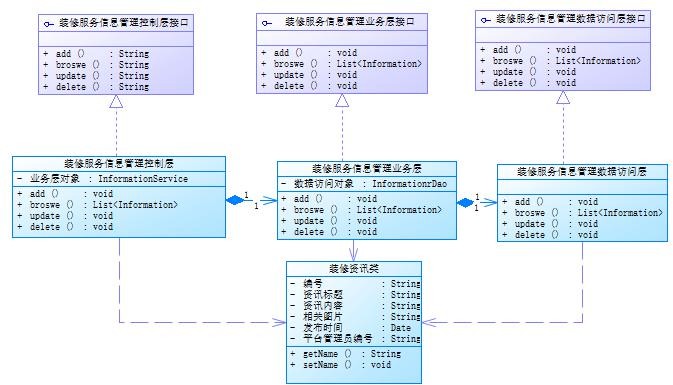
装修服务信息管理类图如图 6-17 所示：

图 6-17 装修服务信息管理类图

表 6-6 装修服务信息管理控制类描述表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 类代号（code） | InformationControllerImpl | | 类名称（中文） | | 装修服务信息管理控制类 |
| 类功能描述： | 此类中包含装修服务信息管理各功能的方法，用于实现前台数据跟后台进行  交互 | | | | |
| 所在包名称： | controller.impl（控制层实现包） | | | | |
| 继承类： | KesunBusinessController | | | | |
| 实现接口： | InformationController | | | | |
| 类属性 | 属性类型 | 属性名称 | | 属性描述 | |
| informationService | InformationService | 装修服务信息管理业务  层对象 | | 业务层对象，用于调用业务层相应的方法 | |
| 主要实现方法 | add()、delete()、browse()、update() | | | | |

* + 1. 时序图设计

装修服务信息管理时序图如图 6-18 所示：

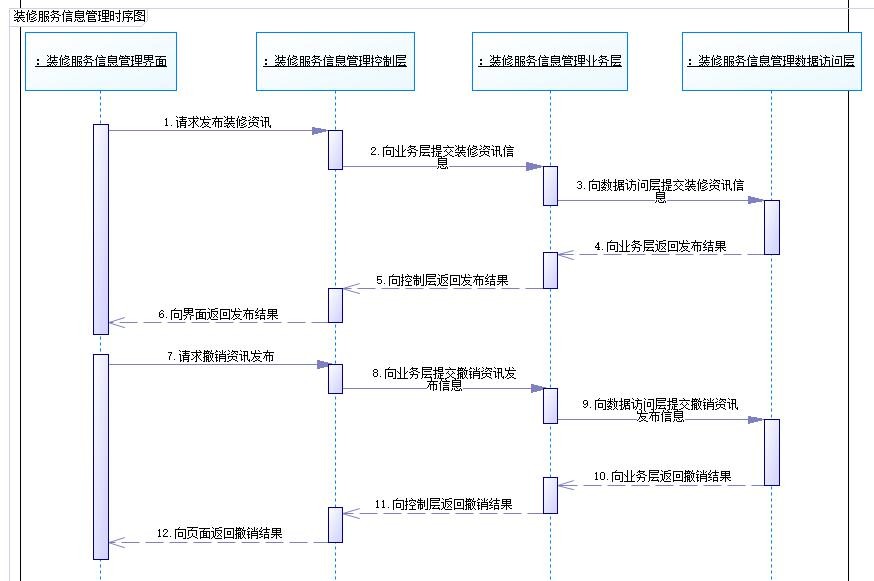


图 6-18 装修服务信息管理时序图

平台管理员在装修服务信息管理界面请求发布装修资讯，控制层收到请求后向业务层提交装修资讯信息，业务层再向数据访问层提交装修资讯信息，数据访问层对数据库进行交互，向业务层返回发布结果，然后业务层将发布结果返回控制层，最后控制层将结果返回至页面，页面响应相应的发布结果。

# 编码

## 代码实现与核心算法

* + 1. 单条数据的新增。首先利用 AbsSuperObject 类中初始化数据，然后再获取客户端的参数信息，将这些信息封装到实体类对象中，执行业务层添加数据的方法，最终把数据传到持久层将数据添加到数据库中完成单条数据的新增。其中控制层实现类中对应的代码如下：

@RequestMapping(value = "/add", method = RequestMethod.POST) @ResponseBody

public Object add(@RequestBody(required = true) Map<String, Object> p) { KesunReturn back = new KesunReturn();//返回操作

AbsSuperObject temp = initParameter(p);

if (temp == null) { back.setCode("0");

back.setMessage("抱歉，系统没有获取到需要数据！"); back.setObj(null);

return back;

}

AbsSuperObject model = updateModel(temp);//获取客户端参数信息if (model == null) {

back.setCode("0");

back.setMessage("抱歉，您提供的数据存在问题，系统在转换成目标数据时失败！"); back.setObj(null);

return back;

}

back=checkServiceObject(getService()); if (back!=null) return back;

back=new KesunReturn(); getService().setModel(model);

if (getService().isAdd() == false) { back.setCode("0");

back.setMessage("抱歉，系统设置不能新增【" + model.getName() + "】的操作,数据是否重复，

请核实！");

back.setObj(null); return back;

}

return getService().add();

}

* + 1. 单条数据修改。首先获取 AbsSuperObject 类中初始化数据，然后再获取客户端的参数信息，将这些信息封装到实体类对象中，执行业务层修改数据的方法，最终把数据传到持久层将修改更新的数据添加到数据库中，完成单条数据修改。其中控制层实现类中对应的代码如下：

@RequestMapping(value = "/edit", method = RequestMethod.POST) @ResponseBody

public Object edit(@RequestBody(required = true) Map<String, Object> p) { KesunReturn back = new KesunReturn();//返回操作

AbsSuperObject temp = initParameter(p); if (temp == null) {

back.setCode("0");

back.setMessage("抱歉，系统没有获取到的修改数据！"); back.setObj(null);

return back;

}

AbsSuperObject model = updateModel(temp);//获取客户端参数信息if (model == null) {

back.setCode("0");

back.setMessage("抱歉，您提供的数据存在问题，系统在转换成目标数据时失败，无法继续完 成修改操作！");

back.setObj(null); return back;

}

back=checkServiceObject(getService()); if (back!=null) return back;

back=new KesunReturn(); getService().setModel(model);

if (getService().isEdit() == false) { back.setCode("0");

back.setMessage("抱歉，系统设置要求取消了【" + model.getName() + "】修改操作,数据是否重

复，请核实！");

back.setObj(null);

return back;

}

return getService().edit();

}

* + 1. 分页查找数据。首先获取客户端的查询参数，以 Map 集合的方式存放，再合并数据过滤条件，然后执行业务层分页查找的方法，把查询参数传递到持久层与数据库进行交互，查询出相应的数据，最终返回给控制层用于渲染页面，显示查询数据。其中控制层实现类中对应的代码如下：

@RequestMapping(value = "/findByPage", method = RequestMethod.POST) @ResponseBody

public Object findByPage(@RequestBody(required = true) Map<String, Object> p) { Map<String, Object> values =getConditionParam(p);//获取查询参数

/\*合并数据过滤条件\*/

if (setFindFilter(p) != null) values.putAll(setFindFilter(p));

KesunSuperService bll = getService();//获取业务对象SearchViewParam pp = ControlTool.GetViewParam(p);//获取查询参数ReturnBeanInPower rbi = new ReturnBeanInPower(); rbi.setHasPower(setPowerFilter());

KesunReturn back=checkFindCondition(bll,values); if (back!=null) return back;

return getService().findByPage(values, pp.pageNumber, pp.rowsCount);

}

* + 1. 文件上传。首先获取客户端传来的文件参数，然后调用文件上传的方法，利用Http 网络协议完成对文件数据的上传，最终返回上传结果，完成对文件的上传。其中控制层实现类中对应的代码如下：

@RequestMapping(value = "/uploadfile", method = RequestMethod.POST) @ResponseBody

public Object uploadFile(

@RequestParam("file") CommonsMultipartFile files[], HttpServletRequest request,

HttpServletResponse response) throws IOException { KesunReturn back=new KesunReturn();

if (request.getSession().getAttribute("user")==

null || request.getSession().getAttribute("user")

instanceof User ==false)

{

back.setCode("0");

back.setMessage("用户没有登录，不允许文件上传操作！"); back.setObj(null);

return back;

}

List<KesunAttachment> lTemp=doUploadFile(files,request,response); if (lTemp==null || lTemp.size()==0)

{

back.setMessage("没有实现任何文件上传！");

}

else

back.setMessage("文件已经上传完完成，共完成了【"+lTemp.size()+"】文件上传！"); back.setObj(lTemp);

return back;

}

## 代码优化分析

针对多个模块功能可能同时使用到通一段代码的初始化等工作，例如多个页面可能同时需要用到 Bootstrap 的资源库，有多个页面可能都使用 EasyUI 或者 LayUI 相关的资源依赖，于是可以将相关的资源用页面单独管理，形成多种初始化页面加载的“资源库”。另外，利用 MD5 对重要说句进行加密，增强系统的安全性。相关代码优化如下：

1. EasyUI 初始化资源

<%@ page

contentType="text/html;charset=UTF-8" language="java" %>

<%@taglib prefix="shiro" uri="<http://shiro.apache.org/tags>" %>

<%

String path = request.getContextPath(); String basePath = request.getScheme()+"://"

+request.getServerName()+":"+request.getServerPort()+path;

%>

<html>

<head>

<title>装修家居服务平台 EasyUI 初始化资源</title>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

<meta name="viewport" content="width=device-width; initial-scale=1.0">

<linkhref="<%=basePath%>/plugins/easyui/themes/icon.css" rel="stylesheet" />

<link href="<%= basePath %>/css/project.css" rel="stylesheet" />

<script src="<%= basePath %>/script/common/jquery-1.8.2.min.js"></script>

<script src="<%= basePath %>/plugins/easyui/jquery.easyui.min.js"></script>

<script src="<%= basePath %>/script/common/clientStorage.js"></script>

<script src="<%= basePath %>/script/common/public.js"></script>

<script src="<%= basePath %>/script/common/project.js"></script>

<script src="<%= basePath %>/script/business/myproject.js"></script>

<style type="text/css">

.mask {

position:absolute; top: 0px;

filter: alpha(opacity=60); background-color: #777; z-index: 1002; left: 0px;

opacity:0.5; -moz-opacity:0.5; width: 100%;

display:none;

vertical-align: middle; text-align: center;

}

</style>

</head>

<body>

<div id="mask" class="mask">

<img src="<%=basePath%>/images/loading.gif"/>

</div>

</body>

</html>

1. MD5 数据加密

public static String GetPwd(String pwd) { try {

// 创建加密对象

MessageDigest digest = MessageDigest.getInstance("md5");

String afterPwd=String.format("%sw#p#h%s", pwd,pwd);

// 调用加密对象的方法，加密的动作已经完成

byte[] bs = digest.digest(afterPwd.getBytes());

// 接下来，要对加密后的结果，进行优化，按照 mysql 的优化思路走

// mysql 的优化思路：

// 第一步，将数据全部转换成正数：

String hexString = ""; for (byte b : bs) {

// 第一步，将数据全部转换成正数：

int temp = b & 255;

// 第二步，将所有的数据转换成 16 进制的形式

if (temp < 16 && temp >= 0) {

// 手动补上一个“0”

hexString = hexString + "0" + Integer.toHexString(temp);

} else {

hexString = hexString + Integer.toHexString(temp);

}

}

return hexString;

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block e.printStackTrace();

}

return "";

}

# 测试

## 测试方案设计

* + 1. 测试策略

测试策略描绘的是测试工作的方法和实现目标。本系统的测试工作主要按照以下策略进行：单元测试、黑盒测试、白盒测试以及集成测试。其中单元测试，主要对系统每个类和具体函数的功能是否能正确运行进行测试；黑盒测试，又称为功能测试，主要是对外界的各类输入系统是否反馈正确进行测试；白盒测试与黑盒测试结合在一起进行，争取实现系统功能点范围 100%覆盖；集成测试是在发布前项目的最后一轮测试，其目的是保证项目的总体功能以及性能达到要求。

* + 1. 测试进度安排

本系统的测试进度安排如表 8-1 所示：

表 8-1 测试进度安排表

|  |  |  |
| --- | --- | --- |
| 测试阶段 | 测试任务 | 用时 |
| 单元测试 | 对类的运行功能进行测试，该阶段测试随着项目的开  发同步进行。 | 20 天 |
| 黑盒测试 | 对系统前后台操作界面进行统一的黑盒功能测试，收  集测试结果和系统反馈 | 3 天 |
| 白盒测试 | 依据系统各模块的的优先级先后进行白盒测试，收集  系统运行信息，确保代码运行不出现逻辑问题。 | 2 天 |
| 集成测试 | 对系统所有的功能进行最后测试，兼顾系统功能、性  能和交互性。 | 5 天 |

* + 1. 测试资源

系统测试需要指定特定的软件环境，系统测试软件环境资源如表 8-2 所示：

表 8-2 软件环境资源表

|  |  |
| --- | --- |
| 资源类型 | 软件 |
| 应用数据库 | Mysql 5.0 |
| 应用服务器 | Tomcat 8.0 |
| 框架版本 | Spring 4.3.10、mybatis3.2.6 |
| 富文本编辑器 | IDEA 2017.2+Maven 3.5.4 |
| JDK | JDK 1.8 |
| 浏览器 | FireFox 浏览器 |
| 浏览器 | Chrome 浏览器 |
| 浏览器 | Opera 浏览器 |

* + 1. 关键测试点

1. 平台用户的登录注册和添加删除用户的管理。
2. 装修交流平台中装修疑惑的管理和装修曝光发布、审核的管理。
3. 装修招标的发布管理和公司投标操作的管理。
4. 装修公司信息的增删改查操作以及上传相应材料的管理。
5. 设计师上传装修案例的管理。
6. 装修资讯和攻略信息的增删改查等操作的管理

## 测试用例构建

* + 1. 测试用例编写约定
       1. 对有不同角色的功能点，需要多次更换角色进行测试；
       2. 对输入框输入信息的判定应该遵循设计规范，注意系统边界，超出应给予提示；
       3. 测试用例的设计必须科学与规范，且每个测试用例被正确执行；
    2. 测试用例设计

基于本系统的特点，测试用例的设计主要从程序功能流程图和各个功能点的用例图用例表下手，制定出测试用例的执行步骤提前预测预期结果，再采用黑盒测试与白盒测试相结合的分析方法，对系统的输入信息和输出信息进行检验和控制，对关键功能的代码执行过程进行详细分析。本系统主要用到的测试用例设计方法有：

1. 边界值分析法

边界值分析法作为对等价类划分法的补充，主要负责对系统边界进行测试，比如系统容易出现边界值的地方，例如信息管理的输入框，咨询表单的输入框等，要进行特殊值和输入数据长度的检验，测试边界。

1. 等价类划分法

等价类划分法是将程序所有可能的输入数据划分成若干个等价类。然后从每个部分中选取具有代表性的数据当作测试用例进行合理的分类，这样可以确定输入值的精确范围和覆盖大部分的输入情况。

* + 1. 关键测试用例

针对系统的所有测试用例，本文摘取了部分关键测试用例如下：

1. 装修曝光管理测试用例表如表 8-3 所示：

表 8-3 装修曝光管理测试用例表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试目的 | | 测试装修曝光管理功能 | |
| 预置条件 | | 平台用户和平台管理员 | |
| 序号 | 测试步骤 | | 预期结果 |
| 1 | 平台用户点击装修曝光导航栏 | | 进入到填写曝光信息页面 |
| 2 | 平台用户填写相关曝光信息，点击提交按钮 | | 系统提示曝光信息上传成功 |

续表 8-3

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试步骤 | | 预期结果 |
| 3 | 平台管理员点击左侧导航栏，选择装修曝光  管理 | | 显示所有装修曝光信息列表 |
| 4 | 平台管理员点击需要审核的曝光信息 | | 弹框显示曝光信息的具体内容 |
| 5 | 平台管理员勾选审核通过，点击提交 | | 系统提示审批完成 |
| 6 | 平台用户点击装修曝光信息导航栏 | | 显示所有已审核通过的曝光信息 |
| 测试状态 | | 上述操作均达到“测试”要求 | |
| 测试结果 | | 经过测试，该模块功能正常，能够对进行正常的曝光信息的发布和曝光审核等  操作，可以对错误的操作进行阻止，并提示错误信息，交互性良好，达到了预 期目标。 | |

1. 装修招标管理测试用例表如表 8-4 所示：

表 8-4 装修招标管理测试用例表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试目的 | | 测试装修招标管理功能 | |
| 预置条件 | | 平台用户和公司负责人登录平台 | |
| 序号 | 测试步骤 | | 预期结果 |
| 1 | 平台用户点击装修招标导航栏 | | 进入到填写招标信息页面 |
| 2 | 平台用户填写相关招标信息，点击招标按  钮 | | 系统提示装修招标信息发布成功 |
| 3 | 公司负责人点击左侧导航栏中装修招标管  理链接，选择装修投标信息 | | 显示所有用户的招标信息列表 |
| 4 | 公司负责人点击需要投标的招标信息 | | 弹框显示招标信息的具体内容 |
| 5 | 公司负责人点击投标按钮 | | 系统提示投标成功 |
| 6 | 公司负责人查看左侧导航栏中已投标信息 | | 显示所有该公司的已投标信息 |
| 测试状态 | | 上述操作均达到“测试”要求 | |
| 测试结果 | | 经过测试，该模块功能正常，能够对进行正常的招标发布和公司投标操作，可  以对错误的操作进行阻止，并提示错误信息，交互性良好，达到了预期目标。 | |

1. 装修公司信息管理测试用例表如表 8-5 所示：

表 8-5 装修公司信息管理测试用例表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试目的 | | 测试装修公司信息管理功能 | |
| 预置条件 | | 平台管理员登录平台 | |
| 序号 | 测试步骤 | | 预期结果 |
| 1 | 平台管理员点击装修公司信息管理导航栏 | | 显示所有平台内装修公司信息列表 |
| 2 | 点击添加公司信息按钮 | | 弹窗显示填写装修公司信息页面 |

表 8-5

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试步骤 | | 预期结果 |
| 3 | 填写装修公司基本信息，点击添加按钮 | | 系统提示添加成功，关闭窗口，刷新公司  信息列表，添加的公司信息显示 |
| 4 | 选择需要查看的公司信息，点击查看按钮 | | 弹窗显示装修公司信息 |
| 5 | 选择需要修改的公司信息，点击编辑按钮 | | 弹窗显示该装修公司信息 |
| 6 | 填写需要修改的公司信息，点击提交按钮 | | 系统提示修改信息成功 |
| 7 | 选择需要删除的公司信息，点击删除按钮 | | 弹窗提示是否删除所选中的信息 |
| 8 | 点击确定按钮 | | 系统提示删除信息成功 |
| 测试状态 | | 上述操作均达到“测试”要求 | |
| 测试结果 | | 经过测试，该模块功能正常，能够对进行正常的装修公司信息的增删改查操作，  可以对错误的操作进行阻止，并提示错误信息，交互性良好，达到了预期目标。 | |

* + 1. 测试用例维护

1. 及时增加新的测试用例

可能因为需求的变化、用例的缺漏或者系统上线后可能发现不足等，导致原本拥有的测试用例没有完全覆盖整个系统的需求，因此在原有基础上应增加新的测试用例。

1. 及时删除过时的测试用例

需求变更会有可能导致原有的部分测试用例变的可能不再适合新的需求要求。例如， 删除了某个功能之后，该功能的测试用例也就不再需要。因此随着需求的一次次变化，都应该删除那些无用的测试用例。

1. 及时删除冗余的测试用例

在进行测试用例的设计过程中，可能会出现多个用例存在的内容和操作相同，这就使得回归测试效率大大降低。因此应该经常性地整理测试用例集，及时删除那些重复冗余的测试用例。

# 总结与展望

## 设计工作总结

在开始项目前，对此平台系统做了大量的需求分析，通过查阅资料以及老师的指导确定了平台系统应该如何开始设计。另外，平台页面结构的展现以及页面颜色与文字的搭配都是系统建设的一个比较重要的方面，故此项目借鉴了多个装修服务门户网站的设计页 面，混合搭配而成。在进行需求分析时，由于没有深入的去了解各个用户应该具有哪些功能，导致需求分析出现遗漏。经过指导后，弥补了之前未分析到的需求。

在进行总体设计阶段，确定了平台系统采用 Spring+SpringMVC+Mybatis 框架以及Maven 项目对象模型进行搭建。数据信息采用 MySQL 数据库进行存储，以及将平台后台系统的功能模块清晰地展现。此系统主要包括用户管理，装修招标管理等模块功能。进行数据库设计时，确定了各个实体之间的关系，确定了各个实体的各个属性，并将它们用图或表的形式展现出来。

在详细设计阶段，主要是对平台系统进行模块化的设计与分析，将平台系统中某个模块具体实现哪些功能，实现功能流程等用 UML 建模方式进行展现说明。详细设计是需求分析与编码阶段之间作为一个交互桥梁作用的重要阶段。

在编码实现阶段，在前面各阶段的基础上，这个阶段主要就是将前期的设计成果借助于代码，将系统以交互性页面的形式进行展现。本平台的编码实现主要是在“统一开发平台”的基础上进行的，该平台封装好了一些常用的方法等。在编码过程中，严格遵循了 MVC 三层架构的开发顺序，借助于 SSM 框架注解和各种配置文件，使整个开发过程有条不紊。另外，前端页面主要使用的是 Bootstrap、EasyUI 和 LayUI 等框架技术。在整个编码过程中，出现了各种各样的问题，通过不断的调试和查找资料，问题也都基本上解决了。

项目完成之后，收尾便是测试阶段。测试阶段主要是确保系统正常运行以及优化用户交互性等问题。测试的时候，发现页面上有些图片未显示，原因是路径不对，于是修改了系统的默认路径，使得图片可以完全的显示出来。还发现了后台传值出现问题，在查阅相关资料后，问题也基本上解决了。当然测试中还有一些小问题，都已修改完成，不需要进行详细的说明，比如说多余的弹窗。

总之，系统需求分析与设计阶段做了明确详尽的系统规划，划分系统模块，确定功能符合需求调研记录以及系统各功能的具体实现设计；编码阶段体现了SSM 框架和Bootstrap 等前端框架的应用与实现。页面的颜色搭配以及字体的选择还有待处理，才可以使页面版面更加美观以及清新。此平台系统需要不断的去更新信息，从而使平台系统更加得完善和实用。

## 未来工作展望

本平台目前只是初步实现了一些基本的功能，从总体上看，系统功能并不是特别完善， 由于时间和能力有限，当下只能完成这些功能模块。随着技术的提升和项目经验的积累， 系统还能进行一些完善和优化。

装修服务最重要的就是信息公开和服务标准化。在功能方面，可以增加公司入驻、设计师预约、装修报价预算、装修成果反馈交流和装修完成后的故障维修申请等功能。增加公司入驻功能，能让更多装修公司入驻平台，平台进行统一管理，有利于服务的标准化和信息的对称性。设计师预约是在装修招标基础上的一个拓展，用户可以自主选择预约心怡的设计师进行交流设计，更加符合用户的需要。装修报价预算主要是在获取多种数据的基础上，进行相关算法的分析，用户可以选择不同的城市、户型和装修风格等，系统自动计算出装修的预算价格，让用户心里有底。装修成果反馈是完善装修交流平台的功能，增加用户之间的公开交流。故障维修申请功能使平台的装修服务从预约到后期保障整个过程更加得完整。

在页面设计方面和用户的实际交互性方面，平台需要进一步优化页面的布局和色调设计，使整个系统页面更加得和谐美观，一些数据的校验和弹窗的设计等方面需要进行完善和优化。另外在平台的实际运行中，经常会出现卡顿或者后台数据加载不流畅的问题，后期需要进行优化，提高用户的交互性。

在系统维护方面，由于平台开发时间的限制，在编码阶段并没有考虑到后期系统的维护工作，系统可以在多方面存在一些安全隐患，在后期需要进一步排查和修复。还有一些数据的维护，需要学习相关的技术和知识，一步一步进行实现。另外，在开发过程中，也没有考虑到系统的多用户操作并发的实际需求，在后期应该将其进行全面的考虑。

总而言之，平台系统目前还有很大的完善空间，未来需要借助更多的技术加以实现。在一步一步完善后，能让平台能够正式上线部署运营，能够做为服务用户的实用平台是设计本平台的最终愿望。

# 谢 辞

# 参考文献

1. 李长腾. 基于 WEB 的网上装修服务系统的设计与实现[D]. 2015.
2. 程亮,杨晨光,张花普. 装修联盟网站的设计与实现[D]. 2017.
3. 张继涛. 基于服务设计理念的 O2O 家装服务平台的设计研究[D]. 2018.
4. 夏炜. A 建筑装饰公司集成化管理体系建立及应用研究[D] 2018.
5. 牛思先. 软件工程思想在管理信息系统开发中的应用探讨[D].信息系统工程.2017.
6. 苗洁. 基于 EasyUI 框架与 Spring MVC 框架的权限管理系统的设计与实现[D].电脑知识与技术.2015.
7. 荣艳冬. 关于 Mybatis 持久层框架的应用研究[D].信息安全与技术. 2015.
8. 王天顺, 程杰, 张志文.MyBatis 在《 Web 中间件技术》 课程中的应用[J]. 教育教学论坛,2019(38):204-205.
9. 高超.基于软件工程理论的信息系统监理的研究与实践[J].中国信息化,2019(02):42-43.
10. 李 德 华 . 基 于 SSM 技 术 的 在 线 考 试 系 统 的 研 究 与 实 现 [J]. 信 息 与 电 脑 ( 理 论版),2019,31(17):51-53
11. 李洋.SSM 框架在 Web 应用开发中的设计与实现[J].计算机技术与发展,2016,26(12):190-194.
12. 李海波. 基于 SSM 的质监系统网站设计开发与实现[D]. 吉林大学.
13. 杨震, 范雪松, 夏平平. 基于 SSM 框架的公文流转系统的设计与实现[J]. 现代信息科技,2019,3(24):92-94.
14. 李鹏楼. 高性能门户网站及其信息发布系统的设计与实现[D].北京邮电大学,2019.
15. 尹维伟.探析计算机软件 Java 编程特点及技术[J].电脑编程技巧与维护,2020(03):38-39+52.
16. 朱运乔. 基于 SpringBoot+SSM 框架的 Web 应用系统搭建与实现[J]. 电脑编程技巧与维护,2019(10):23-25.
17. 曹阳.软件工程 Web 技术开发平台[J].电子技术与软件工程,2019,0(5):34-34.
18. 朱丹丹.[浅析 JAVA 语言的开发平台及 J2EE 编程技术](http://www.cnki.com.cn/Article/CJFDTOTAL-HBYD201511072.htm)[J];信息通信;2015 年 11 期.
19. 刘颖.软件工程技术在系统软件开发中的运用[J].电子技术与软件工程,2019,0(5):31-31.
20. 徐盼.[我国社会智库有效性的评价和提升策略研究](http://cdmd.cnki.com.cn/Article/CDMD-10358-1017295596.htm)[J];中国科学技术大学;2017 年.
21. 熊立勇.[国际比较视域下中国特色新型智库建设问题研究](http://cdmd.cnki.com.cn/Article/CDMD-10358-1017295550.htm)[J];中国科学技术大学;2017 年.
22. 张海藩, 吕云翔. 《软件工程(第 4 版）》［M］. 北京：人民邮电出版社，2013. 39-42.
23. 杨开振, 周吉文. 《Java EE E 联网轻量级框架整合开发》［M］. 北京：电子工业出版社， 2017.224-255.
24. 王雨竹. 《MySQL 入门经典》［M］. 机械工业出版社,2014.158-232.
25. Mingyi Qi. Analysis on Humanized Appeal of UI Interface Design and Teaching Guiding Significance 2018.
26. S. Niepostyn. Analysis and specification of consistency rules for UML diagrams; Warsaw Univ. of Technology (Poland) 2019.
27. Cassineri Stefano,Duff Jonathan,Curioni Michele,Banks Andrew,Scenini Fabio. CRUD deposition in accelerated high-temperature water: Investigation on the effect of substrate material and water chemistry[J]. North-Holland,2020,529(C).
28. Abdelhak Mesbah,Jean-Louis Lanet,Mohamed Mezghiche. Reverse engineering a Java Card memory management algorithm[J]. Computers & Security,2017,66.
29. [Ogihara](http://search.dangdang.com/?key2=Ogihara&medium=01&category_path=01.00.00.00.00.00), [Mitsunori](http://search.dangdang.com/?key2=Mitsunori&medium=01&category_path=01.00.00.00.00.00).《Fundamentals of JAVA Programming》[M] Springer. 2019.8.
30. Cassineri Stefano,Duff Jonathan,Curioni Michele,Banks Andrew,Scenini Fabio. CRUD deposition in accelerated high-temperature water: Investigation on the effect of substrate material and water chemistry[J]. North-Holland,2020,529(C).

# 附录 A 外文翻译—原文部分

原文出处： Suryotrisongko H , Jayanto D P , Tjahyanto A . Design and Development of Backend Application for Public Complaint Systems Using Microservice Spring Boot[J]. Procedia Computer Science, 2017, 124:736-743.

### Introduction

E-government consists of the use of electronic communication technologies such as internet, for improving citizens access to public services. Implementation of E-Government provides the efficiency and speed of management on the reporting administration system, as well as the transparency of processes that occur in government administration. Through it, an aspect emerged which is called good governance. However, the implementation of E-government in Indonesia faces many issues such as expense issues for developing and operating e-government applications, technical issues such as security issues, privacy, and system update and human resource issues in which there is lack of capability to manage it. Cloud technology becomes one of the alternative answers to tackle those issues. This model allows consumers to use applications that exist in the cloud online through providers that can be accessed in various kinds of devices without worrying about those issues.

The use of microservice technology can give some advantages to the e-government system in the cloud. The concept of modularity in microservice allows the management of services that exist separately in an application. The impact to a development of a particular service is that it will not interfere with other services. Capacity building of a ervice can be distinguished among other services so that the resource is used appropriately. Also, the development of services can be developed with different programming languages .

### Related works

Sam Newman developing a microservice application, first to conduct is determining the boundaries of the environment or bounded context. Simply, to specify the bounded context of the application that can be seen from the application business process which can then be grouped according to functional groups from the user, e.g., the finance department is in charge of the payment, and the warehouse department is engaged in customer orders. Then each function will be put together into one module. The module then becomes the bounded context to create microservices that match the purpose of the module made. The microservices that have been made will apply the concept of loose coupling between other modules and high cohesion microservice that are interconnected with the module made. Finally, the part which will do data writing to the database and read data from the database can be determined.

Janssen & Joha explained that the use of the Software as a Service (SaaS) model in the public sector is still infrequent. Although SaaS in the public/e-government sector promises many

advantages, such as cost-saving and effectiveness, the challenges are severe, e.g., quality, security, privacy, and also the need to customize different systems in the region One with other areas.

### Methodology

* 1. *Functional requirement analysis*

Table 1. Functional requirements

|  |  |  |
| --- | --- | --- |
| ID | Actor | Functional Requirements |
| FR1 | Administrator | Customer registration |
| FR2 | Administrator | Citizen ID management |
| FR3 | Administrator | Category management |
| FR4 | Citizen | ID verification |
| FR5 | Citizen | Sent a complaint |
| FR6 | Citizen | Check complaint |
| FR7 | Government Work Unit | Show complaint recap |
| FR8 | Government Work Unit | Answer a complaint |
| FR9 | Government Work Unit | Delete a complaint |
| FR10 | Vendor | View all customer |

Functional requirement analysis was conducted by looking at some similar applications. For example, the public complaint web application in Kediri city. The analysis was attained through looking at design document of the application. Also, the analysis was made from literature studies to find some information related to the functional requirement of e-government public reporting applications.

Functional requirement consists of actor requirement and functional requirement. The actor is a user who will use the application. Few actors who use the application include an administrator, vendor, citizen, and the government work unit. The functional requirement itself is the function of the application. A list of some functional requirements of the application is in Table 1.

* 1. *Model the microservice*

After a functional requirement was made, the next step is to model the microservice. This step splits the functional requirement which is also known as bounded context into some microservices that match the purpose of the bounded context. In other words, the microservice is a group consisting of one or more microservices, and they are interconnected to perform a business process or a function. The microservices in this application is in Table 2.

Table 2. Microservice requirements.

ID Microservice Requirements

FR1 Create a new customer data

FR2 Create citizen ID data

FR2 Show list of citizen ID data

FR2 Delete citizen ID data

FR3 Create a category

FR3 Delete a category

FR3 Show list of category

FR4 Get a citizen ID data

FR5 Create a new complaint

FR5 Get a ticket number

FR6 Show complaint status by ticket

FR7 Show all complaint with particular status

FR8 Update answer in a complaint data

FR9 Delete a complaint

FR10 Show all registered customer

* 1. *Designing use case*

The designing use case was generated from the analysis of functional requirement. Use case shows the interaction between the actor with the application like the Fig. 1.

* 1. *Application development*

This phase is the process of coding the application based on functional requirement, microservices requirement, and use case. This research focuses on developing the back-end application using java programming language and springboot framework. Springboot framework was favored for it has several advantages.

Spring has supported MVC and provides a RESTful Web Service feature. A database connection has also been provided in the spring package. Spring framework also supports a dependency injection. Dependency injection is the ease of configuration dependency in the application so that in the application development process becomes more convenient. Each spring framework also supports Aspect Object Programming (AOP). The additional advantage using spring boot is that a tomcat server could easily be included and can be run directly.

Developing the front-end application is also needed for communicating to the back-end for making business process. The front-end development is made using typescript programming language and Angular2 framework.

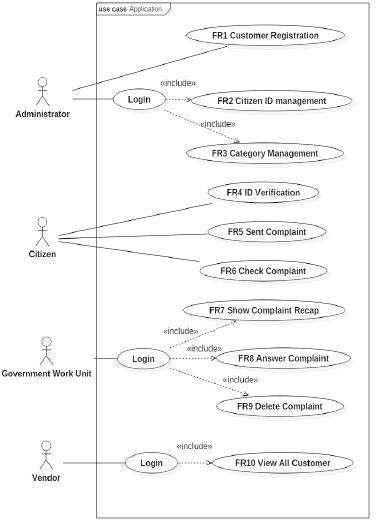


Fig. 1. Application use case.

* 1. *Microservice blackbox testing*

Blackbox testing is a test conducted to check whether the functional application is running correctly without knowing the processes occur within the app. Blackbox testing was done by creating a test case in the form of test input and output expected from functional applications. Testing can be done on applications that do not use algorithms or low granularity levels, so it does not take much time. Although for testing microservice two testings are needed, which are white box and black box testing, this research just performs the black box test because the application does not use any sophisticated algorithm and has low granularity details then the black box test is enough.

### Application business process

The public complaint application is operated and managed by a specific unit or team. In some cases, in areas such as Bandung city, this kind of application is operated and managed in command center so that its work is optimal.

This public complaint application is a cloud-based application. Local governments can rent the application services by paying some money to the vendor. An administrator can be assigned by the government for operating the applications. The administrator can upload citizen ID data.

Citizens create complaint via the application in web. To begin with, citizens must verify their ID based on the uploaded data ID by the administrator. After that, their complaint will be saved on the database and ready to be managed by the government work unit. First, there is a selection stage by the system to filter the content of the complaint. The selection process is managed by the government work unit in the command center. Valid complaint content will be classified by category that has been created by the administrator and then sent to the right government unit outside the command center so that that complaint can be answered with the right answer and right follow-up in the field. If the complainant report is addressed or handled in the field, then the citizen who sent the complaint may change the complaint status to completion to inform the government that the complaint has been adequately addressed.

### Application architecture

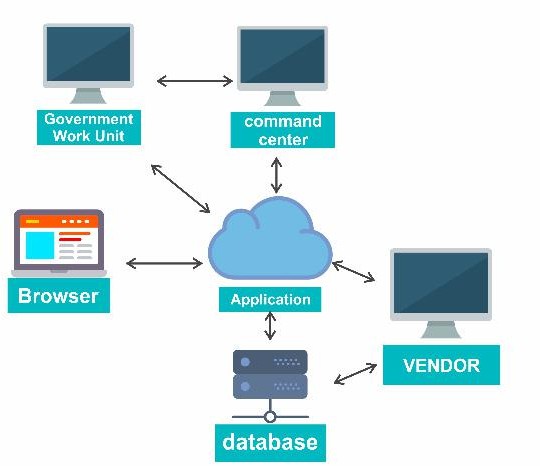


Fig. 2. Application architecture.

The public complaint application that uses cloud technology has a cloud architecture like Fig.

2. Front-end applications will be stored in the cloud as well as back-end applications along with application databases in the same cloud service to save resource usage. The application is accessible to the public on the web through a browser. Complaints submitted through the web application will be sent into the database for further management by the government which in this case is the command center and other government work unit. Vendors access the applications either back-end or front-end to perform maintenance on the application.

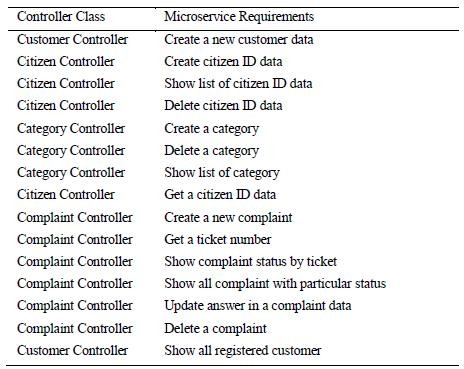
### Microservice architecture

There is a tool that can make it easier for developers to see all the microservice available on the application. The tool is called swagger. Swagger is a standard framework that allows developers to quickly find and understand all the services on the application without accessing the program code, application development documentation, and without requiring inspection of the application service network. Swagger maps the microservice based on the controller class that has been created using springboot framework. Class controller (or can be called a controller) is a class file in springboot which consists of methods to perform an input-output process according to business needs made. In the controller class, the methods can be made into a REST

API. Swagger shows all the controllers that have been created in the application. In each swagger controller the REST API can be accessed with the description of the method used (the REST method used is usually GET, POST, DELETE, PUT), then there is the REST API URL, and also the method name in the controller class. REST API is what will be used by the front-end application as access to a microservice with a predefined communication method that has been made.

* 1. *Class controller*

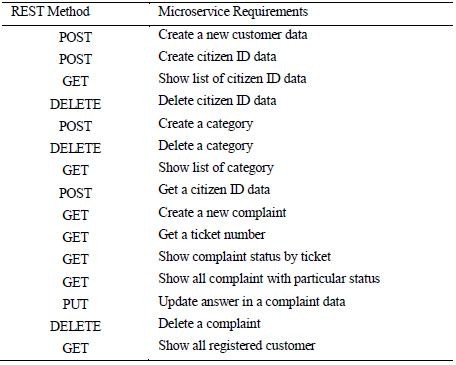
Table 3 shows class controllers that exist in the application with the associated microservices

Table 3. Controller class.

* 1. *Rest method*

Table 4 is a table containing REST method information that exists in the application with the associated microservices.

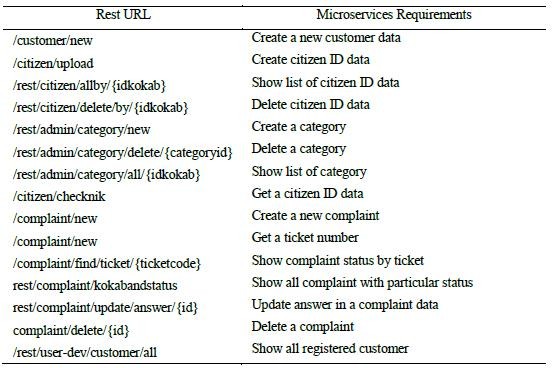
Table 4. REST method.



* 1. *REST URL*

Table 5 contains the URL of the REST. The URL is used to communicate between front end and backend.

Table 5. REST URL.



* 1. *Architecture diagram*

The front-end can be divided into four user interfaces (UI), namely admin dashboard (Admin UI) for administrator, Operator UI for government work unit, UI complaint that is used by citizen to create and check compliant, and Vendor UI that is used by the vendor. Each UI will be associated with some microservice as shown in Fig. 3.

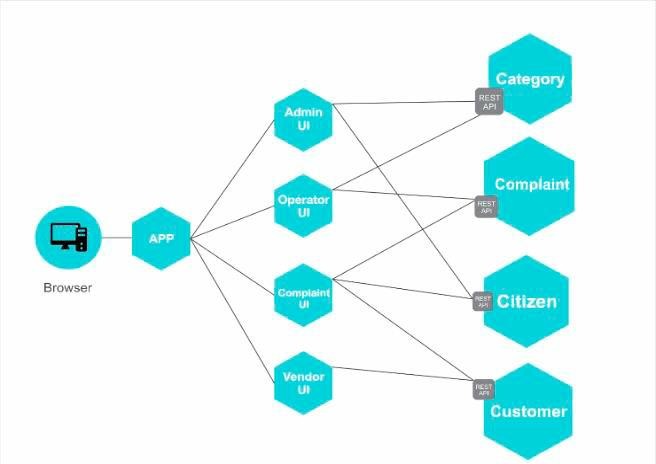


Fig. 3. Microservices achitecture.

* 1. *Securing microservice*

Fig. 1 depicts that administrator, government work unit, and the vendor have login functionality. The login itself is to be used to secure microservices for those actors. Those actors

have to fill their username and password, and then the system will compare them to saved data in the database. If username and password match, the system will send a Jason web token (JWT). JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties which are the back-end and the front-end. JWT contains some claims such as user scope that refers to the user actor and their dashboard, expired token time. So, when the time comes it will require a login process again and some additional information which is not of sensitive information that was required. The password itself need to be saved in database safely. Hashing function needs to be done so the password in the database is not a just plain text so it can be stolen easily. This research is using a BCrypt algorithm. BCrypt is an irreversible algorithm with low hash collision, using which our password is protected with proper protection. The Java language provides bcrypt library, so this is a convenient way.

### Conclusion

Based on the processes that have been done in this research, conclusions are as follows: 1) The public complaint is an application as a service developed with microservice architecture using springboot framework. Therefore, this application was deployed in the cloud. In deploying the application, a jar file was made from the back-end. The front-end was also deployed into the same cloud system as well as database applications used. To access the application, it takes a browser that will create communication to applications that are in the cloud. 2) In developing a microservice application, the functional needs should be broken down into several microservices that are interconnected to form a unified application by business processes that have been determined. 3) The advantages of using microservice architecture are that more functional needs can be added and also microservices within it without affecting other microservices. Thus, the developer can add more functionality independently, and that creates a business value which saves time to further development.

1. 介绍

# 附录 B 外文翻译—译文部分

电子政务包括使用互联网等电子通信技术，以改善公民获得公共服务的机会。电子政务的实施提供了报告管理系统的管理效率和速度，以及政府管理过程中透明度。通过它， 城市出现了一个良好治理的形象。然而，印度尼西亚电子政务的实施面临许多问题，例如开发和运营电子政务应用的费用问题，安全问题，隐私和系统更新等技术问题以及缺乏能力的人力资源问题和如何管理它。云技术成为解决这些问题的替代解决方案之一。此模型允许消费者通过可在各种设备中访问的提供商在线使用云中存在的应用程序，而无需担心这些问题。

微服务技术的使用可以为云电子政务系统带来一些优势。微服务中的模块化概念允许管理在应用程序中单独存在的服务。对特定服务的开发的影响是它不会干扰其他服务。 可以在其他服务之间区分服务的能力建立，以便适当地使用资源。此外，可以使用不同的编程语言开发服务的开发。

1. 相关工作

Sam Newman 开发微服务应用程序，首先要进行的是确定环境的边界或有界的上下文。简单地说，指定应用程序的有界上下文，可以从应用程序业务流程中看到，然后可以根据用户的功能组进行分组，例如，财务部门负责支付，仓库部门参与 在客户订单中。 然后将每个函数放在一个模块中。 然后，该模块成为要创建的有界上下文，符合模块目的的微服务。 已经制造的微服务将应用其他模块和高内聚微服务之间的松耦合概念，这些模块与所制造的模块互连。最后，可以确定将数据写入数据库并从数据库读取数据的部分。

Purnama 和 Yatini 使用 Node.js 开发了一个论文管理应用程序，旨在避免主题或论文题目的任何相似性，这通常是剽窃的情况。Node.js 是使用微服务架构构建的，旨在简化应用程序的开发。当存在新功能添加的情况时，不需要重新创建应用程序，并且可以独立地添加功能。 进一步开发需要的时间更少。

Janssen 和 Joha 解释说，在公共部门使用软件即服务（SaaS）模式的情况仍然很少。虽然公共/电子政务领域的 SaaS 承诺许多优势，例如节省成本和有效性，但挑战是严峻的， 例如质量，安全性，隐私，以及需要在该地区定制不同的系统。

1. 研究方法
   1. 功能需求分析

|  |  |  |
| --- | --- | --- |
|  | 表 1 | 功能要求 |
| ID | 活动者 | 功能要求 |
| FR1 | 管理员 | 用户注册 |
| FR2 | 管理员 | 公民身份 ID 管理 |
| FR3 | 管理员 | 分类管理 |
| FR4 | 城市居民 | 身份 ID 验证 |

|  |  |  |
| --- | --- | --- |
|  | | 续表 1 |
| FR5 | 城市居民 | 发表一个投诉 |
| FR6 | 城市居民 | 检查投诉内容 |
| FR7 | 政府工作小组 | 显示投诉回顾 |
| FR8 | 政府工作小组 | 回复投诉内容 |
| FR9 | 政府工作小组 | 删除投诉内容 |
| FR10 | 应用供应商 | 查看所有的用户 |

通过查看一些类似的应用程序进行功能需求分析。例如，Kediri 市的公共投诉网络申请。通过查看应用程序的设计文档来获得分析。此外，分析来自文献研究，以找到与电子政务公共报告应用的功能要求相关的一些信息。

功能需求包括演员要求和功能要求。actor 是将使用该应用程序的用户。使用该应用程序的演员很少包括管理员，供应商，公民和政府工作单位。功能需求本身就是应用程序的功能。表 1 列出了该应用程序的一些功能要求。

* 1. 微服务模型

在完成功能要求之后，下一步是对微服务进行建模。此步骤将功能需求（也称为有界上下文）拆分为一些与有界上下文的目的相匹配的微服务。换句话说，微服务是由一个或多个微服务组成的组，并且它们互连以执行业务过程或功能。本应用中的微服务见表 2。

表 2 微服务要求

ID 微服务要求

FR1 创建新的用户数据

FR2 创建公民身份 ID 数据

FR2 显示公民身份 ID 数据列表

FR2 删除公民身份 ID 数据

FR3 创建一个类别

FR3 删除一个类别

FR3 显示类别列表

FR4 获取公民身份 ID 数据

FR5 创建新投诉

FR5 获取票证信息

FR6 按票证显示投诉状态

FR7 显示具有特定状态的所有投诉

FR8 更新投诉数据中的回复

FR9 删除投诉

FR10 显示所有已注册的用户

* 1. 用例设计

设计用例是从功能需求分析中产生的。用例显示了 actor 与应用程序之间的交互，如图

1 所示。

* 1. 应用开发

此阶段是基于功能需求，微服务需求和用例对应用程序进行编码的过程。本研究的重点是使用 java 编程语言和 springboot 框架开发后端应用程序。Springboot 框架受到青睐， 因为它有几个优点。

Spring 支持 MVC 并提供 RESTful Web 服务功能。Spring 包中还提供了数据库连接。Spring 框架还支持依赖注入。依赖注入是应用程序中配置依赖的简易性，因此在应用程序开发过程中变得更加方便。每个 spring 框架还支持 Aspect Object Programming（AOP）。使用 spring boot 的另一个好处是可以轻松地包含 tomcat 服务器并可以直接运行。

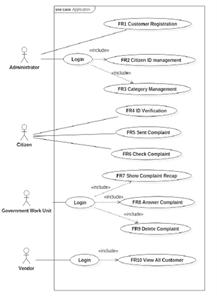
除此之外，还需要开发前端应用程序，以便与后端进行通信以进行业务流程。前端开发使用 typescript 编程语言和 Angular2 框架。

图 1 应用程序用例

* 1. 微服务黑盒测试

黑盒测试是为了检查功能应用程序是否正确运行而不知道应用程序中发生的进程而进行的测试。黑盒测试是通过以功能应用程序预期的测试输入和输出的形式创建测试用例来完成的。可以对不使用算法或低粒度级别的应用程序进行测试，因此不需要花费太多时间。虽然对于测试微服务需要两个测试，即白盒和黑盒测试，但这项研究只是执行黑盒测试， 因为应用程序不使用任何复杂的算法，并且具有低粒度细节，然后黑盒测试就足够了。

1. 申请业务流程

公共投诉申请由特定单位或团队运营和管理。在某些情况下，在万隆市等地区，这种应用程序在指挥中心进行操作和管理，以使其工作最佳。

此公共投诉应用程序是基于云的应用程序。 地方政府可以通过向供应商支付一些钱来租用应用程序服务。 政府可以指派管理员来操作应用程序。 管理员可以上传公民身份数

据。

公民通过网络应用程序创建投诉。首先，公民必须根据管理员上传的数据 ID 验证其 ID。之后，他们的投诉将保存在数据库中，并准备由政府工作单位进行管理。首先，系统有一个选择阶段来过滤投诉的内容。选择过程由指挥中心的政府工作单位管理。有效的投诉内容将按照管理员创建的类别进行分类，然后发送到指挥中心以外的权利政府部门，以便在正确的答案和正确的后续跟踪中回答该投诉。如果在现场处理或处理投诉人报告，则发送投诉的公民可以将投诉状态更改为完成，以通知政府投诉已得到充分解决。

1. 应用架构

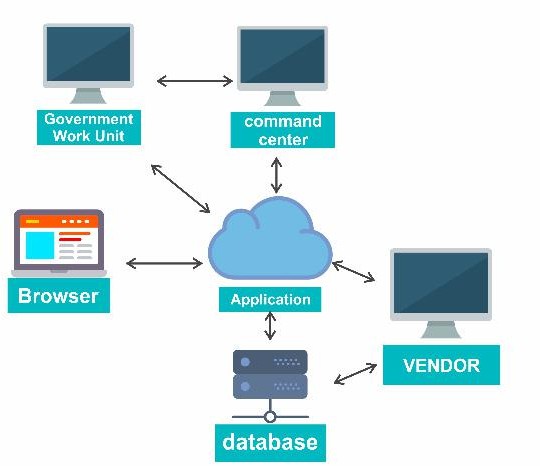


图 2 应用程序架构

使用云技术的公共投诉应用程序具有如图 2 所示的云架构。前端应用程序将存储在云端以及后端应用程序中，同时存储在同一云服务中的应用程序数据库以节省资源使用。该应用程序可通过浏览器在 Web 上公开访问。通过网络应用程序提交的投诉将被发送到数据库以供政府进一步管理，在这种情况下，政府是指挥中心和其他政府工作单位。供应商访问后端或前端应用程序以对应用程序执行维护。

1. 微服务架构

有一个工具可以让开发人员更容易地看到应用程序上可用的所有微服务。该工具称为Swagger。Swagger 是一个标准框架，允许开发人员快速查找和理解应用程序上的所有服务， 而无需访问程序代码，应用程序开发文档，也无需检查应用程序服务网络。Swagger 根据使用 springboot 框架创建的控制器类映射微服务。类控制器（或可称为控制器）是 springboot 中的类文件，它包含根据业务需求执行输入输出过程的方法。在控制器类中，可以将方法制作为 REST API。Swagger 显示了在应用程序中创建的所有控制器。在每个 swagger 控制器中，可以使用所用方法的描述访问 REST API（使用的 REST 方法通常是GET，POST， DELETE，PUT），然后是 REST API URL，以及控制器类中的方法名称。REST API 将由前端应用程序用作访问微服务的方法，该方法具有已经实现的预定义通信方法。

* 1. 控制器类

表 3 显示了应用程序中存在的具有相关微服务的类控制器。

* 1. **REST** 方法

表 3 控制器类

控制器类 微服务要求

用户控制器 创建新的用户数据

城市居民控制器 创建公民身份 ID 数据

城市居民控制器 显示公民身份 ID 数据列表城市居民控制器 删除公民身份 ID 数据

类别控制器 创建一个类别

类别控制器 删除一个类别

类别控制器 显示类别列表

城市居民控制器 获取公民身份 ID 数据投诉控制器 创建新投诉

投诉控制器 获取票证信息

投诉控制器 按票证显示投诉状态

投诉控制器 显示具有特定状态的所有投诉投诉控制器 更新投诉数据中的回复

投诉控制器 删除投诉

用户控制器 显示所有已注册的用户

表 4 是包含 REST 方法信息的表，该信息存在于具有相关微服务的应用程序中。表 4 REST 方法

控制器类 微服务要求

POST 创建新的用户数据

POST 创建公民身份 ID 数据

GET 显示公民身份 ID 数据列表

DELETE 删除公民身份 ID 数据

POST 创建一个类别

DELETE 删除一个类别

GET 显示类别列表

POST 获取公民身份 ID 数据

GET 创建新投诉

GET 获取票证信息

GET 按票证显示投诉状态

PUT 更新投诉数据中的回复

DELETE 删除投诉

GET 显示所有已注册的用户

* 1. **REST** 的 **URL** 链接

表 5 包含 REST 的 URL，URL 用于在前端和后端之间进行通信。

表 4 REST 的 URL

REST URL 微服务要求

/customer/new 创建新的用户数据

/citizen/upload 创建公民身份 ID 数据

/rest/citizen/allby/{idkokab} 显示公民身份 ID 数据列表

/rest/citizen/delete/by/{idkokab} 删除公民身份 ID 数据

/rest/admin/category/new 创建一个类别

/rest/admin/category/delete/{categoryid} 删除一个类别

/rest/admin/category/all/{idkokab} 显示类别列表

/citizen/checknik 获取公民身份 ID 数据

/complaint/new 创建新投诉

/complaint/new 获取票证信息

/complaint/find/ticket/{ticketcode} 按票证显示投诉状态

rest/complaint/kokabandstatus 显示具有特定状态的所有投诉

rest/complaint/update/answer/{id} 更新投诉数据中的回复

complaint/delete/{id} 删除投诉

/rest/user-dev/customer/all 显示所有已注册的用户

* 1. 架构图

前端可分为四个用户界面（UI），即管理员的管理仪表板（管理界面），政府工作单位的操作员界面，公民用于创建和检查合规的 UI 投诉，以及供应商 UI 给供应商使用。每个 UI 将与一些微服务相关联，如图 3 所示。

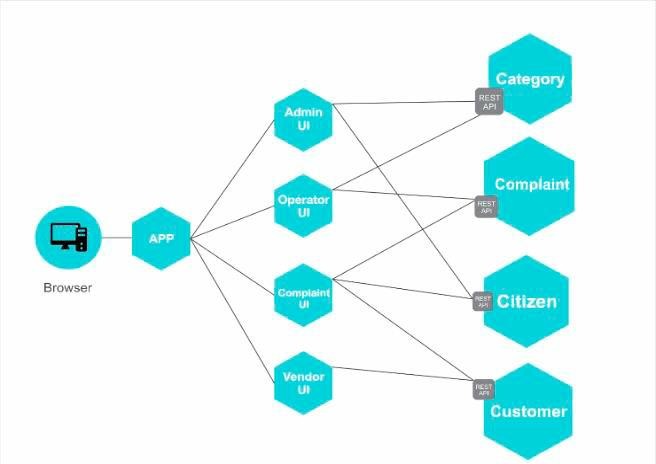


图 3 微服务架构

* 1. 微服务的安全性

图 1 描绘了管理员，政府工作单位和供应商具有登录功能。登录本身将用于保护这些角色的微服务。这些 actor 必须填写他们的用户名和密码，然后系统会将它们与数据库中保存的数据进行比较。如果用户名和密码匹配，系统将发送 Jason Web 令牌（JWT）。JSON

Web 令牌（JWT）是一种紧凑的，URL 安全的方式，用于表示在作为后端和前端的两方之间转移的声明。JWT 包含一些声明，例如引用用户 actor 及其仪表板的用户范围，过期的令牌时间。因此，当时机成熟时，它将需要再次登录过程以及一些不需要敏感信息的附加信息。密码本身需要安全地保存在数据库中。需要进行散列函数，因此数据库中的密码不仅仅是纯文本，因此很容易被盗。这项研究使用的是 BCrypt 算法。BCrypt 是一种具有低哈希冲突的不可逆算法，使用该算法可以保护我们的密码并提供适当的保护。Java 语言提供了 bcrypt 库，因此这是一种方便的方法。

1. 总结

基于本研究中已经完成的过程，结论如下：

1. 公众投诉是一种应用程序，即使用 springboot 框架通过微服务架构开发的服务。因此，此应用程序部署在云中。在部署应用程序时，从后端制作了一个 jar 文件。前端也部署在同一个云系统以及使用的数据库应用程序中。要访问该应用程序，需要一个浏览器来创建与云中应用程序的通信。
2. 在开发中在微服务应用程序中，功能需求应分解为若干微服务，这些微服务通过已确定的业务流程互连以形成统一的应用程序。
3. 使用微服务架构的优点是可以添加更多的功能需求以及其中的微服务而不影响其他微服务。因此，开发人员可以独立地添加更多功能，并创建业务价值，从而节省进一步开发的时间。

# 附录 C 软件使用说明书

装修家居服务平台是一个将装修公司、设计师和平台用户（即业主）汇聚一体的合作服务交流平台，平台的功能有：装修公司入驻、装修招标、装修公司信息管理、设计师风采展示、装修交流平台管理和装修服务信息管理。

1. 装修公司入驻

装修公司入驻是有关装修公司或者企业入驻平台的操作，由公司负责人填写入驻表相关信息，提交之后待平台管理员确认入驻本平台。相关装修公司入驻申请页面如图 1 所示：

1. 装修招标

图 1 装修公司入驻申请页面

装修招标是提供给平台用户（即业主）在登录平台后，在平台发布装修服务招标信息。平台用户填写相关的招标信息，发布成功后供公司进行投标，双方达成合作。相关装修招标页面如图 2 所示：



图 2 装修招标页面

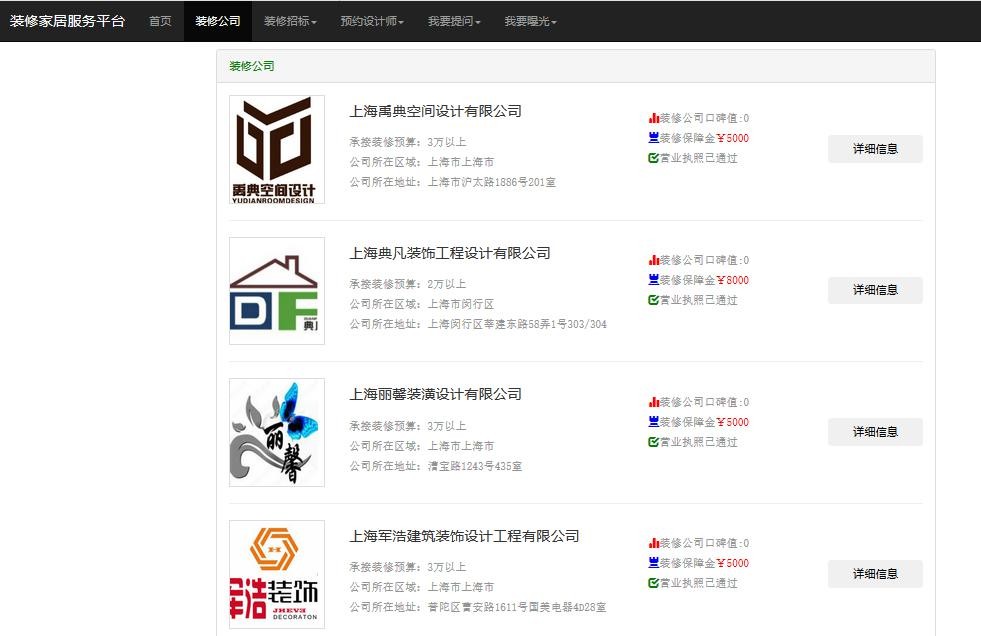
1. 装修公司信息管理是在公司入驻平台，待平台管理员添加装修公司信息后，公司的信息会在平台前台展示，平台用户可以浏览相关信息资料，查看公司信息和装修服务等。相关装修公司信息展示页面如图 3 所示：

图 3 装修公司信息展示页面

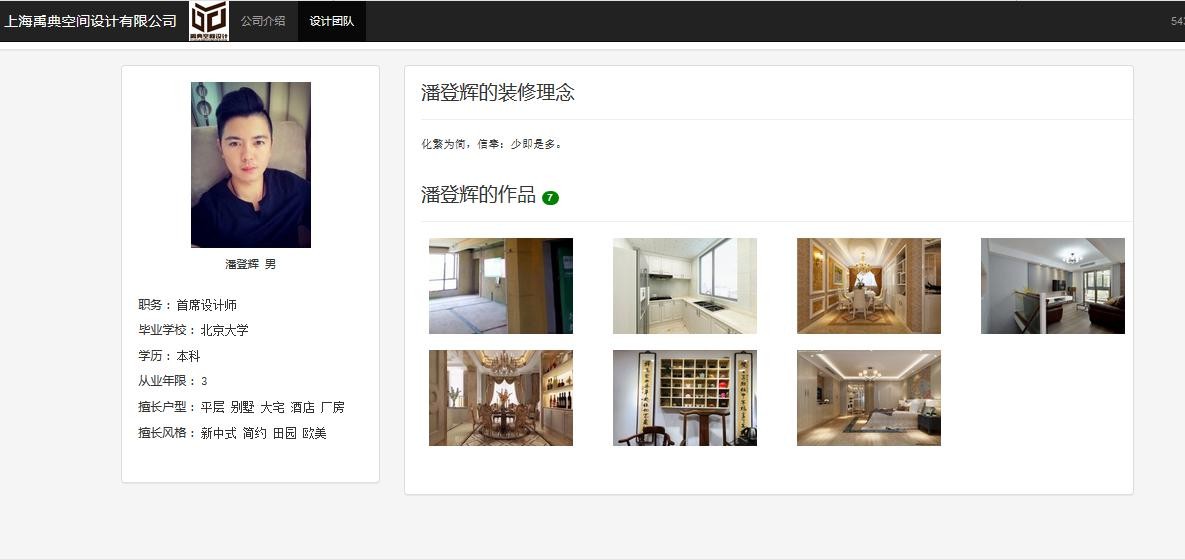
1. 设计师风采展示是在设计师上传个人装修案例成果图片等信息后，相关效果图等信息会在平台展示，平台用户可以在平台的设计师风采页面查看设计师相关信息以及装修案例图片等信息。相关设计师风采展示页面如图 4 所示：

图 4 设计师风采展示页面

1. 装修交流平台管理包括装修曝光和装修疑惑管理。平台用户如果对装修公司服务等不满意，可以直接在平台上发布曝光信息，信息由平台管理员核查通过后在平台展示曝光。平台用户可以发布装修疑惑，公司负责人可以对疑惑进行解答。相关装修疑惑信息页面如图 5 所示：



图 5 装修疑惑信息页面

1. 装修服务信息管理是由平台管理员登录系统后进行装修资讯和和装修攻略等信息的发布管理，发布成功后在平台展示。相关装修服务信息管理页面如图 8 所示：

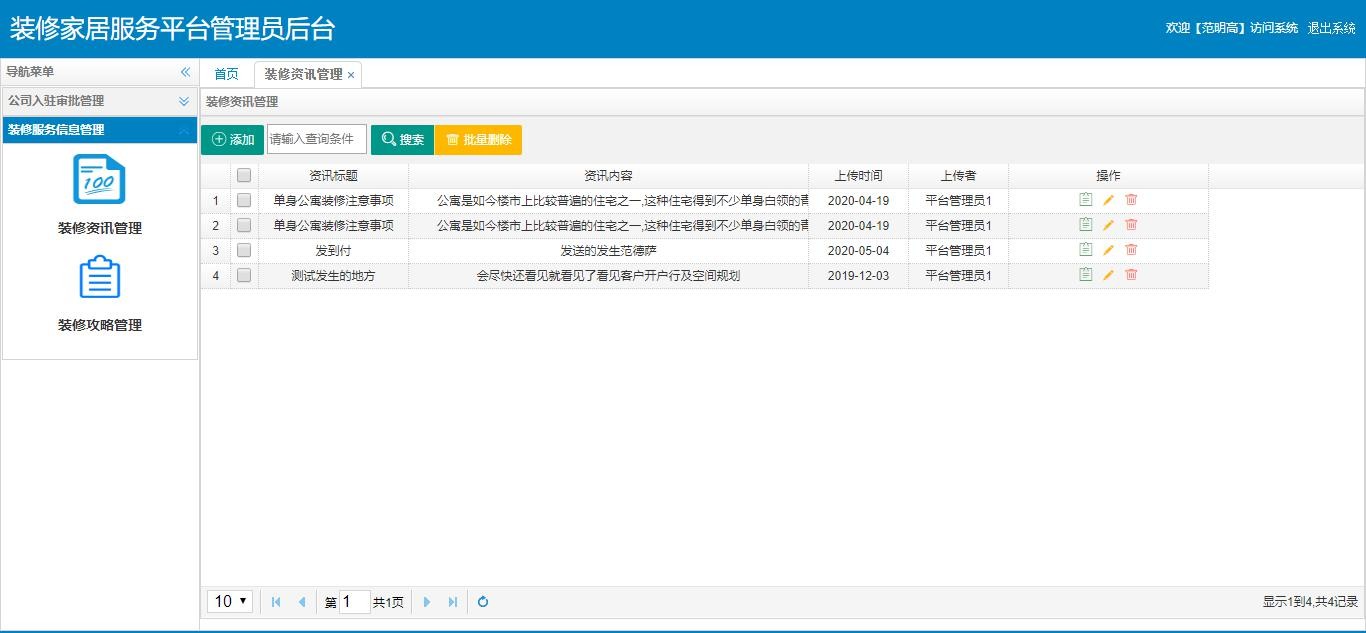


图 6 装修服务信息管理

# 附录 D 主要源代码

@RequestMapping(value = "/add", method = RequestMethod.POST) @ResponseBody

public Object add(@RequestBody(required = true) Map<String, Object> p) { KesunReturn back = new KesunReturn();//返回操作

AbsSuperObject temp = initParameter(p);

if (temp == null) { back.setCode("0");

back.setMessage("抱歉，系统没有获取到需要数据！"); back.setObj(null);

return back;

}

AbsSuperObject model = updateModel(temp);//获取客户端参数信息if (model == null) {

back.setCode("0");

back.setMessage("抱歉，您提供的数据存在问题，系统在转换成目标数据时失败！"); back.setObj(null);

return back;

}

back=checkServiceObject(getService()); if (back!=null) return back;

back=new KesunReturn(); getService().setModel(model);

if (getService().isAdd() == false) { back.setCode("0");

back.setMessage("抱歉，系统设置不能新增【" + model.getName() + "】的操作,数据是否重复，

请核实！");

back.setObj(null); return back;

}

return getService().add();

}

@RequestMapping(value = "/batchSave", method = RequestMethod.POST) @ResponseBody

public Object batchSave(@RequestBody(required = true) List<Map<String,Object>> items) {

if (items!=null && items.size()>0)

{

List lTemp=new ArrayList(); for(Map<String,Object> obj:items)

{

AbsSuperObject temp=initParameter(obj);

if (temp.getId()==null || temp.getId().equals("")) temp.setId(Tool.CreateID());

lTemp.add(temp);

}

return getService().batchAdd(lTemp);

}

KesunReturn back=new KesunReturn(); back.setCode("0");

back.setMessage("抱歉，您提供的数据为空，无法进行数据批量操作！");

back.setObj(0); return back;

}

@RequestMapping(value = "/edit", method = RequestMethod.POST) @ResponseBody

public Object edit(@RequestBody(required = true) Map<String, Object> p) { KesunReturn back = new KesunReturn();//返回操作

AbsSuperObject temp = initParameter(p);

if (temp == null) { back.setCode("0");

back.setMessage("抱歉，系统没有获取到的修改数据！"); back.setObj(null);

return back;

}

AbsSuperObject model = updateModel(temp);//获取客户端参数信息if (model == null) {

back.setCode("0");

back.setMessage("抱歉，您提供的数据存在问题，系统在转换成目标数据时失败，无法继续完 成修改操作！");

back.setObj(null);

return back;

}

back=checkServiceObject(getService()); if (back!=null) return back;

back=new KesunReturn(); getService().setModel(model);

if (getService().isEdit() == false) { back.setCode("0");

back.setMessage("抱歉，系统设置要求取消了【" + model.getName() + "】修改操作,数据是否重

复，请核实！");

back.setObj(null); return back;

}

return getService().edit();

}

@RequestMapping(value = "/del", method = RequestMethod.POST) @ResponseBody

public Object del(@RequestBody(required = true) Map<String, Object> p) { KesunReturn back = new KesunReturn();//返回操作

AbsSuperObject temp = initParameter(p); if (temp == null) {

back.setCode("0");

back.setMessage("抱歉，系统没有获取到的删除数据！"); back.setObj(null);

return back;

}

AbsSuperObject model = updateModel(temp);//获取客户端参数信息if (model == null) {

back.setCode("0");

back.setMessage("抱歉，您提供的数据存在问题，系统在转换成目标数据时失败，无法继续完 成删除操作！");

back.setObj(null); return back;

}

back=checkServiceObject(getService());

if (back!=null) return back; back=new KesunReturn(); getService().setModel(model);

if (getService().isDelete() == false)

{

back.setCode("0");

back.setMessage("抱歉，系统设置要求取消了【" + model.getName() + "】删除操作！"); back.setObj(null);

return back;

}

if (getService().isInUse())

{

back.setCode("0");

back.setMessage("抱歉，您指定删除【" + model.getName() + "】数据在应用中，不能删除！"); back.setObj(null);

return back;

}

return getService().del();

}

@RequestMapping(value = "/deleteAll", method = RequestMethod.POST) @ResponseBody

public Object deleteAll(@RequestBody(required = true) Map<String, Object> p) { String ids = p.get("ids").toString();//解析参数

List<String> cons = new ArrayList<String>(); String[] arrays = ids.split(",");

for (String id : arrays) {

if (!id.trim().equals("")) cons.add(id);

}

KesunReturn back = new KesunReturn();//返回操作if (cons.size() == 0) {

back.setCode("0");

back.setMessage("抱歉，系统没有获得有效的删除数据参数，请核实！"); back.setObj(null);

return back;

}

back=checkServiceObject(getService()); if (back!=null) return back;

return getService().deleteAll(cons);

}

@RequestMapping(value = "/findByPage", method = RequestMethod.POST) @ResponseBody

public Object findByPage(@RequestBody(required = true) Map<String, Object> p) { Map<String, Object> values =getConditionParam(p);//获取查询参数

/\*合并数据过滤条件\*/

if (setFindFilter(p) != null) values.putAll(setFindFilter(p));

KesunSuperService bll = getService();//获取业务对象

SearchViewParam pp = ControlTool.GetViewParam(p);//获取查询参数ReturnBeanInPower rbi = new ReturnBeanInPower(); rbi.setHasPower(setPowerFilter());

KesunReturn back=checkFindCondition(bll,values); if (back!=null) return back;

return getService().findByPage(values, pp.pageNumber, pp.rowsCount);

}

@RequestMapping(value = "/getMe", method = RequestMethod.POST) @ResponseBody

public Object getMe(@RequestBody(required = true) Map<String, Object> p) { KesunReturn back = new KesunReturn();//返回操作

AbsSuperObject temp = initParameter(p);

if (temp == null) { back.setCode("0");

back.setMessage("抱歉，系统没有获取有效参数！"); back.setObj(null);

return back;

}

AbsSuperObject model = updateModel(temp);//获取客户端参数信息if (model == null) {

back.setCode("0");

back.setMessage("抱歉，您提供参数数据存在问题，系统在转换成目标数据时失败，无法继续

完成数据提取！");

back.setObj(null); return back;

}

back=checkServiceObject(getService()); if (back!=null)

return back;

else

back=new KesunReturn(); getService().setModel(model); Object tmp = getService().getMe();

if (tmp != null && tmp instanceof AbsSuperObject) { model=(AbsSuperObject)tmp; model.setOldId(model.getId()); back.setCode("1");

back.setMessage("系统已经获取到您提取的数据！"); back.setObj(model);

}

else

{

back.setCode("0");

back.setMessage("系统没有提取到符合您要求的数据！"); back.setObj(null);

}

return back;

}