|  |  |
| --- | --- |
| 评分 | 指导教师签名 |
|  |  |





**专业综合设计**

**系统设计说明书**

|  |  |  |
| --- | --- | --- |
| 项目名称 | **：** | **“牵伴”APP——你牵挂着我，我陪伴着你** |
| 组长 | **：** | **1409030235：佘华扬** |
| 组员 | **：** | **张颖 廖娴静 宫春燕** |
| 班级 | **：** | **信管1402** |
| 指导教师 | **：** | **叶焕倬** |
| 完成时间 | **：** | **2017年X月X日** |

目录

[1 系统概述 2](#_Toc478077545)

[1.1编写目的 2](#_Toc478077546)

[1.2适用范围 3](#_Toc478077547)

[1.3产品目标 3](#_Toc478077548)

[1.4设计约定 3](#_Toc478077549)

[2 概要设计 3](#_Toc478077550)

[2.1设计思路 3](#_Toc478077551)

[2.2功能设计 4](#_Toc478077552)

[2.2.1用例描述 4](#_Toc478077553)

[2.2.2功能视图 5](#_Toc478077554)

[2.3逻辑架构 8](#_Toc478077555)

[2.4 物理架构 11](#_Toc478077556)

[2.5开发架构 11](#_Toc478077557)

[2.6运行环境设计 12](#_Toc478077558)

[2.7架构设计 13](#_Toc478077559)

[2.7.1系统分层 13](#_Toc478077560)

[2.7.2系统包、资源规划的准备工作 13](#_Toc478077561)

[3 详细设计 16](#_Toc478077562)

[3.1 流程设计 16](#_Toc478077563)

[3.2功能设计 27](#_Toc478077564)

[3.2.1模块设计 27](#_Toc478077565)

[3.2.2界面设计 28](#_Toc478077566)

[3.3接口设计 59](#_Toc478077567)

[3.3.1用户接口 59](#_Toc478077568)

[3.3.2外部接口 60](#_Toc478077569)

[3.3.3内部接口 60](#_Toc478077570)

[3.4非功能性设计 61](#_Toc478077571)

[3.4.1 非功能整体需求 61](#_Toc478077572)

[3.4.2 性能需求 61](#_Toc478077573)

[3.4.3 软件质量要求 62](#_Toc478077574)

[3.4.4 安全保密性要求 62](#_Toc478077575)

[3.4.5 数据管理能力要求 62](#_Toc478077576)

[3.4.6 故障处理要求 63](#_Toc478077577)

[3.4.7 其他专门要求 63](#_Toc478077578)

[3.5 数据库设计 63](#_Toc478077579)

[3.5.1物理数据库概述 63](#_Toc478077580)

[3.5.2数据库表设计 67](#_Toc478077581)

[3.6补充说明 69](#_Toc478077582)

[4 模块与代码设计 70](#_Toc478077583)

[4.1 模糊搜索算法 70](#_Toc478077584)

[4.1.1算法中的相关概念 70](#_Toc478077585)

[4.1.2算法的整体设计 70](#_Toc478077586)

[4.1.3数据结构的设计 70](#_Toc478077587)

[4.1.4模糊搜索的实现 72](#_Toc478077588)

[4.2 热度加权算法 72](#_Toc478077589)

[5.分解描述 73](#_Toc478077590)

[5.1通信模块 73](#_Toc478077591)

[5.1.1打开服务器端通信 73](#_Toc478077592)

[5.1.2打开客户端服务器 74](#_Toc478077593)

[5.2登录、注册、个人资料设置模块 75](#_Toc478077594)

[5.2.1登录功能 76](#_Toc478077595)

[5.2.2注册功能 76](#_Toc478077596)

[5.2.3个人信息设置模块 77](#_Toc478077597)

[5.3.语音聊天模块 78](#_Toc478077598)

[5.3.1添加老人 78](#_Toc478077599)

[5.3.2查看老人资料与删除 79](#_Toc478077600)

[5.3.3聊天 80](#_Toc478077601)

[5.4.论坛模块 81](#_Toc478077602)

[5.4.1创建贴子 81](#_Toc478077603)

[5.4.2查看贴子 82](#_Toc478077604)

[5.4.3评论贴子 82](#_Toc478077605)

[5.5.消息接受与推送模块 83](#_Toc478077606)

[5.5.1消息接受与提示 83](#_Toc478077607)

[5.5.2推文推送 84](#_Toc478077608)

[5.6.地图模块 84](#_Toc478077609)

[5.6.1地图显示 85](#_Toc478077610)

[5.6.2实时定位与历史轨迹 85](#_Toc478077611)

[6 接口描述 87](#_Toc478077612)

[6.1百度地图接口 87](#_Toc478077613)

[6.2闹钟接口 87](#_Toc478077614)

[6.3相册接口 87](#_Toc478077615)

[6.4 通讯接口 87](#_Toc478077616)

[7 出错处理设计 87](#_Toc478077617)

[7.1 错误异常处理机制和原则 87](#_Toc478077618)

[7.2 出错信息 88](#_Toc478077619)

[7.3 补救措施 90](#_Toc478077620)

[7.4 系统维护设计 90](#_Toc478077621)

1 系统概述

1.1编写目的

本系统设计说明书是针对“牵伴”APP的研发而编写。目的是对该产品进行详细设计，在需求分析的基础上进一步明确系统结构，详细地介绍系统的设计思路、各类架构及数据库设计等，为该产品的开发、测试提供设计参考以及为进行后续的设计和完善提供方便。

本系统设计说明书的预期读者为本项目小组的成员（开发者+运行维护者）以及对该产品感兴趣，日后想对该产品进行进一步了解的人员。

1.2适用范围

本文适合于“牵伴”APP的开发人员、测试人员阅读。

1.3产品目标

本APP的目标用户是年龄到一定程度的父母和子女，其中父母方面可能由于某些特殊原因并不能熟练使用智能手机中繁复的APP界面。

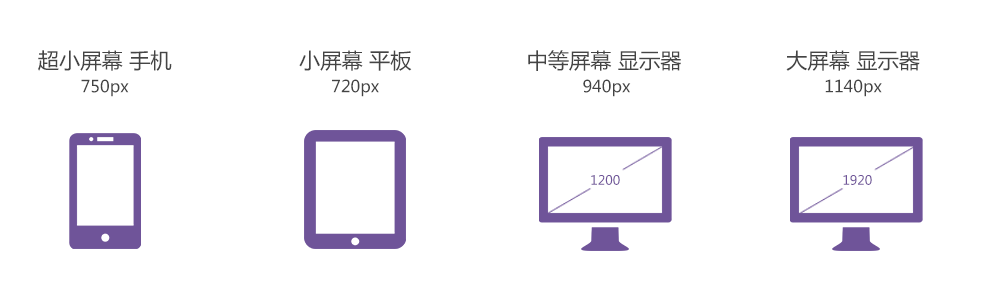
因此需要满足以下目标：

（1）产品能够提供友好的用户界面，使操作人员的工作量最大限度的减少。

（2）产品具有良好的运行效率，及时进行信息反馈，不因为卡顿等其他技术方面的问题影响用户的体验感。

（3）产品的整体风格要契合用户审美，设计过程中代码的重用性要高。

1.4设计约定

[](https://raw.githubusercontent.com/ColdXu/grid-design/master/img/icon.gif)

以上是四种适用于不同尺寸设备屏幕的方案

* 超小屏幕 手机 = 750px
* 小屏幕 平板 = 720px
* 中等屏幕 桌面显示器 = 940px
* 大屏幕 大桌面显示器 = 1140px

2 概要设计

2.1设计思路

根据系统需求分析报告，进行系统设计。根据相关功能要求，结合实际条件，具体设计实现逻辑模型、物理模型以及最终系统。

系统设计包括总体设计和模块设计。按功能不同进行技术层次划分，使各层功能相对独立。同时以接口形式来描述各层之间的调用关第，以达到层次之间的松散耦合。各层所提供功能不依赖于一种具体的技术或产品实现，应该提供一定范围的技术选择。技术架构不和具体的应用架构绑定，应具备较宽的使用范围，适合未来应用的扩展。

成果将以系统设计说明书为主。

2.2功能设计

根据需求分析说明书，“牵伴”分为父母版本、子女版本和管理员管理。

父母版本有3个主要模块，分别是：账户管理模块（个人注册、个人登陆、注销）、亲情互动模块（查看子女动态、语音交流）、收听电台等功能模块。

子女版本有6个主要模块，分别是：账户管理模块（个人注册、个人登陆、注销）、亲情互动模块（上传照片、语音交流、闹钟设置）、GPS定位模块（最新位置、历史轨迹）、论坛模块（发帖、评论）、推文模块（查看推文、历史推文）、其他模块（用户通知、意见反馈、检查更新、关于我们）等功能模块。

管理员版本有5个主要模块，分别是：用户管理模块（添加用户、搜索用户、修改用户信息、删除用户）、管理员管理模块（添加角色、修改角色信息、删除角色、权限管理）、推文管理模块（添加推文、搜索推文、删除推文等）、用户通知模块、系统管理（图片管理、数据字典）。

2.2.1用例描述

1、父母用例



图2.2.1.1 参与者父母用例图

参与者父母表示使用“牵伴”的老年人群体。

2、子女用例图

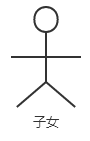


图2.2.1.2 参与者子女用例图

参与者子女表示使用“牵伴”的子女群体。

3、管理员用例图

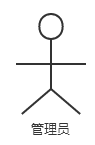


图2.2.1.2 参与者管理员用例图

参与者管理员表示 “牵伴”的后台管理员。

2.2.2功能视图

1、父母用例

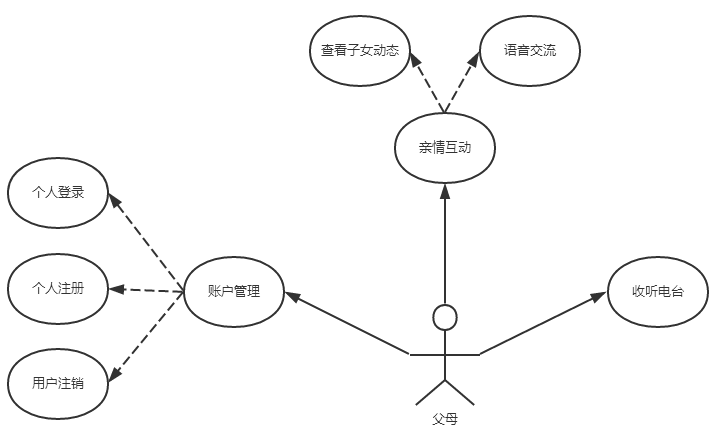


图2.2.2.1 父母用例图

父母用例图由1个参与者“父母”和8个不同用例组成。其中账户管理包含个人注册、个人登陆和注销，亲情互动包含查看子女动态和语音交流，收听电台不包含任何用例。

|  |  |  |
| --- | --- | --- |
| 功能模块 | 功能名称、标识符 | 描述 |
| 账户管理 | 个人登录 | 用于用户登陆。 |
| 个人注册 | 用于新用户注册。 |
| 用户注销 | 用于用户注销。 |
| 亲情互动 | 查看子女动态 | 父母可以查看子女上传的照片。 |
| 语音交流 | 父母可以跟子女进行语音交流。 |
| 收听电台 | 收听电台 | 父母用于收听电台、广播等节目。 |

2、子女用例图



图2.2.2.2 子女用例图

子女用例图由1个参与者“子女”和22个不同用例组成。其中账户管理包含个人注册、个人登陆和注销，亲情互动包含上传照片、语音交流和闹钟设置，GPS定位包含最新位置和历史轨迹，论坛包含发帖和评论，推文包含查看推文和历史推文，其他包含用户通知、意见反馈、检查更新和关于我们。

|  |  |  |
| --- | --- | --- |
| 功能模块 | 功能名称、标识符 | 描述 |
| 账户管理 | 个人登录 | 用于用户登陆。 |
| 个人注册 | 用于新用户注册。 |
| 用户注销 | 用于用户注销。 |
| 亲情互动 | 上传照片 | 子女可以上传照片给父母看。 |
| 语音交流 | 子女可以跟父母进行语音交流。 |
| 闹钟设置 | 子女可以设置闹钟提醒父母吃药等。 |
| GPS定位 | 最新位置 | 子女可以查看父母最新的位置信息。 |
| 历史轨迹 | 子女可以查看父母走动的历史轨迹。 |
| 论坛 | 发帖 | 子女可以在论坛发帖交流赡养父母的心得。 |
| 评论 | 子女可以评论交流赡养父母的心得。 |
| 推文 | 查看推文 | 子女可以查看父母相关的美文。 |
| 历史推文 | 子女可以查看历史推文。 |
| 其他 | 用户通知 | 用户可以查看系统通知。 |
| 意见反馈 | 用户可以反馈对“牵伴”的意见。 |
| 检查更新 | 用于系统更新的检查。 |
| 关于我们 | “牵伴”及开发团队介绍。 |

3、管理员用例

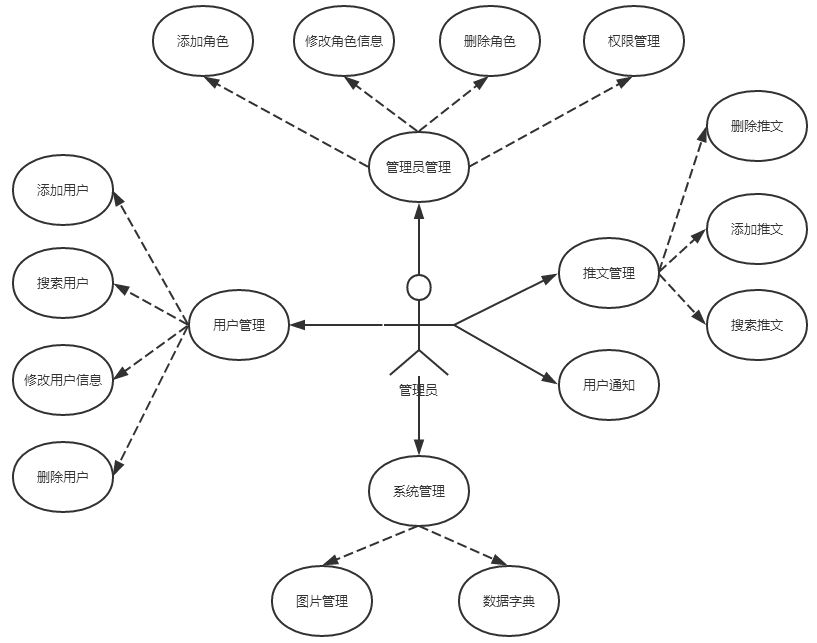


图2.2.2.3 管理员用例图

管理员用例图由1个参与者“管理员”和22个不同用例组成。其中用户管理包含添加用户、搜索用户、修改用户信息和删除用户，管理员管理包含添加角色、修改角色信息、删除角色和权限管理，推文管理包含添加推文、搜索推文和删除推文，系统管理包含图片管理和数据字典，用户通知不包含任何用例。

|  |  |  |
| --- | --- | --- |
| 功能模块 | 功能名称、标识符 | 描述 |
| 用户管理 | 添加用户 | 用于管理员添加新用户。 |
| 搜索用户 | 用于管理员搜索用户信息。 |
| 修改用户信息 | 用于管理员修改用户信息。 |
| 删除用户 | 用于管理员删除用户信息。 |
| 管理员管理 | 添加角色 | 用于管理员添加新角色。 |
| 修改角色信息 | 用于管理员修改角色信息。 |
| 删除角色 | 用于管理员删除角色信息。 |
| 权限管理 | 用于管理员对各角色进行权限的管理。 |
| 系统管理 | 图片管理 | 用于管理员对系统内图片进行管理。 |
| 数据字典 | 用于管理员对系统内数据进行管理。 |
| 推文管理 | 删除推文 | 管理员可以删除已发出的推文。 |
| 添加推文 | 管理员可以发送新推文。 |
| 搜索推文 | 管理员可以搜索已发出的推文。 |
| 用户通知 | 用户通知 | 对用户进行系统通知。 |

2.3逻辑架构

系统逻辑架构设计如下图：

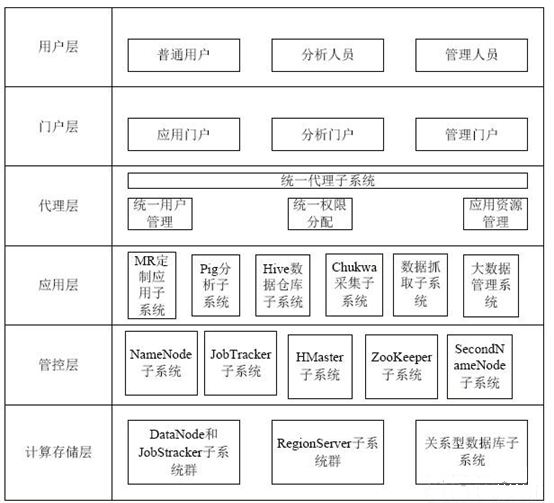


图2.3 逻辑架构图

系统的逻辑架构由用户层、门户层、代理层、应用层、管控层、计算存储层共6 层组成。管控层和计算存储层构成了系统的平台层，应用层的主要功能是支撑系统业务应用的开发和各子节点的后台管理。

（1）用户层

用户层是所有访问大数据系统的用户集合，根据用户职能的不同，大数据系统的用户可划分为3 类：普通人员、分析人员和系统管理人员。

（2）门户层

门户层是用户访问大数据系统的统一入口，用户经应用层管理系统统一认证后，门户子系统按照用户身份的不同分别提供相应的个性化门户，用户只能查阅和操作系统授权的数据信息和应用功能。

（3）代理层

代理层是门户层和应用层各应用系统之间的统一代理，代理层由统一代理子系统构成。统一代理子系统由统一用户管理、统一应用资源管理、统一权限管理三个核心模块构成。代理层向门户层开放出WebService 服务，供门户层的系统调用。当WebService 服务接收到门户层发来的请求信息时，会向应用层的应用系统发出调用请求。

（4）应用层

应用层由数据应用类、数据采集类、后台管理类三类子系统构成。应用类子系统包括MR 定制子系统、Pig 分析子系统和Hive 数据仓库子系统，主要为普通用户和分析人员提供大数据分析和挖掘服务。数据采集类子系统包括Chukwa 采集子系统和数据抓取子系统，主要是从企业现有系统、大数据应用系统和外部应用系统中采集数据并存储在大数据系统中。后台管理类子系统主要是对大数据系统的用户信息、节点设备信息和节点运行状态进行管理。

（5）管控层

管控层主要由NameNode 子系统、JobTracker 子系统、HBase 的Master 子系统、ZooKeeper子系统和SecondNameNode 子系统构成。

NameNode 子系统是HDFS 系统的重要组成部分，主要提供目录和文件的元数据管理，是分布式文件读写的统一入口。SecondNameNode 子系统是其备份子系统。

JobTracker 子系统是分布式作业系统的重要组成部分，是MapReduce 分布式程序的服务器端，主要是对作业的任务进行调度分配和运行状态的管理。

HBase 的Master 子系统是HBase 系统的主要组成部分，它主要负责引导初始安装，为Region 域数据分别注册到HBase 集群中的域服务器HRegionServer，监控HRegionServer 的运行，并在出现故障时进行恢复。

ZooKeeper 子系统主要是为HMaster 和HRegionServer 提供共享信息的统一协同服务。

（6）计算存储层

计算存储层由HDFS 系统的DataNode 和分布式作业系统的TaskTracker 组成的子系统、HRegionServer 子系统和关系型数据库管理子系统构成。

HDFS 系统的DataNode 和分布式作业系统的TaskTracker 组成的子系统共同部署在一个机器节点上，为分布式文件提供数据块存储和数据块计算服务。

HRegionServer 子系统主要是为客户端提供分布式数据库系统的访问服务。

关系型数据库子系统主要是存储应用层后台管理系统和Hive 数据仓库系统提供用户信息、机器设备信息、元数据、运行状态监测信息的存储服务。

2.4 物理架构

物理架构的目的是确定物理节点和物理节点的拓扑结构；其中物理节点包括服务器、PC机、专用机、软件安装部署以及系统软件的选型；拓扑结构明确物理节点的关系。本软件采用的物理架构如图：

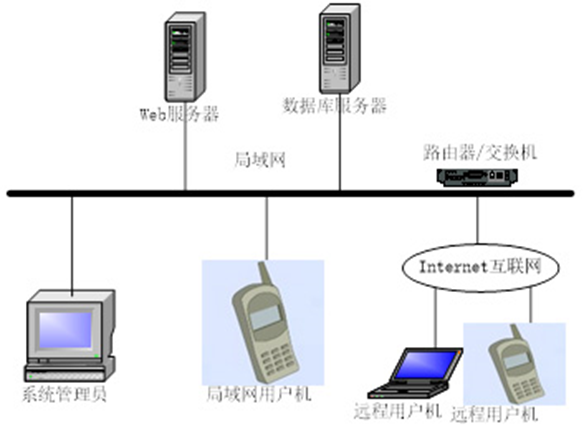


图2.4 物理架构图

2.5开发架构

本次开发使用经典的三层架构，即表示层(UI)、业务逻辑层(BLL)和数据访问层(DAL)。其中表示层(UI)是向用户展现特定业务数据，采集用户的输入信息和操作。本着用户至上，兼顾简洁和不包含任何业务相关的逻辑处理这一原则；

业务逻辑层(BLL)是从DAL中获取数据,在UI显示,从UI中获取用户指令和数据,执行业务逻辑或通过DAL写入数据源，本层作为U层与Ｄ层的桥梁,目的在于展现清晰的函数结构,只负责数据处理传递,不涉及SQL语句和ADO.NET;数据访问层(DAL)是直接操作数据库，针对数据的增添 删除、修改、查找，具体为业务逻辑层或表示层提供数据服务；这一层是专门操作数据库,不考虑数据合法性.数据库错误返回－1,逻辑错误返回0,并告知错误原因,成功返回1。分层架构如下：

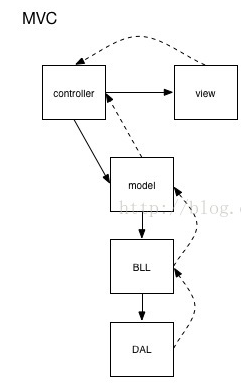


图2.5开发架构图

2.6运行环境设计

硬件环境：

1） 客户机：普通Android手机

 Android Version: Android 4.0以上

 Memory ：1024MB以上

 分辨率：推荐使用768\*1280-320dpi

 Data disk capacity：8192MB以上

2） Web服务器

 CPU： P4 2.4GHz

 Memory ：1GB以上

 HDD：80GB以上

 网卡：KMb/s速度网卡

3） 数据库服务器

 CPU： P4 2.4GHz

 Memory ：1GB以上

 HDD：80GB以上

软件环境是指操作系统，数据库系统以及一些开发工具的选用。本系统开发所需要的软件环境如下：

操作系统：Windows2000或以上版本

数据库：MySQL5.6

开发工具包：JDK Version 1.7

开发环境：MyEclipse6.0

Web服务器：Tomcat7.0

浏览器：IE6.0或以上版本

2.7架构设计

2.7.1系统分层

项目分为了四个层级：模型层、接口层、核心层、界面层。模型层定义了所有的模型；接口层封装了服务器提供的API；核心层处理所有业务逻辑；界面层就处理界面的展示。几个层级之间的关系如下图所示：

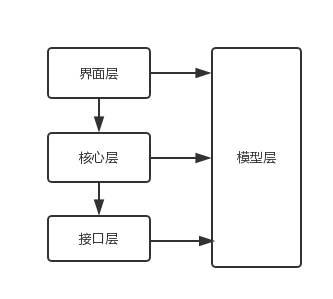


图1.1系统分层图

2.7.2系统包、资源规划的准备工作

（1）服务器端的开发：

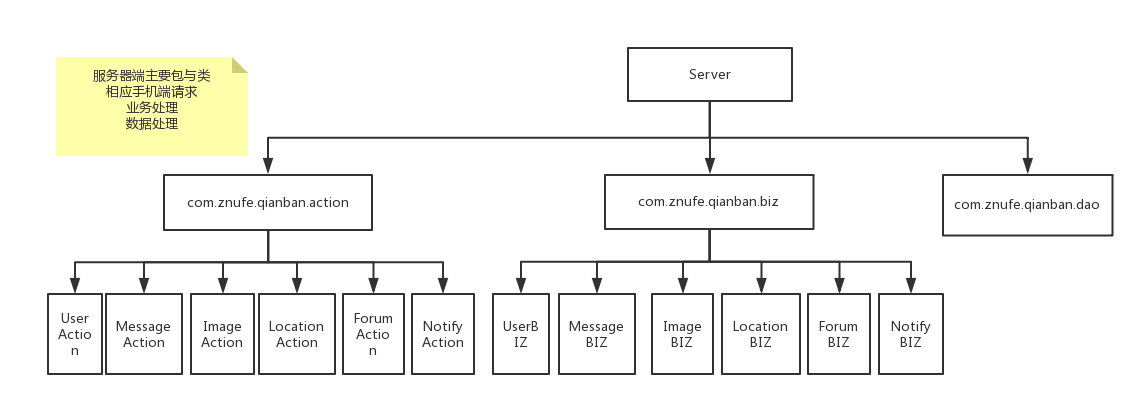


图1.2.1服务器端架构设计

相关包及主要类如下：

com.znufe.qianban.action包：负责响应手机端发出的请求。该包下共有6个action，类及类的功能如下：

UsersAction.java负责用户登录请求；

MessageAction.java负责语音及闹铃类的请求，请求分别为添加语音、闹铃以及查询。

ImageAction.java负责图片类的请求，请求分别为添加图片和图片查询。

LocationAction.java负责地理位置类的请求，请求分别为最新位置信息查询以及历史路线信息查询。

ForumAction.java负责帖子及评论类的请求，请求分别为添加帖子及评论及查询帖子和评论列表。

NotifyAction.java负责通知类的请求，请求系统通知列表。

com.znufe.qianban.biz包：负责业务处理。该包下的类及类的功能如下：

UserBIZ.java负责用户的业务处理，该类有负责用户登录UserLogin()和注销UserLogout()的方法；

MessageBIZ.java负责语音及闹铃的业务处理，该类有四个方法，分别是根据用户名及接收人获取语音列表getVoiceListByURid()和添加新语音addVoice()的方法,和根据用户名（子女）获取闹铃列表getClockListByUid()和添加闹铃addClock()的方法；

ImageBIZ.java负责图片的业务处理，该类有根据用户名（子女）获取图片列表getImageListByUid()和添加新图片addImage()的方法；

LocationBIZ.java负责地理位置信息的业务处理，该类有根据用户名（父母）和时间查询信息getLocationByUT()的方法；

ForumBIZ.java负责论坛的业务处理。该类有根据发帖人获取帖子列表getPostBySend()和根据标题获取帖子列表getPostByTitle()和getAllPost()以及添加新帖子addPost()根据帖子编号获取评论列表getCommentByPid()和添加新评论addComment()的方法等；

NotifyBIZ.java负责系统通知的业务处理。该类有获取所有系统通知getAllNotification()的方法。

com.znufe.qianban.dao包负责数据处理，主要是对数据库的操作。共有6个DAO，命名分别为UsersDAO，MessageDAO，ImageDAO，LocationDAO，ForumDAO，NotifyDAO。

（2）客户端开发：

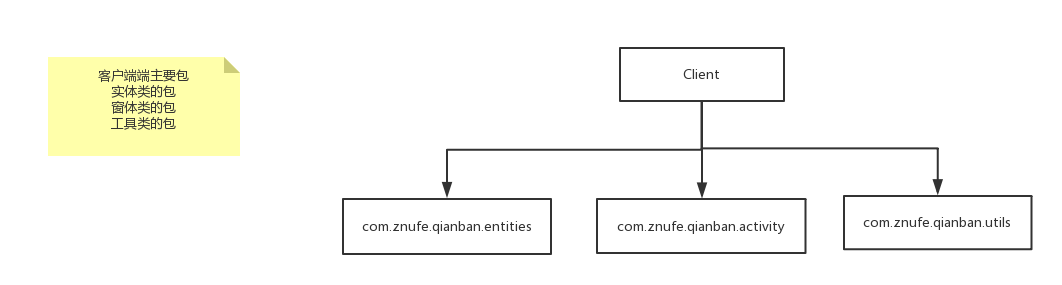


图1.2.2客户端架构设计

相关包及类的内容如下：

com.znufe.qianban.entities包包含本项目涉及的实体类。

子女用户类SonsUser.java，属性有cname是子女的用户名、cpwd是子女的登录密码、ctel是子女的联系方式、csex是子女的性别、cage是子女的年龄、cimg是子女的头像；

父母用户类ParentsUser.java，属性有pname是父母的用户名、ppwd是父母的登录密码、ptel是父母的联系方式、psex是父母的性别、page是父母的年龄、pimg是父母的头像、padd是父母的地址、pbirth是父母生日的时间；

管理员类Admin.java属性有aname是管理员用户名、apwd是管理员密码；

闹钟类Clock.java，属性clday是星期几、clhour是小时、clmin是分钟；

消息推送基类ReqBaseMessage.java，属性有tousername，fromusername，creattime，messagetype，msgid；

图片消息ReqImageMessage.java继承基类ReqBaseMessage.java,属性有图片链接picurl；

地理位置消息ReqLocationMessage.java继承基类ReqBaseMessage.java，属性有纬度location\_x,经度location\_y，地图缩放大小scale，地理位置信息label；

语音消息ReqVoiceMessage.java继承基类ReqBaseMessage.java,属性有语音识别码MediaId，语音格式format；

推文Tweets.java继承基类ReqBaseMessage.java,属性有标题twtitle；

评论Comments.java继承基类ReqBaseMessage.java；

系统通知消息Notification.java继承基类ReqBaseMessage.java；

帖子Posts.java，属性有pmtitle是贴子标题，pmsendid是发帖人id，pmtime是发帖时间，pmcontent发帖内容。

com.znufe.qianban.utils包提供了客户端所用到的工具类，有编码转换类CodeUtil.java，客户端请求服务端的类ConnectWeb.java。

com.znufe.qianban.activity包包含了本项目中出现的窗体类。

欢迎窗体类Welcome.java，登录窗体类ViewLogin.java，注册窗体类ViewSignUP.java，主窗体类ViewMain.java，聊天窗体类ViewMessage.java，闹铃列表窗体类ClockListView.java，闹铃详情窗体类DetailClockView.java，添加老人窗体类AddOlderView.java,，照片列表窗体类ImageListView.java，地理信息窗体类ViewLocation.java，帖子列表窗体类PostListView.java，帖子详情窗体类PostDetailView.java，评论列表窗体类CommentListView.java，电话列表窗体类PhoneListView.java。

3 详细设计

3.1 流程设计

1、用户总体业务流程

首先启动系统，用户进行登录，系统对用户的用户名密码进行验证，通过后用户进入系统，可对系统进行相应操作，最后退出系统，流程结束。流程图如下：

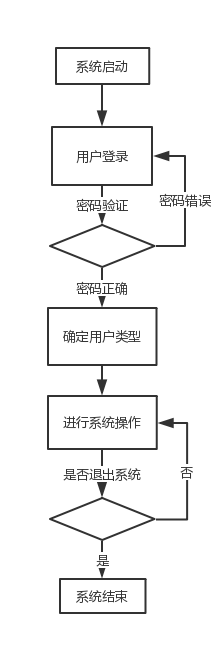


图3.1.1用户总体业务流程

2．用户注册流程

首先用户进入登录注册界面，点击注册链接，进行角色选择后填写个人信息，点击提交，系统对个人信息进行检查，若有效则提交到数据库，注册成功，若无效返回填写信息界面。流程图如下：

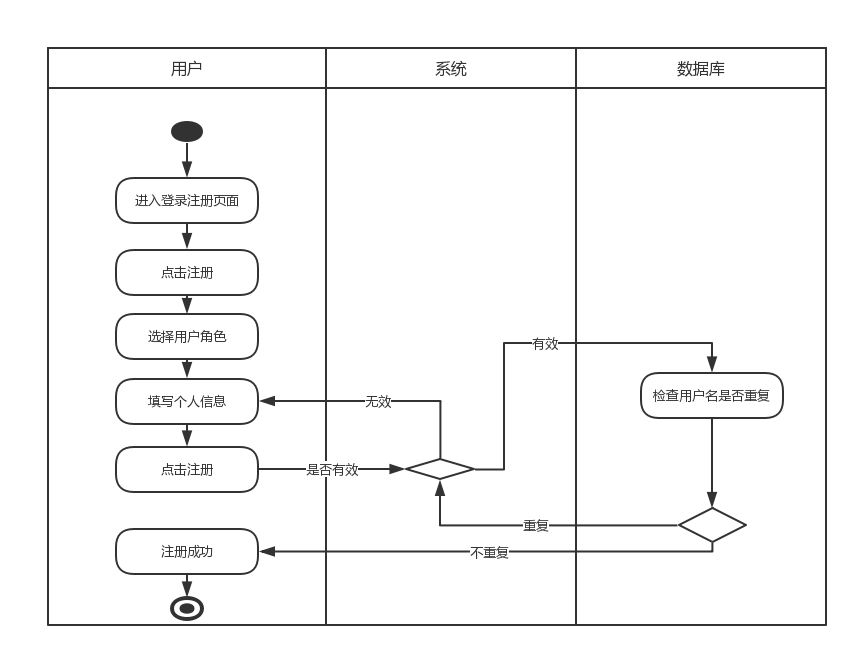


图3.1.2用户注册流程图

3.用户查看实时定位流程

首先进行用户登录,登录成功后，点击查看位置信息，选择要查看的用户，系统获取该用户所要查看用户的位置信息，显示在界面上。流程图如下：

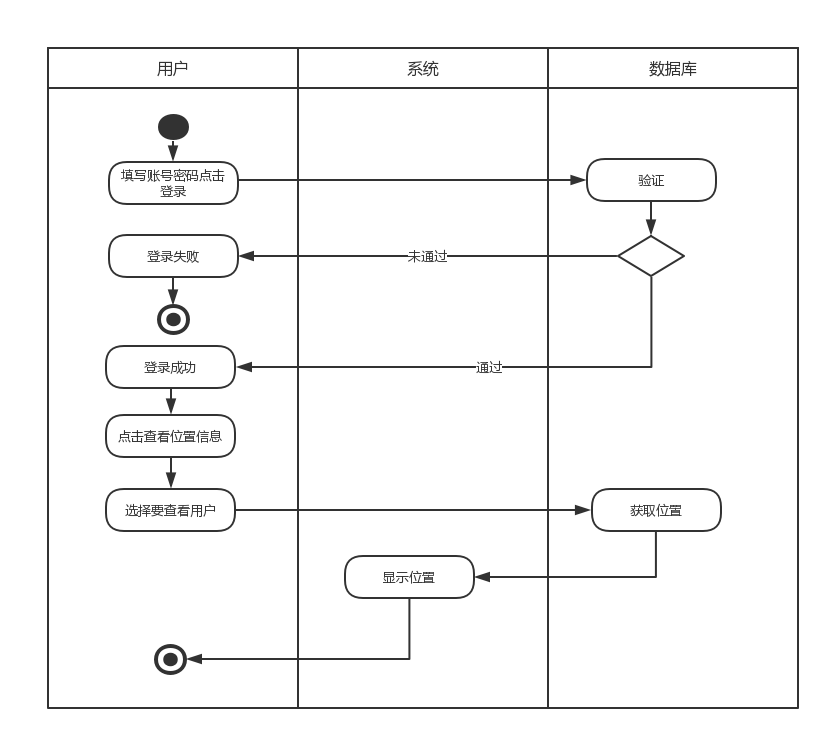


图3.1.3子女用户查看实时定位流程图

4.用户查看历史轨迹流程

首先进行用户登录,登录成功后，点击查看位置信息，选择要查看的用户，系统获取该用户所要查看用户的历史位置信息，显示在界面上。流程图如下：

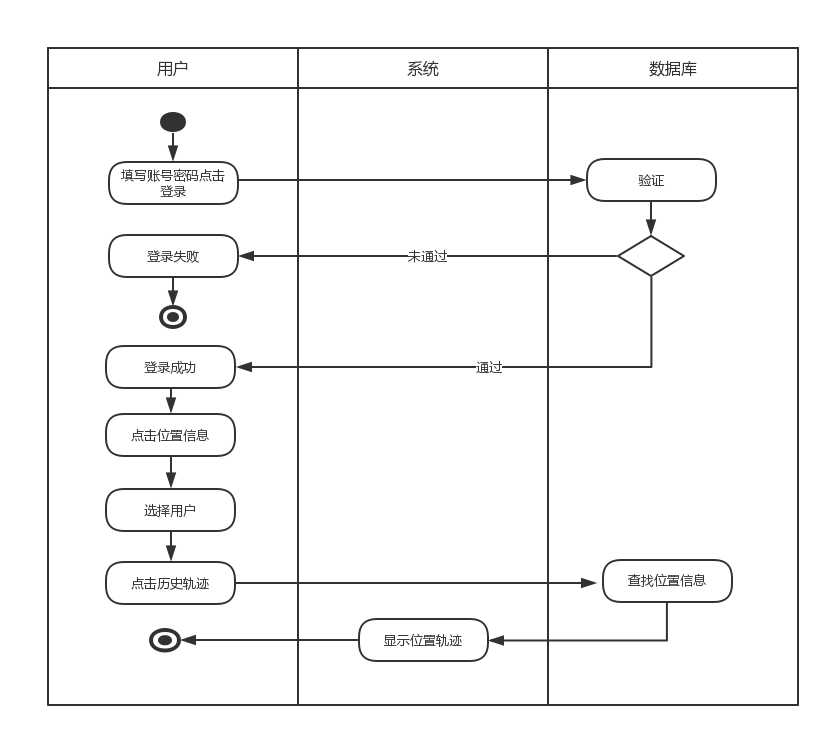


图3.1.4子女用户查看历史轨迹流程图

5.用户上传照片流程

首先进行用户登录,登录成功后，点击消息功能，点击上传照片，选择要上传的照片，点击完成，图片详情显示在界面上。流程图如下：

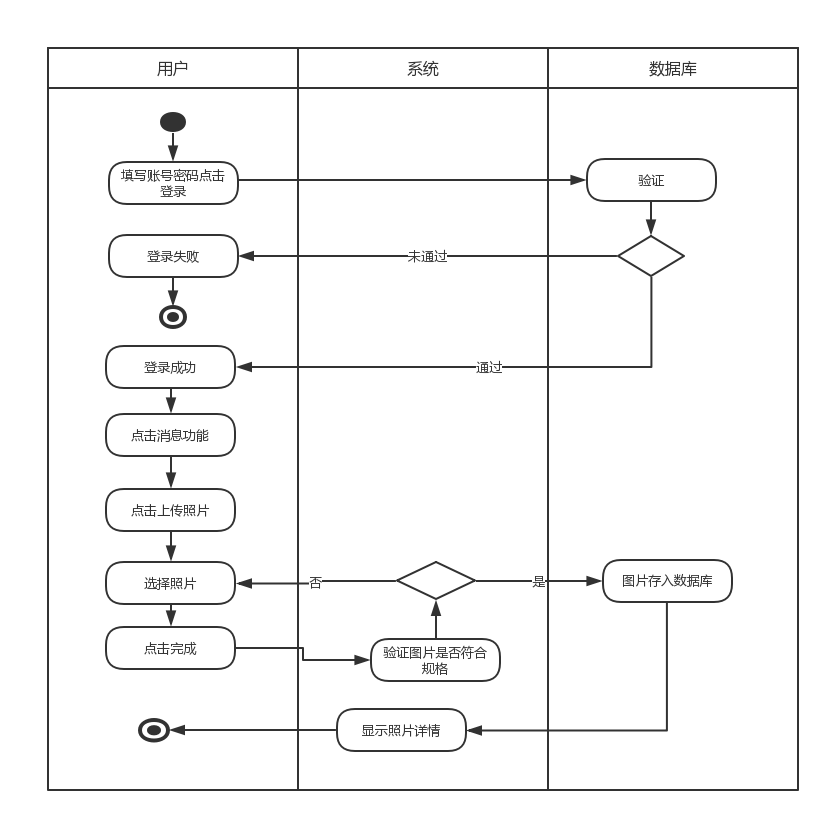


图3.1.5子女用户上传照片流程图

6.用户设置闹钟流程

首先进行用户登录,登录成功后，点击消息功能，点击上传照片，选择用户，点击添加闹钟，填写详细信息，点击完成，闹钟详情显示在界面上。流程图如下：

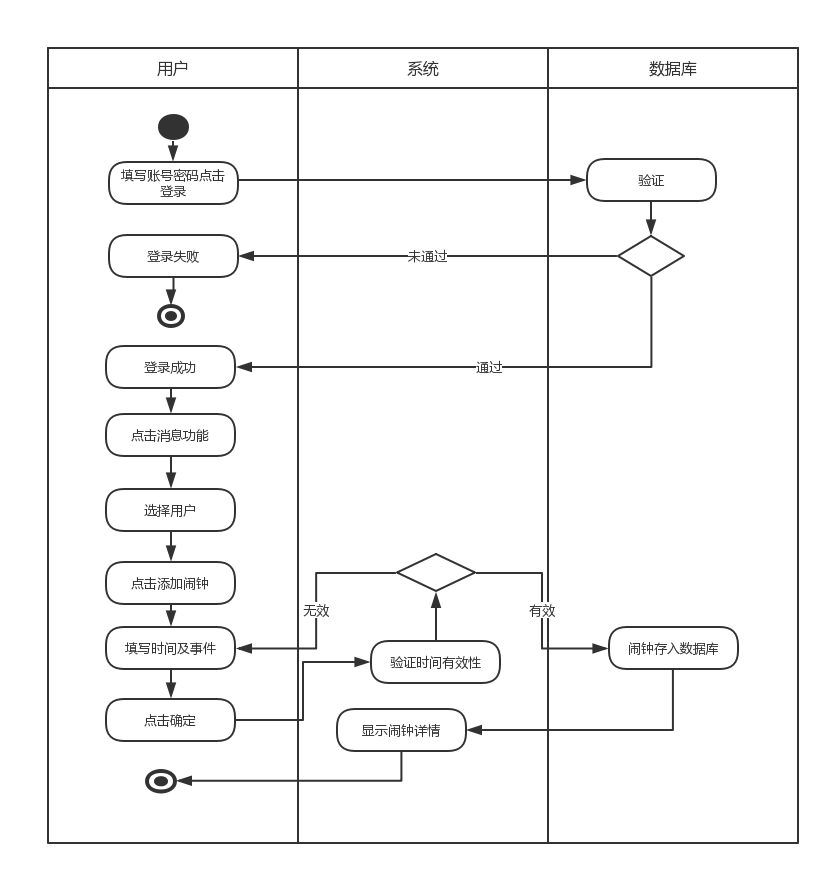


图3.1.6子女用户设置闹钟流程图

7.用户发送语音流程

首先进行用户登录,登录成功后，点击消息功能，选择用户，按住发送语音按钮，松开，完整消息显示在界面上。流程图如下：

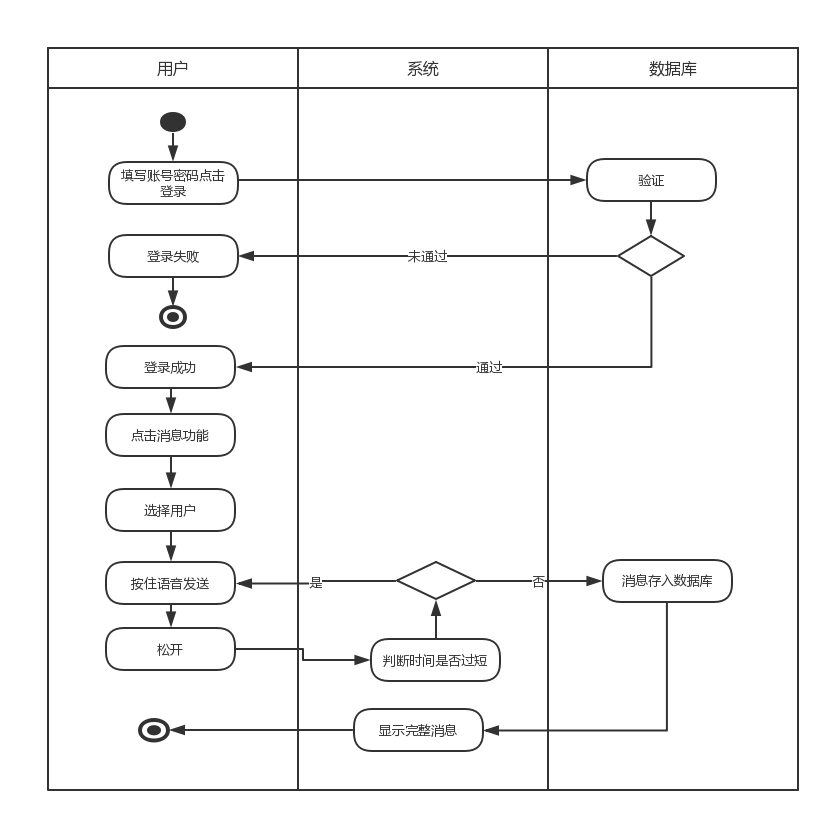
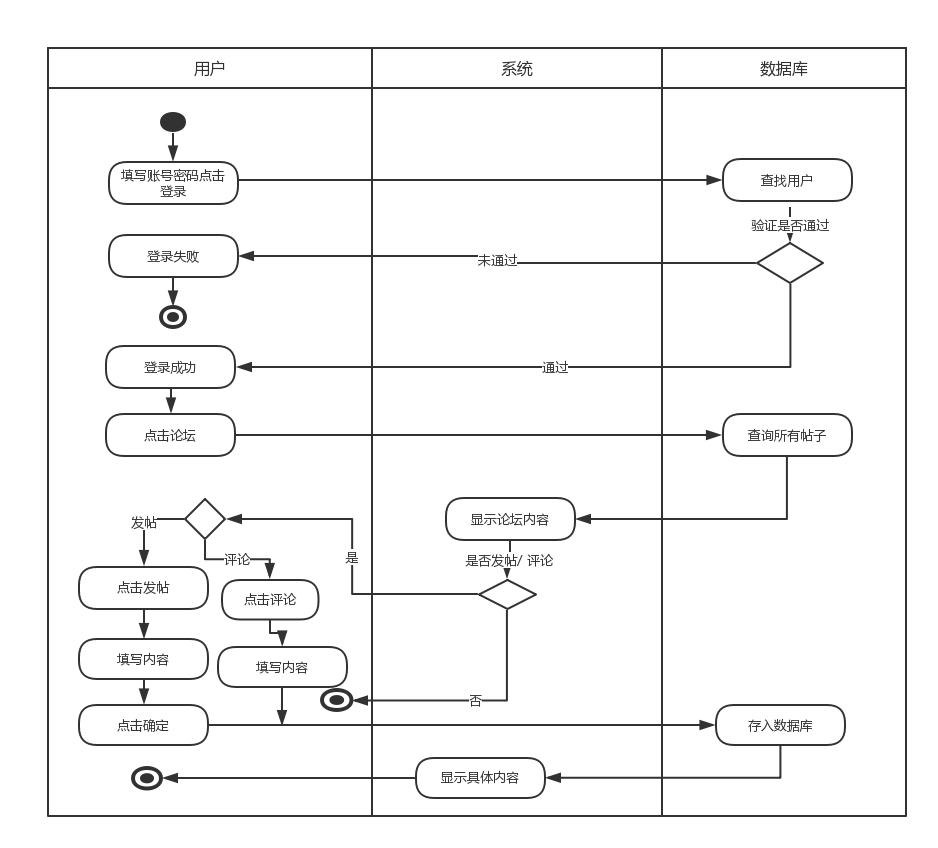


图3.1.7用户发送语音消息流程

8.用户发帖评论流程

首先进行用户登录,登录成功后，点击论坛功能，进行查看帖子，发贴，评论等操作，具体内容显示在界面上。流程图如下：



3.1.8子女用户发帖评论流程图

9.用户查看照片流程

首先进行用户登录,登录成功后，点击查看照片，照片显示在界面上。流程图如下：

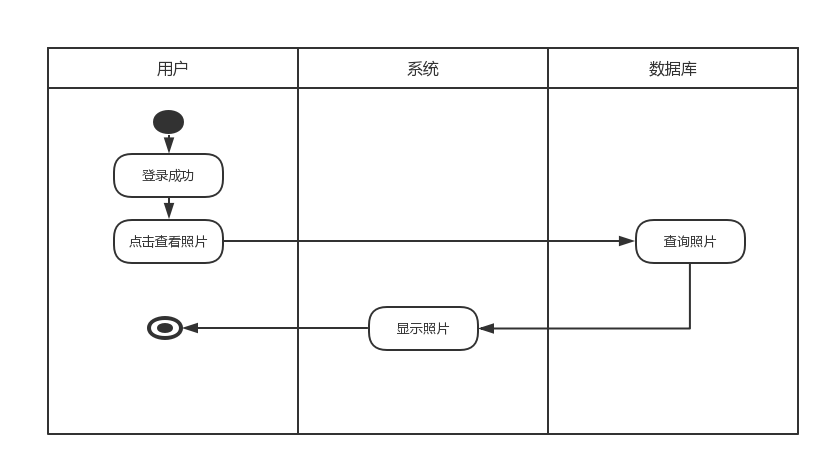


图3.1.9父母用户查看照片流程图

10.用户收听电台流程

首先进行用户登录,登录成功后，点击电台功能，选择电台频道进行收听。流程图如下：

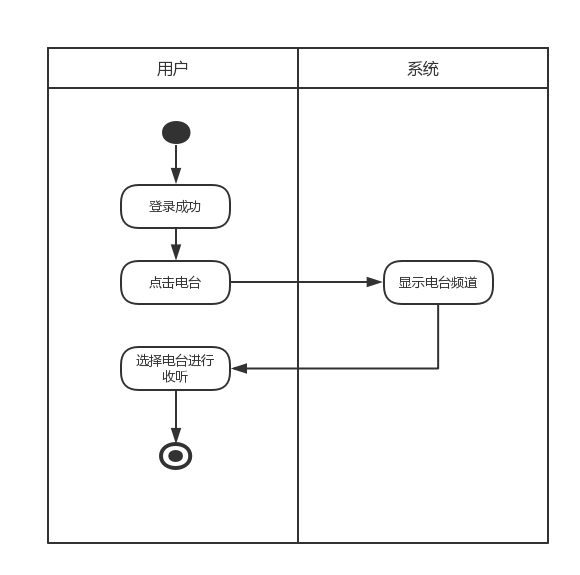


图3.1.10父母用户收听电台流程图

11.管理员总业务流程

首先启动系统，管理员进行登录，系统对用户的用户名密码进行验证，通过后用户进入系统，可对系统进行相应操作，最后退出系统，流程结束。管理员就是对用户系统各个功能中的信息进行简单的管理与操作。流程图如下：

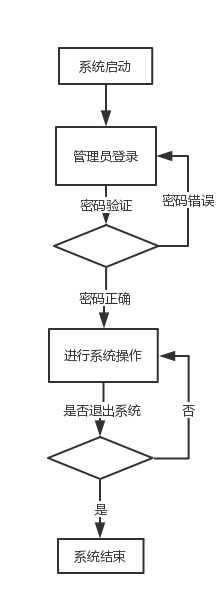


图3.1.11管理员总业务流程图

3.2功能设计

3.2.1模块设计

1、父母版本模块设计

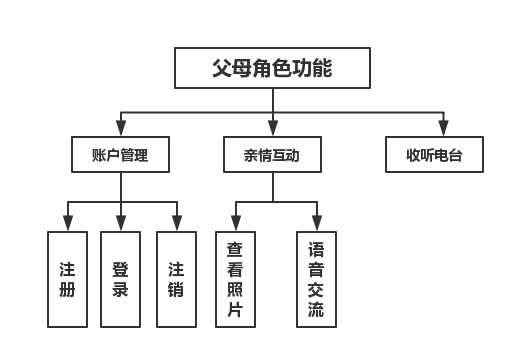


图3.2.1.1父母角色功能

父母版本有3个主要模块，分别是：账户管理亲情互动和收听电台。在账户管理模块中，用户进入账户界面，新用户先注册再登录。在亲情互动模块中，父母可以查看子女发布的照片同时也可以进行语音交流。在收听电台模块中可以进行电台收听，避免无聊。

2、子女版本模块设计

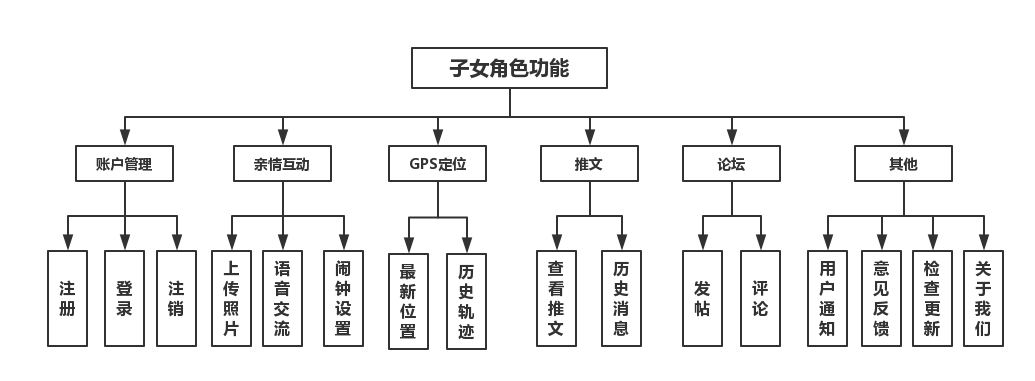


图3.2.1.2 子女角色功能

子女版本有6个主要模块，分别是：账户管理、亲情互动、GPS定位、推文、论坛和其他。在账户管理模块中，用户进入账户界面，新用户先注册再登录。在亲情互动模块中，子女可随时上传生活图片（父母端可实时观看，类似刷动态），双方可进行语音交流，同时在可进行闹钟设置、提醒父母记得吃药。在GPS定位模块中，子女可实时观看父母的最新定位以免发生危险时因信息的缺失而不能及时赶到，子女还可以在历史位置中看到父母去过哪些地点。在论坛模块中，用户可以发帖和评论，谈论的内容包括但不限于父母的生活日常等。在推文模块中，用户可以看到管理员发布的有关老年人身体健康和饮食习惯等的美文，用户可以通过查看历史消息来观看所有文章。在其他模块中，用户可以查看历史消息、用户通知、关于我们、检查更新等。

3.2.2界面设计

1、用户注册登陆



图3.2.2.1.1 用户登陆界面

下载APP后，打开“牵伴”APP后出现该界面，新用户点击注册按钮进行注册，已注册过的用户点击登录进行登录。



图3.2.2.1.2 用户角色选择界面

点击注册按钮，跳转到角色选择界面，进行角色选择，选择“我是父母”或者“我是子女”。



图3.2.2.1.3 父母注册界面

选择父母角色进行信息填写，填写完成后点击完成注册。



图3.2.2.1.3 子女注册界面

选择子女角色进行信息填写，填写完成后点击完成注册。

2、父母版本界面

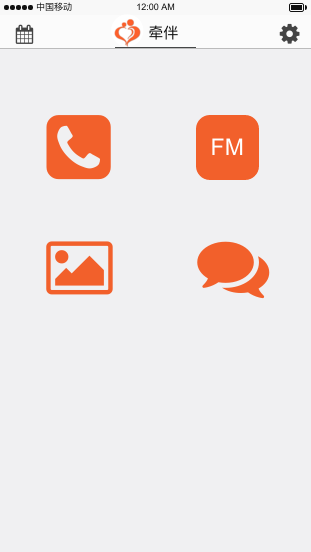


图3.2.2.2.1 父母版本主界面

上图是父母角色功能界面，包含打电话、听电台、查看动态和实时聊天等功能。

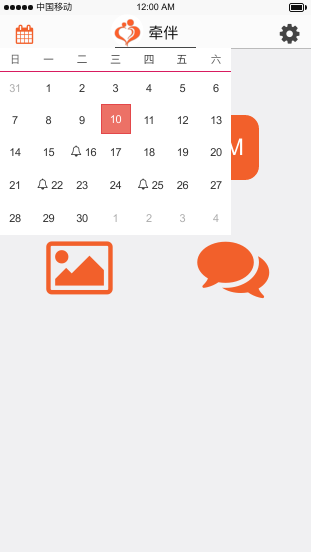


图3.2.2.2.2 查看日历界面

点击日历按钮，可以看到今日日期。

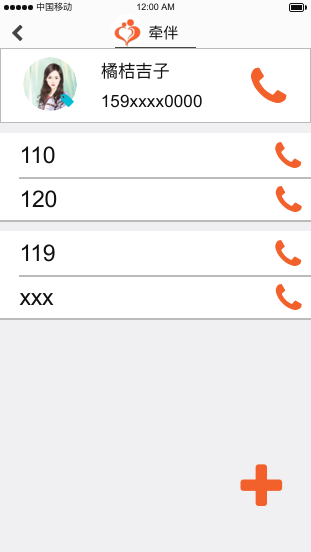


图3.2.2.2.3 一键通讯界面

点击电话图标，可以与已预存的联系人“一键通讯”。



图3.2.2.2.4 添加号码界面

点击添加号码，可以进行号码添加。



图3.2.2.2.5 语音聊天界面

点击聊天图标，可以与子女语音聊天。

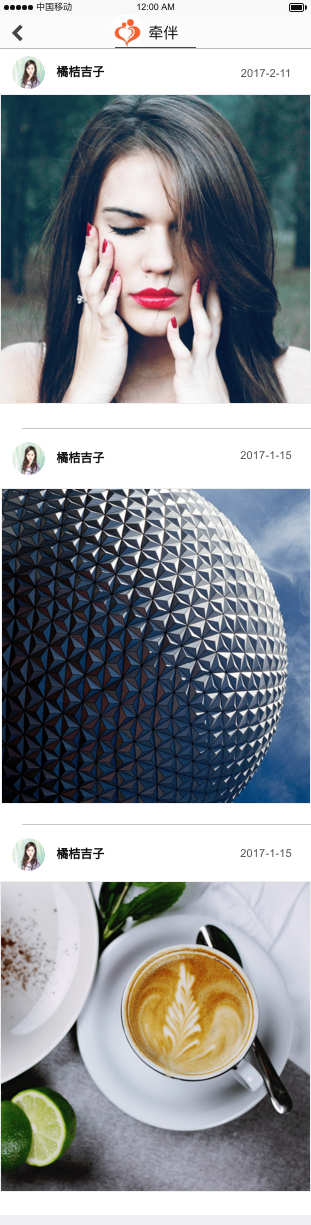


图3.2.2.2.6 查看子女动态界面

点击图片图标，可以查看子女上传动态。



图3.2.2.2.7 电台界面

点击FM图标，可以实时听广播。

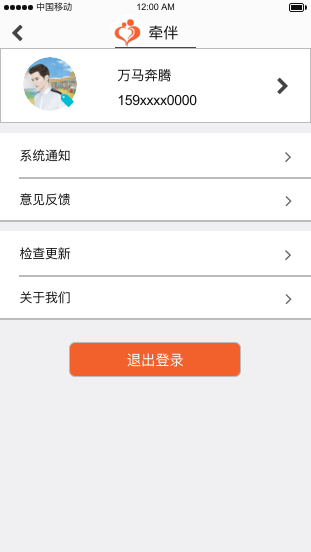


图3.2.2.2.8 “我的”界面

点击右上方设置按钮，查看系统通知、进行意见反馈、检查软件更新、退出。

3、子女版本界面



图3.2.2.3.1 子女版本主界面

上图是子女界面首页。



图3.2.2.3.2 查看日历界面



图3.2.2.3.3 今日推文界面



图3.2.2.3.4 历史推文界面

点击右上角图标，进入“查看历史推文”界面，用户可自行查看历史推文。



图3.2.2.3.5 语音聊天界面

点击聊天图标，与绑定的父母进行实时语音聊天。



图3.2.2.3.6 切换聊天角色（老人）界面

可切换聊天角色。

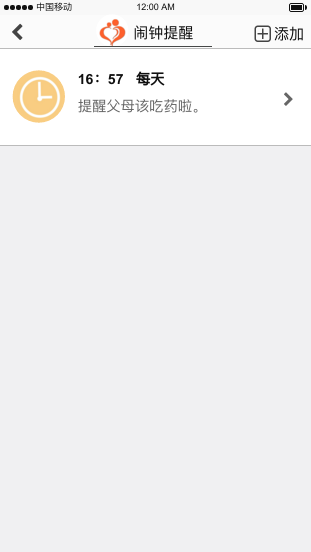


图3.2.2.3.7 闹钟提醒界面

点击语音聊天界面右下方的按钮进行闹钟设置，如某月某日该吃药等。

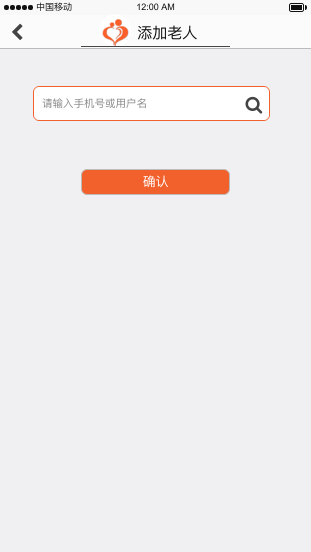


图3.2.2.3.8 添加老人界面

点击聊天界面右上方的添加按钮，可以添加老人，实现一对多的关系。

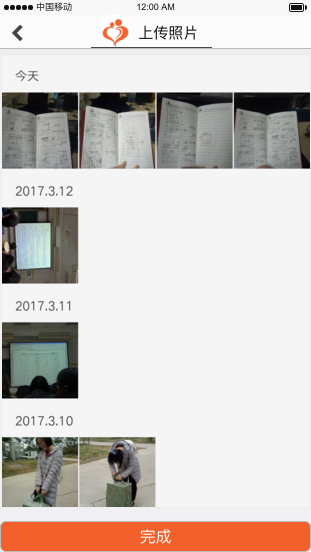


图3.2.2.3.9 上传照片界面

点击上传照片进行动态更新。



图3.2.2.3.10 成功上传照片界面

选好图片点击完成，图片上传成功。

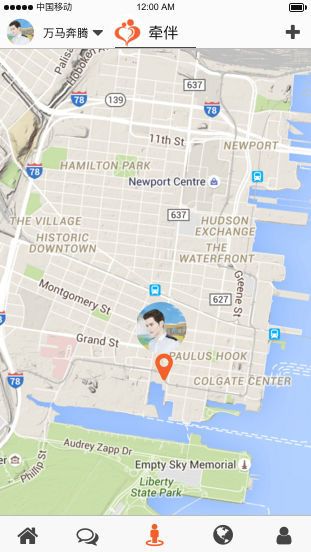


图3.2.2.3.11 GPS实时定位界面

点击定位图标，对老人进行实时定位。



图3.2.2.3.12 GPS查看历史轨迹界面

点击“+”图表中的历史轨迹，查看老人的历史轨迹。



图3.2.2.3.13 切换定位老人界面

切换不同老人进行定位。

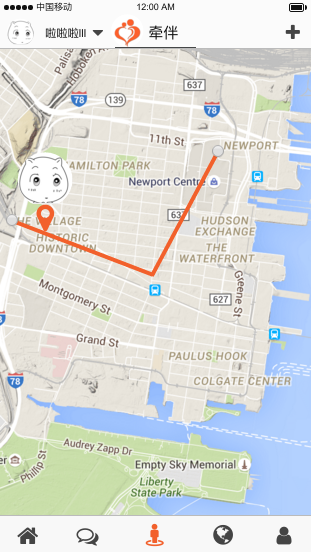


图3.2.2.3.14 查看历史轨迹界面

查看不同老人的历史轨迹。



图3.2.2.3.15 实时定位所有老人界面

对两个老人同时定位。



图3.2.2.3.16 查看所有老人历史轨迹界面

查看在同一底图内查看不同老人的历史轨迹。



图3.2.2.3.17 论坛界面

点击论坛图标，进入论坛界面。



图3.2.2.3.18 发帖界面

点击“+”按钮，开始发帖。



图3.2.2.3.19 评论界面

点进帖子后，可进行评论及查看他人评论。

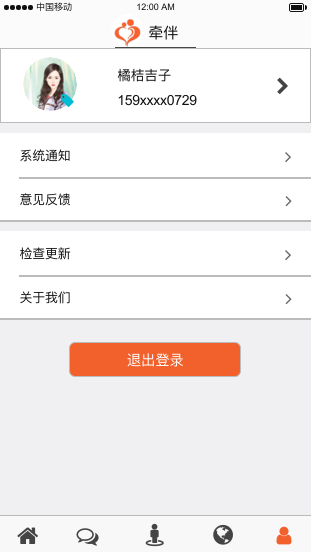


图3.2.2.3.20 “我的”界面

在这个功能模块可以查看系统通知、进行意见反馈、查看更新和退出登录等。

3.3接口设计

3.3.1用户接口

用户接口用来说明将向用户提供的命令和它们的语法结构，以及软件的回答信息。

在本系统中，用户接口设计如下：

表3.3.1 用户接口表

|  |  |
| --- | --- |
| 接口 | 功能 |
| 填写及修改个人信息 | 填写修改完善用户个人信息 |
| 查看历史消息 | 用户可查看系统历史消息，历史推文等 |
| GPS定位 | 用于子女查看父母当前地址、历史轨迹 |
| 亲情互动 | 用于子女与父母语音交流、上传或查看动态 |
| 一键求助 | 当老人遇到紧急情况时可按一键求助拨打电话 |
| 发帖 | 用户间可以发帖交流赡养老人的心得 |
| 评论 | 用户间可以发帖交流赡养老人的心得 |

3.3.2外部接口

外部接口用来说明本系统同外界的所有接口的安排包括软件与硬件之间的接口、本系统与各支持软件之间的接口关系。

在本系统中，外部接口设计如下：

* 安卓2.0以上的操作系统。
* 操作系统：Windows2000或以上版本；
* 数据库：MySQL5.6；
* 运行环境： Eclipse6.0；
* 服务器：Tomcat7.0；

软件之间交换数据的目的：达到信息同步的效果。

3.3.3内部接口

内部接口用来说明本系统之内的各个系统元素之间的接口的安排。

在本系统中，各模块之间主要采用函数调用,参数传递,返回值的方式进行信息传递，具体如下图：

父母版本设有账户管理模块（个人注册、个人登陆、注销）、亲情互动模块（查看子女动态、语音交流）、收听电台等功能模块。

子女版本设有账户管理模块（个人注册、个人登陆、注销）、亲情互动模块（上传照片、语音交流、闹钟设置）、GPS定位模块（最新位置、历史轨迹）、论坛模块（发帖、评论）、推文模块（查看推文、历史推文）、其他模块（用户通知、意见反馈、检查更新、关于我们）等功能模块。

管理员版本设有用户管理模块（添加用户、搜索用户、修改用户信息、删除用户）、管理员管理模块（添加角色、修改角色信息、删除角色、权限管理）、推文管理模块（添加推文、搜索推文、删除推文等）、用户通知模块、系统管理（图片管理、数据字典）。

|  |  |
| --- | --- |
| 接口 | 功能 |
| 用户注册 | 向数据库添加用户信息 |
| 添加角色 | 向数据库添加管理员信息 |
| 添加推文 | 向数据库添加新的推文并发布 |
| 修改个人信息 | 向数据库修改用户信息 |
| 修改角色信息 | 向数据库修改管理员信息 |
| 添加帖子 | 向数据库添加新的帖子信息 |
| 评论帖子 | 向数据库添加帖子的评论信息 |
| 历史推文 | 向数据库查询历史推文信息 |
| 删除用户 | 删除数据库中的用户信息 |
| 删除角色 | 删除数据库中的管理员信息 |
| 删除推文 | 删除数据库中的推文 |
| 删除帖子 | 删除数据库中的帖子 |
| 删除评论 | 删除数据库中的评论 |

3.4非功能性设计

3.4.1 非功能整体需求

对于APP风格方面的需求，应该尽量做到APP结构简单明了，逻辑条理清晰，页面精美、功能实用；而对于各页面方面的要求，则应该设计构思力求精巧、布局要求简捷合理、APP整体设计风格统一。

（1）界面设计精美、友好，信息化设计，体现企业级应用的特色。

（2）更快的信息访问速度、更加省时和智能化的用户界面，老年版界面则应美观大方简洁明了。

（3）更简便、智能化程度更高的浏览查询流程。

（4）后台管理功能强大、录入效率高。

（5）易于查看用户建议和评论以及后台信息。

3.4.2 性能需求

（1）用户数要求: 系统将能适应1000个平均在线用户；在高峰时段，系统将能适应1万个用户。

（2）响应时间需求：由硬件条件的不同决定，但用户提交了任务之后，客户端对任务的响应时间不能超过6秒。

（3）更新处理时间需求：由硬件条件及处理的数据量决定，但用户向系统提交信息后，系统将在5秒内向用户显示确认信息。

（4）数据的转换和传送时间需求：根据具体的转换和传送数据不同所需的时间不同，系统的界面将在5秒内全部加载完成。

（5）精度需求：用户输入精度的要求取决于相应功能所需参数的精度要求，管理员输入的精度要求主要取决于数据库的相关数据类型要求，当用户或管理员输入的参数与要求不符时将得到提示并重新进行操作。

3.4.3 软件质量要求

（1）可用性：除维护时间外，APP前台全天24小时对用户可用。后台管理系统只对管理员开放，且限定IP地址登陆。

（2）健壮性：由于不可见原因导致用户操作中断的，用户可以在重新连接后继续操作。

（3）可伸缩性：当用户增加时，系统也将维持较高的服务质量。

（4）易用性：APP系统可操作性强，易于老人使用，便于子女使用。

（5）可维护性：在修改bug、调整功能、提高适应性等方面进行系统维护时，系统易于进行修改点定位并实施修改。

3.4.4 安全保密性要求

（1）用户个人信息保密需求：所有涉及功能信息或个人身份信息的网络事务，都要按照规定进行加密操作。

（2）用户操作安全需求：除浏览菜单外，所有用户都必须登录后才能进行其他如上传照片、发送语音等操作。

（3）用户安全需求：用户的登陆受手机系统访问控制策略的限制。

（4）系统安全需求：只有授权成为管理员的才能登陆管理后台完成相关数据操作。

3.4.5 数据管理能力要求

（1）数据备份需求：保证数据每周完全备份，每天进行局部备份，保证数据的安全性。

（2）数据库日志需求：记录数据操作日志，并有专人负责定期查看，有效管理用户数据。

（3）数据库负载需求：数据库负载要能够支持系统的正常运行。

（4）数据容量需求：及时检查数据库占用空间，保证有足够的数据库容量储存数据。

3.4.6 故障处理要求

（1）用户非法操作：当用户进行如输入不合理数据的操作时，系统能够合理提示而非导致系统错误或程序停止运行。

（2）服务器和网络故障：程序运行时，对服务器和网络通信故障能够识别并提示，当故障排除后，程序恢复正常运行。

（3）数据库需求：数据库要求有灾难备份机制，以防止数据的全部丢失。若运行时遇到不可恢复的系统错误，系统也必须保证数据库完好无损。

（4）应用日志需求：应用程序应能提供易于查看的日志，便于系统的维护、调试、错误定位等。

3.4.7 其他专门要求

易于老人使用：老人版界面应简洁大方、清晰明了，能够让手指不灵活、视力不太好的老人使用。

3.5 数据库设计

在UML建模分析和系统功能分析后，就需要在此前的基础上建立相关的数据库，设计出能够满足用户需求的各种实体以及它们之间的关系，这些实体包含各种具体信息，以及通过相互作用形成的数据的流动。数据库设计包括需求分析、概念设计、逻辑设计、物理设计和实施维护。

3.5.1物理数据库概述

数据库设计的目标就是为用户的应用系统提供一个高效数据库系统。为了保证数据库设计更合理，数据库的设计工作通常分阶段进行，在不同的阶段采用不同的方法完成不同的设计内容。

数据库设计分为以下6个阶段：

（1）需求分析阶段。收集和分析用户对系统的信息需求和处理需求，得到设计系统所必需的需求信息，建立系统的需求说明文档。

（2）概念结构设计阶段。通过对用户的需求进行综合、归纳与抽象，形成一个独立于具体DBMS的概念模型。

（3）逻辑结构设计阶段。在概念模型的基础上导出一种DBMS支持的逻辑数据库模型，该模型应满足数据库存取、一致性及运行等各方面的用户需求。

（4）物理结构设计阶段。为逻辑数据模型选取一个最适合应用环境的物理结构。

（5）数据库实施阶段。根据逻辑设计和物理设计的结果建立数据库，编写与调试应用程序，将数据录入到数据库中，同时进行数据库系统的试运行。

（6）数据库运行与维护阶段。数据库应用系统试运行后，即可以投入正式运行。在此过程中必须不断地对其进行评价、调整与修改。

数据库采用本地数据库。本数据库中有12张表：子女信息表、父母信息表、管理员信息表、闹钟信息表、语音信息表、地图信息表、图片信息表、发帖信息表、评论信息表、推文信息表、系统通知信息表、意见反馈信息表。每个表均只有一张表，存储其对应的所有信息。

1、概念结构设计

数据库的概念设计是指针对现实世界，通过对其中信息实体的收集、分类、聚集和概括，建立数据库概念结构的过程，应该反映系统组织的业务模式、信息结构、信息间的相互制约关系，以及对信息存储、查询和加工的处理要求等。

“牵伴”APP中存在的实体是子女、父母和管理员。根据这个，设计出针对不同使用者的功能。

2、逻辑结构设计

数据库逻辑结构设计阶段就是指出所需的实体和关系，并进行实体规范化。将上面的数据库概念结构转化为数据库所支持的逻辑数据模型。

本系统的数据库总共有12个表。

（1）子女信息表。

这个表包含了子女用户的基本信息，下面来介绍一下该表中的各个属性。

cname是子女的用户名

cpwd是子女的登录密码

ctel是子女的联系方式（主键）

csex是子女的性别

cage是子女的年龄

cimg是子女的头像

（2）父母信息表。

pname是父母的用户名

ppwd是父母的登录密码

ptel是父母的联系方式（主键）

ctel是子女的联系方式（外键）

psex是父母的性别

page是父母的年龄

pimg是父母的头像

padd是父母的地址

pbirth是父母生日的时间

（3）管理员信息表。

aid是管理员编号（主键）

aname是管理员用户名

apwd是管理员密码

（4）闹钟信息表。

clid是闹钟编号（主键）

clday是星期几

clhour是小时

clmin是分钟

（5）语音信息表。

vid是语音编号（主键）

talker是进行语音的双方

content是语音内容

lvbuffer是缓存内容

（6）地图信息表。

mid是地图编号（主键）

mlot是经度位置

mlat是维度位置

ptel是父母手机号（对谁定位）

mtime是当前时间

（7）图片信息表。

pid是图片编号（主键）

ptalker是发送图片的人

ptime是发送图片的时间

pimagpath是存储图片的地址

（8）发帖信息表。

pmid是贴子编号（主键）

pmtitle是贴子标题

pmtalker是发帖人

pmtime是发帖时间

（9）评论信息表。

reid是评论编号（主键）

pmid是贴子编号（外键）

retalker是评论人

recontent是评论内容

retime是评论时间

（10）推文信息表。

twid是推文编号（主键）

aid是管理员编号（外键）

twtitle是推文标题

twcontent是推文内容

twtime是推文时间

（11）系统通知信息表。

inid是通知编号（主键）

aid是管理员编号（外键）

intitle是通知标题

incontent是通知内容

intime是通知时间

（12）意见反馈信息表。

sid是意见编号（主键）

cid是用户编号

scontent是意见内容

3、物理结构设计

数据库物理设计的任务是为上一阶段得到的数据库逻辑模式，即数据库的逻辑结构选择合适的应用环境的物理结构，既确定有效地实现逻辑结构模式的数据库存储模式，确定在物理设备上所采用的存储结构和存取方法，然后对该存储模式进行性能评价、修改设计，经过多次反复，最后得到一个性能较好、可用性较高的存储模式。数据库物理设计内容包括记录存储结构的设计，存储路径的设计。由于是测试实验，我们的数据库选用了MySQL的数据库，并将数据库架设在一台电脑上。

4、数据库实施

数据库的实施主要是根据逻辑结构设计和物理结构设计的结果，在计算机系统上建立实际的数据库结构、导入数据并进行程序的调试。它相当于软件工程中的代码编写和程序调试的阶段。用具体的DBMS提供的数据定义语言(DDL)，把数据库的逻辑结构设计和物理结构设计的结果转化为程序语句，然后经DBMS编译处理和运行后，实际的数据库便建立起来。

5、数据库运行与维护

数据库系统投入正式运行，意味着数据库的设计与开发阶段的基本结束，运行与维护阶段的开始。数据库的运行和维护是个长期的工作，是数据库设计工作的延续和提高。

在数据库运行阶段，完成对数据库的日常维护，工作人员需要掌握DBMS的存储、控制和数据恢复等基本操作，而且要经常性地涉及物理数据库、甚至逻辑数据库的再设计，因此数据库的维护工作仍然需要具有丰富经验的专业技术人员(主要是数据库管理员)来完成。

数据库的运行和维护阶段的主要工作有：

（1）对数据库性能的监测、分析和改善

（2） 数据库的转储和恢复

（3）维持数据库的安全性和完整性

（4）数据库的重组和重构

### 3.5.2数据库表设计

“牵伴”APP数据库总共包括12个表，它们分别是子女信息表、父母信息表、管理员信息表、闹钟信息表、语音信息表、地图信息表、图片信息表、发帖信息表、评论信息表、推文信息表、系统通知信息表、意见反馈信息表。

（1）应聘人员信息表：

表3.5.2.1子女信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子女信息表（child） | | | | | |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 是否为主键/外键 |
| ctel | varchar | 50 | 子女电话 | 否 | 主键 |
| cname | varchar | 50 | 子女用户名 | 否 |  |
| cpwd | varchar | 50 | 子女登录密码 | 否 |  |
| csex | varchar | 2 | 性别 | 是 |  |
| cage | int | 11 | 年龄 | 是 |  |
| cimg | varchar | 100 | 头像 | 是 |  |

（2）父母信息表：

表3.5.2.2父母信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 父母信息表（parents） | | | | | |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 是否为主键/外键 |
| ptel | varchar | 50 | 父母电话 | 否 | 主键 |
| ctel | varchar | 50 | 子女电话 | 否 | 外键 |
| pname | varchar | 50 | 父母用户名 | 否 |  |
| ppwd | varchar | 50 | 父母登录密码 | 否 |  |
| psex | varchar | 2 | 性别 | 是 |  |
| page | int | 11 | 年龄 | 是 |  |
| pimg | varchar | 100 | 头像 | 是 |  |
| padd | varchar | 50 | 父母地址 | 否 |  |
| pbirth | varchar | 20 | 父母生日 | 是 |  |

（3）管理员信息表：

表3.5.2.3管理员信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| aid | int | 11 | 管理员编号 | 否 | -1 |
| aname | varchar | 50 | 管理员用户名 | 否 | 0 |
| apwd | varchar | 20 | 管理员密码 | 否 | 0 |

（4）闹钟信息表：

表3.5.2.4闹钟信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| clid | int | 11 | 闹钟编号 | 否 | 主键 |
| clday | varchar | 20 | 星期几 | 否 |  |
| clhour | varchar | 20 | 小时 | 否 |  |
| clmin | varchar | 20 | 分钟 | 否 |  |

（5）语音信息表：

表3.5.2.5语音信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| vid | int | 11 | 语音编号 | 否 | 主键 |
| vtalker | varchar | 50 | 进行语音的人 | 否 |  |
| vcontent | varchar | 50 | 语音内容 | 否 |  |
| lvbuffer | varchar | 50 | 缓存 | 是 |  |

（6）地图信息表：

表3. 5.2.6地图信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| mid | int | 11 | 地图编号 | 否 | 主键 |
| mlot | varchar | 50 | 经度位置 | 否 |  |
| mlat | varchar | 50 | 纬度位置 | 否 |  |
| ptel | varchar | 50 | 父母联系方式 | 否 |  |
| mtime | datetime | - | 当前时间 | 是 |  |

（7）图片信息表：

表3. 5.2.7图片信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| pid | int | 11 | 图片编号 | 否 | 主键 |
| ptalker | varchar | 50 | 发送图片的人 | 否 |  |
| pimagpath | varchar | 50 | 图片路径 | 否 |  |
| ptime | datetime | - | 发送图片时间 | 否 |  |

（8）发帖信息表：

表3. 5.2.8发帖信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| pmid | int | 11 | 贴子编号 | 否 | 主键 |
| Pmtitle | varchar | 50 | 贴子标题 | 否 |  |
| pmtalker | varchar | 50 | 发帖人 | 否 |  |
| pmcontent | varchar | 50 | 贴子内容 | 否 |  |
| pmtime | datetime | - | 当前时间 | 是 |  |

（9）评论信息表：

表3. 5.2.9评论信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| reid | int | 11 | 评论编号 | 否 | 主键 |
| Pmid | int | 11 | 贴子编号 | 否 | 外键 |
| retalker | varchar | 50 | 评论人 | 否 |  |
| recontent | varchar | 50 | 评论内容 | 否 |  |
| retime | datetime | - | 评论时间 | 是 |  |

（10）推文信息表：

表3. 5.2.10推文信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| twid | int | 11 | 推文编号 | 否 | 主键 |
| aid | int | 11 | 管理员编号 | 否 | 外键 |
| twtitle | varchar | 50 | 推文标题 | 否 |  |
| twcontent | varchar | 50 | 推文内容 | 否 |  |
| twtime | datetime | - | 推文时间 | 是 |  |

（11）系统通知信息表：

表3. 5.2.11系统通知信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| inid | int | 11 | 系统通知编号 | 否 | 主键 |
| aid | int | 11 | 管理员编号 | 否 | 外键 |
| Intitle | varchar | 50 | 通知标题 | 否 |  |
| incontent | varchar | 50 | 通知内容 | 否 |  |
| intime | datetime | - | 通知时间 | 是 |  |

（12）意见反馈信息表：

表3. 5.2.12意见反馈信息表设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 描述 | 可否为空 | 主键/外键 |
| sid | int | 11 | 意见编号 | 否 | 主键 |
| cid | int | 11 | 用户 | 否 | 外键 |
| scontent | varchar | 50 | 意见内容 | 否 |  |

3.6补充说明

数据库设计会在开发过程中不断修改，直至开发完成。

4 模块与代码设计

4.1 模糊搜索算法

通过查阅资料，本组认为可以采用一种分词算法并基于 KMP 算法改进出几个函数实现一种按模糊相似度进行模糊查找的模糊搜索算法，其主要原理是通过用户设定模糊比较级别来对字符串的模糊匹配进行数据筛选，从而获得用户所需信息。

4.1.1算法中的相关概念

⑴ 模糊相似度： 本文将用户查询关键字与数据库相应字段按照字符进行比较， 相同字符所占比例称为查询字段的模糊相似度，实际应用时本文将模糊相似度扩大10倍后取整（这样做的目的是方便用户接口处精度的设置）。

⑵ 分词： 将用户输入的一个完整句子按照词典有实体意义的词进行拆分的过程称之为分词， 实现这一过程所用的方法称之为分词算法。

⑶ 模糊比较级别： 把用户查询过程中对查询精度的要求按照上面定义的模糊相似度进行区分，并留有系统设置接口，用户以系统管理员的身份通过设定的模糊相似度影响查询精度这一过程称之为模糊比较级别的设定。

4.1.2算法的整体设计

算法的整体设计过程如下：

（1）将用户输入的关键字进行分词处理；

（2）原串（用户输入的关键字）与数据库中的串（数据库中有实体意义的字段）进行模糊比较；

（3）根据步骤（2）的比较结果得出模糊相似度；

（4）将模糊匹配有关联的串入队列；

（5）判断模糊相似度，如果模糊相似度符合查找要求则执行步骤（4）；如果模糊相似度达不到要求则将要求降低一级转向执行（2），相似度值加 1 ；

（6）相似度值达到限制最大值如有匹配项则按照队列的元素输出否则提示模糊匹配不成功，转向步骤（7）；

（7）退出程序。

4.1.3数据结构的设计

1、中文分词算法的数据结构如下：

@param int char // 参数的汉字说明，用于接收查询界面 get 函数传递的参数（即搜索的关键字）

int main (int argc, char\* argv[])

{

程序开始执行 （当前查询和结果表号） i-Query.GetInputs();

开始具体搜索程序 iQuery.SetStart();

调用 gettimeofday 函数开始计时获取程序运行时间差;

利用 iHzSeg.SegmentSentenceMM 函数将查询变量分词分成多个字段的形式;

将以 "/" 划分开的关键字一一顺序放入一个向量容器 iQuery.ParseQuery 中;

调用 iQuery.GetRelevantRst 检查参数是否结束;

调用 gettimeofday 结束搜索;

显示查询结果 CDisplayRst iDisplayRst;

return 0;

}

2、在对查询关键字做完分词处理之后我们需要定义一个结构体用来存储匹配的结果和过程值，结构体名 indextype。该记录包括两项：整型变量fmdegree和布尔型变量find ,其中fmdegree 用来存储 KMP 算法过程中的相似度，find 用来区别是否传统匹配成功，我们对find赋值TRUE或FALSE ，其中TRUE表示匹配成功，FALSE表示匹配失败。如下：

struct indextype

{

int fmdegree ；

Boolean find;

}m;

3、数据结构设计如下：

引入 indextype 结构体；

indexf (string s,string t, indextype m)

{

变量初始化；

获取主串和模式的字符长度并做记录；

while(分词长度小于数据库关键词字段)

{

利用匹配程序进行匹配；

} //end while，如果匹配到的字符个数大于长度较短字符串的个数则返回,否则返回假

} //end indexf 进行循环比较

4.1.4模糊搜索的实现

引入 indextype 结构体；

函数头 Fuzzysearch( 数据库关键字字段，分词，模糊相似度)

{

调用分词程序对用户关键字进行分词并计算字段长度；

判断是否为词典中有实体意义的最小单位的词，是则继续，否则返回分词；

if 调用indexf函数的返回值为真则说明首次查询成功，直接返回真，结束程序；

else 相似度为1，但是indexf 返回的find 为假则返回假

{

按照相似度循环,直到得到和理解过，结束程序；

调用回送程序，显示查询结果；

}

}//end

4.2 热度加权算法

单位时间内用户发布的帖子，会由机器计算热度数值，按照热度由高至低排序，热门帖子会被推送到app首页上。热门帖子的热度数值是根据该帖子评论数、发布时间等因素计算出热度基数，再考虑热度权重，得出最终热度值。

为了不断提升社区用户的阅读体验，确保一个更完善的内容消费生态，管理员可以适时对特定类型的内容做加权处理，当然对应的也会对帖子上有碍阅读体验类型的内容降权。热度权重规则如下：（d、e、f、g、h、j、k表示每一项代表的加权值）

1. 热度加权：

（1）所发帖子图片内容适中\*d；

（2）所发帖子内容丰富\*e；

（3）实名用户（绑定老人手机号、完善个人资料等）\*f。

2、热度降权：

（1）非原创内容并且附加外链、广告等\*g；

（2）同一时段下同一个用户最多有2条帖子上首页，并且第2条帖子热度要降权调整\*h；

（3）内容相似的帖子按时间排序，第二条帖子要降权\*j；

（4）文字过于冗长要降权\*k。

假设a为某篇帖子的具体发布时间，精确到秒；b为一个固定的时间常量，比如说可以是“牵伴”app正式上线的时间2017-5-20 00:00:00，则可以计算得到和间隔的秒数t。

再引入c表示某篇帖子的评论数，则可以计算得到变量Z：

再引入权重，则可以计算得到变量R：

1）时间点b是一个固定的值，所以t反映的是帖子的新鲜程度。  
2）45000 代表的是一个12.5小时周期内的总秒数。它与t一起使用，随着时间的推移，新帖子的得分会逐渐超越高评论数的老帖子，从而实现自动更新的效果。  
3）的使用可以使得早期的评论分量更大。

按照R的大小进行排序，选取前十个帖子推送到首页上。

5.分解描述

5.1通信模块

1、简介

该模块负责客户端与服务端之间的通信，客户端使用java的socket技术、服务器端使用mfc的csocket技术。

2、功能列表

本模块主要包括打开服务器端通信以及打开客户端通信。

5.1.1打开服务器端通信

1、功能设计描述

本功能用于创建服务器端的通信结构，等待和处理客户端发来的信息。

1. **类**

**1）C久违服务器Dlg**

该类存在于服务端，用于提供给管理员一个服务器界面并与管理员进行交互，管理员可以打开和关闭服务器，以及查看当前登录用户个数和账号信息和时间。

**2）CServerSocket**

该类存在于服务器，用于接受并处理客户端发来的信息。

1. **类与类之间的关系**
2. **文件列表**

表5.1.1 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| ServerSocket.h | C++ | 久违服务器/头文件/ ServerSocket.h | CServerSocket类的成员变量和函数的声明部分 |
| ServerSocket.cpp | C++ | 久违服务器/源文件/ServerSocket.cpp | CServerSocket类的成员函数定义部分 |
| 久违服务器Dig.h | C++ | 久违服务器/头文件/久违服务器Dig.h | C久违服务器Dig类的成员变量和函数的声明部分 |
| 久违服务器Dig.cpp | C++ | 久违服务器/头文件/久违服务器Dig.cpp | C久违服务器Dig类的成员函数的定义部分 |

2、功能实现说明

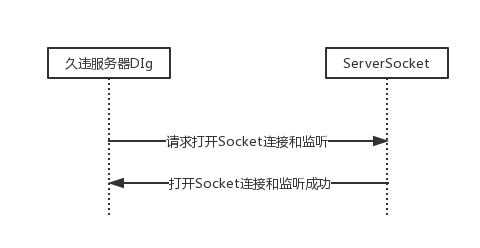


图5.1.1 功能实现说明图

5.1.2打开客户端服务器

1、功能设计描述

本功能用于打开客户端通信，请求与服务器端的链接或者向服务器发送信息以及接收服务器发回的信息并进行广播。

**（1）类**

**1）client**

该类继承于线程，用于打开收发信息的线程。

**2）getMsg**

该类继承于service类用于打开客户端client线程。

**3）inputThread**

该类继承于线程，用于接收服务器发来的信息。

**4）outputThread**

该类继承于线程，用于发送信息到服务器端

1. **类与类之间的关系**

**（3） 文件列表**

表5.1.2 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| client.java | Java |  | 客户端线程类，用于启动和收发信息线程。 |
| getMag.java | Java |  | 服务类，用于启动客户端线程以及广播信息 |
| inputThreads.java | Java |  | 接收信息线程类，用于接收服务器端发来的信息 |
| outputThreads.java | Java |  | 发送信息的线程类，用于发送信息到服务器端 |

2、功能实现说明

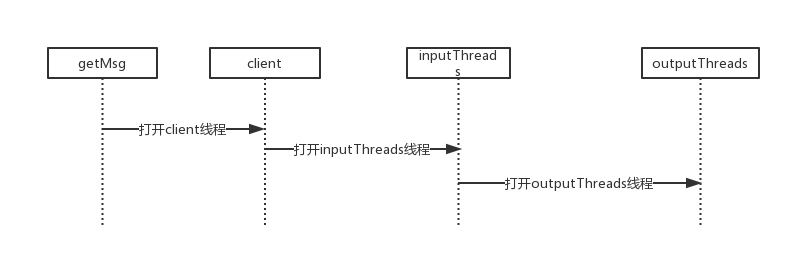


图5.1.2 功能实现说明图

5.2登录、注册、个人资料设置模块

**1、简介**

本模块实现了登录、注册和个人资料的不全和更改功能，数据上传存储到服务器端。

**2、功能列表**

本模块主要包括注册新的账号、登录自己账号、个人资料的补全和更改

5.2.1登录功能

1、功能设计描述

本功能将账号和密码信息发送到服务器端、等待服务器端的验证和回复，验证成功则登录成功，否则失败。

**（1）类**

**1）MainActivity**

该类提供给用户一个UI交互界面，负责将账号密码信息发送到服务器并等待服务器回复，验证成功则跳转到软件主界面，否则进行错误提示。

1. **类与类之间的关系**

**（3） 文件列表**

表5.2.1 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| MainActivity.java | Java |  | 提供给用户界面以及向服务器发送账号和密码等待回复，并进行不同的操作。 |

2、功能实现说明

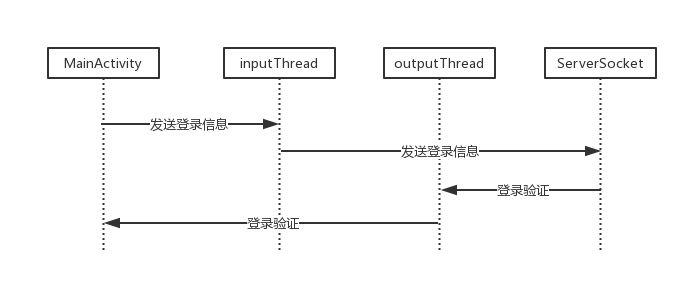


图5.2.1 功能实现说明图

5.2.2注册功能

1、功能设计描述

本模块提供UI界面并负责把注册信息发送到服务器，等待服务器验证和回复。

**（1）类**

**1）RegisterActivity**

该类提供给用户一个UI交互界面并负责发送注册信息到服务器，等待服务器的验证和回复。

1. **类与类之间的关系**

**（3） 文件列表**

表5.2.2 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| RegisterActivity.java | Java |  | 提供给用户界面并且发送注册信息到服务器端等待验证和回复，根据回复进行不同的操作。 |

2、功能实现说明

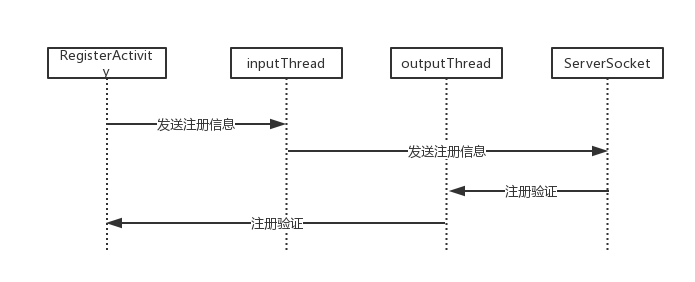


图5.2.2 功能实现说明图

5.2.3个人信息设置模块

1、功能设计描述

本模块提供一个个人信息的补全或者更改的功能。

**（1）类**

**1）info\_selfActivity**

该类提供给用户一个UI交互界面并且把信息发送到服务器等待服务器的验证和回复。

1. **类与类之间的关系**

**（3） 文件列表**

表5.2.3 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| Info\_Activity.java | Java |  | 提供给用户界面并且把信息发送到服务器端等待验证和回复。 |

2、功能实现说明

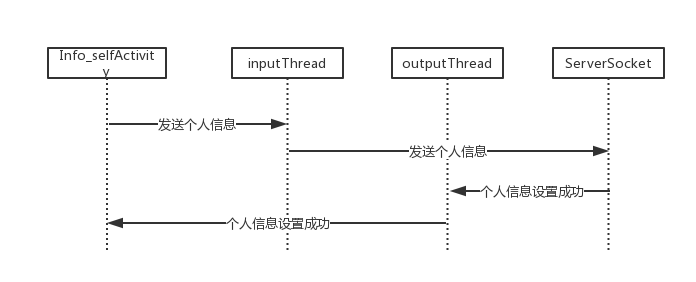


图5.2.3 功能实现说明图

5.3.语音聊天模块

**1、简介**

本模块实现了添加老人、查看老人资料和实时聊天的功能。

**2、功能列表**

功能包括添加老人、查看老人资料、实时聊天

5.3.1添加老人

1、功能设计描述

本模块提供一个软件添加用户的窗口，用户可以请求添加

**（1）类**

**1）addParentActivity**

该类提供给用户一个添加窗口，用户可以进行请求添加。

**2）child\_infoActivity**

该类提供一个老人信息界面，用户可以查看已绑定老人的信息。

1. **类与类之间的关系**

**（3） 文件列表**

表5.3.1 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| addParentActivity.java | Java |  | 该类提供给用户一个添加界面 |
| child\_infoActivity | Java |  | 该类提供给用户一个查看老人个人信息界面 |

2、功能实现说明

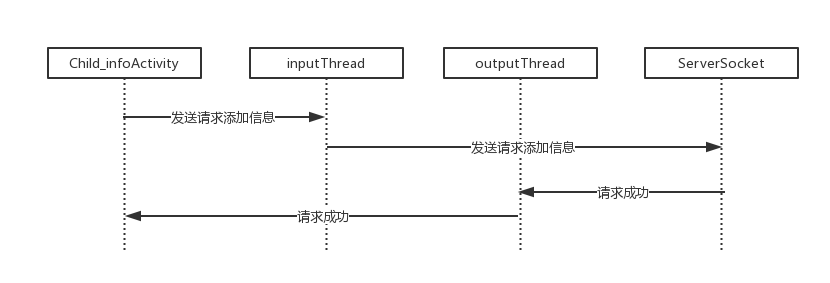


图5.3.1 功能实现说明图

5.3.2查看老人资料与删除

1、功能设计描述

本模块提供一个老人列表以及老人个人资料界面，在老人个人资料界面可以将老人删除。

**（1）类**

**1）ParentMainFragment**

该类提供给用户一个老人列表

**2）parent\_infoActivity**

该类提供一个老人信息界面，用户可以在该界面中进行删除。

1. **类与类之间的关系**

**（3） 文件列表**

表5.3.2 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| addParentActivity.java | Java |  | 该类提供给用户一个老人列表 |
| Parent\_infoActivity | Java |  | 该类提供给用户一个查看老人个人信息界面，并提供删除功能 |

2、功能实现说明

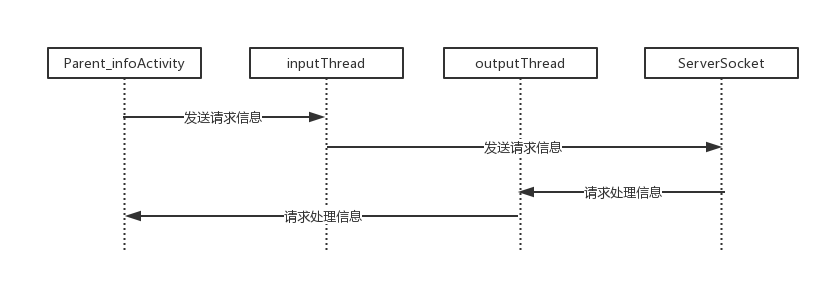


图5.3.2 功能实现说明图

5.3.3聊天

1、功能设计描述

本模块提供一个父母和子女双方实时聊天功能。

**（1）类**

**1）chatActivity**

该类提供给用户一个聊天界面并且可以发送信息给老人以及接受老人的信息。

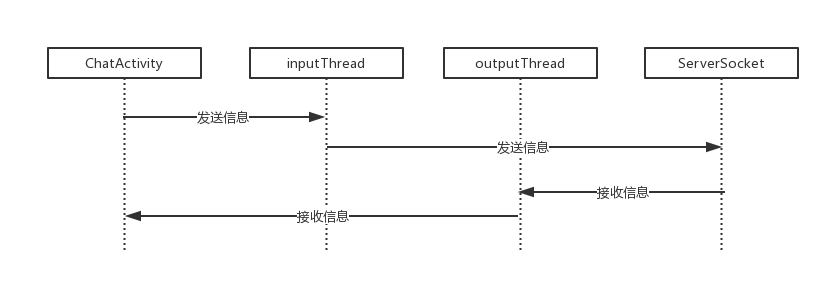
1. **类与类之间的关系**

**（3） 文件列表**

表5.3.3 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| chatActivity.java | Java |  | 该类提供给用户一个UI界面并且把信息发送到服务器端等待服务器的验证和回复。 |

2、功能实现说明

图5.3.3 功能实现说明图

5.4.论坛模块

1、简介

本模块实现了在论坛中发帖，看帖和评论功能。

2、功能列表

功能包括创建贴子，查看贴子信息，对活动进行评论。

5.4.1创建贴子

1、功能设计描述

本模块提供给用户一个贴子创建界面，用户可以创建新帖并上传到服务器中。

**（1）类**

**1）craetActivity**

该类提供给用户一个新帖发布界面，用户可以创建一个新帖。

1. **类与类之间的关系**

**（3） 文件列表**

表5.4.1 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| creatActivity.java | Java |  | 该类提供给用户一个新帖发布界面，用户可以创建一个新贴。 |

2、功能实现说明

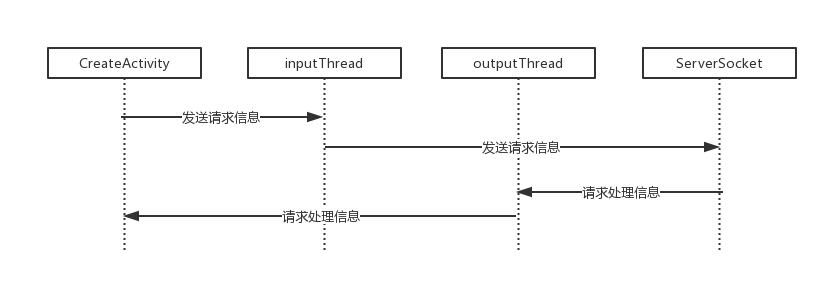


图5.4.1 功能实现说明图

5.4.2查看贴子

1、功能设计描述

本模块提供一个贴子查看界面，用户可以自行查看感兴趣的贴子。

**（1）类**

**1）watchActivity**

该类提供给用户一个查看贴子界面

1. **类与类之间的关系**

**（3） 文件列表**

表5.4.2 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| watchActivity.java | Java |  | 该类提供给用户一个查看界面 |

2、功能实现说明

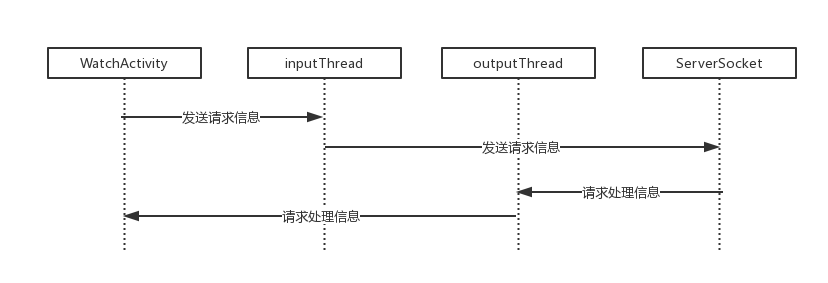


图5.4.2 功能实现说明图

5.4.3评论贴子

1、功能设计描述

本模块给用户提供评论窗口，用户可以对感兴趣的贴子进行评论。

**（1）类**

**1）CommentActivity**

该类提供给用户一个评论窗口，用户可以自行评论。

1. **类与类之间的关系**

**（3） 文件列表**

表5.4.3 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| CommentActivity.java | Java |  | 该类提供给用户一个评论界面 |

2、功能实现说明

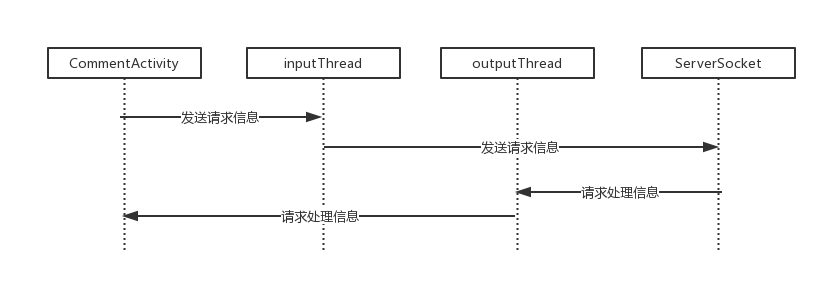


图5.4.3 功能实现说明图

5.5.消息接受与推送模块

5.5.1消息接受与提示

1、功能设计描述

本模块提供给用户一个新消息的提醒界面用于消息的实时接收并且存储到本地数据库的功能，当软件不在前台时进行消息推送。

**（1）类**

**1）newMessage**

该类提供给用户一个新消息的提醒界面

1. **类与类之间的关系**

**（3） 文件列表**

表5.5.1 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| newMessage.java | Java |  | 提供给用户一个新消息的提醒界面用于消息的实时接收并且存储到本地数据库的功能，当软件不在前台时进行消息推送 |

2、功能实现说明

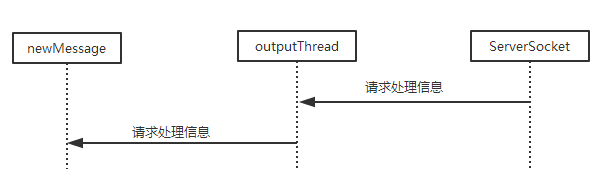


图5.5.1 功能实现说明图

5.5.2推文推送

1、功能设计描述

本模块提供给用户一个今日推文的推送入口，用户可以自行点击查看今日推文的内容。

**（1）类**

**1）Article**

该类提供给用户一个今日推文的推送入口

1. **类与类之间的关系**

**（3） 文件列表**

表5.5.2 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| Article.java | Java |  | 提供给用户一个今日推文的推送入口 |

2、功能实现说明

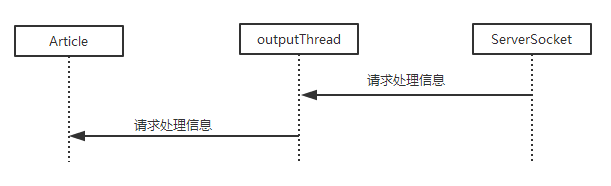


图5.5.2 功能实现说明图

5.6.地图模块

**1、简介**

本模块实现了子女对父母的历史移动轨迹与实时位置的监测。

**2、功能列表**

功能包括实时定位、历史轨迹。

5.6.1地图显示

**1、功能设计描述**

本模块提供给用户一个地图模块显示界面。

**（1）类**

**1）ViewLocation**

该类提供给用户一个地图模块显示界面，用户可以选择实时定位、历史轨迹功能。

1. **类与类之间的关系**
2. **文件列表**

表5.6.1

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| ViewLocation.java | Java |  | 该类提供给用户一个地图模块显示界面，用户可以选择实时定位、历史轨迹功能。 |

5.6.2实时定位与历史轨迹

**1、功能设计描述**

用户通过该模块可以实现对父母实时定位以及历史轨迹的监测。

**（1）类**

**1）LocationAction**

该类负责地理位置类的请求，请求分别为最新位置信息查询以及历史路线信息查询。

1. **类与类之间的关系**
2. **文件列表**

图5.6.2 类表

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 存放位置 | 说明 |
| LocationAction.java | Java |  | 该类负责地理位置类的请求，请求分别为最新位置信息查询以及历史路线信息查询。 |

**2、功能实现说明**

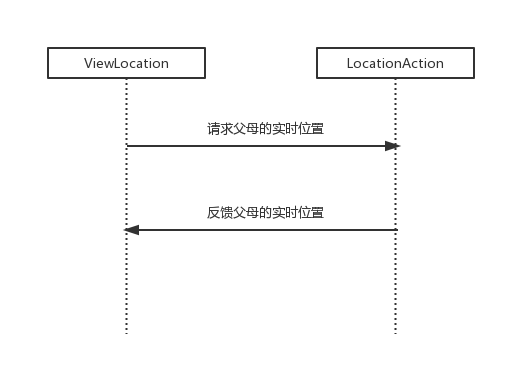


图5.6.2.1 实时定位功能的实现

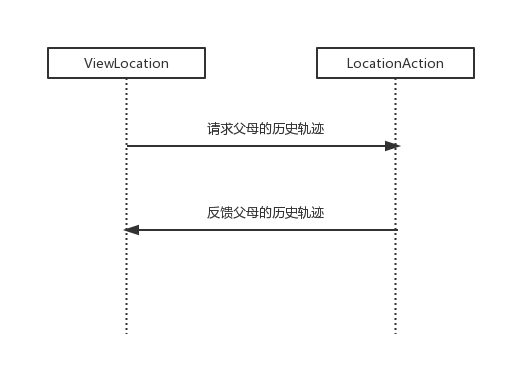


图5.6.2.2 历史轨迹功能的实现

6 接口描述

6.1百度地图接口

名称：百度地图API

说明：百度地图API是为开发者免费提供的一套基于百度地图服务的应用接口。

定义：此百度地图API为Android版，为Android设备地图应用开发提供基本地图、定位等服务。用户使用该套SDK，需要先申请key。

6.2闹钟接口

名称：AlarmManager

说明：在Android系统中，闹钟和唤醒功能都是由Alarm Manager Service控制并管理的。

定义：此API为Android系统的Framework层把底层细节做了封装并统一提供的API，为Android设备的应用开发提供设置一次性闹钟或重复闹钟等服务。

6.3相册接口

名称：UploadFileServlet

说明：在Android系统中， UploadFileServlet为图片上传接口（Servlet实现）支持参数和多张图片的上传。

定义：此API为Android系统应用程序层本身就集成的API，为Android设备的应用开发提供上传照片等文件的服务。

6.4 通讯接口

名称：SendCallClickListener

说明：在Android系统中，打电话功能都是由SendCallClickListener控制并管理的。

定义：此API为Android系统应用程序层本身就集成的API，为Android设备的应用开发提供打电话的服务。用户使用时应在AndroidManifest.xml中进行权限配置：<uses-permission android:name="android.permission.CALL\_PHONE" />

7 出错处理设计

7.1 错误异常处理机制和原则

由Try代码块保护的代码所发生的任何异常，甚至包括在不含Try代码块的被调函数或方法以内的异常都将被Catch代码块内的代码处理。当然，除非catch代码块自己也扔出了异常，而在这种情况下异常会被扔到下一个级别更高的try代码块，哪怕那意味着从当前正在执行的函数中扔出异常。

异常的4条重要规则：

* 如果catch块有若干个，那么异常将根据其类型被扔给最适当的一个catch接受处理。
* 如果没有找到可接受的catch块，则异常被从当前的try块扔到从调用顺序链中找到的下一个可用的catch块。
* 异常对象的类型给出了发生错误本质的重要信息。
* 异常可以通过throw关键词显式扔出。

代码就可以选择性地只处理那些有能力处理的错误，同时相信其他问题都会统统交给调用堆栈，哪怕至少作为通知处理。在实际应用中，因为让CLR观察扔出的异常对性能有一定的影响，所以使用单个try块同时对应多个特殊异常的catch语句是检查代码多个特定错误的最佳方式。

根据系统应用，常见错误异常分为数据存储部分，应用部分，核心库部分，商业层部分。

数据存储部分：主要系统在与数据库产品交互时，常见的一些错误。比如数据库连接错误，数据库对象不存在或数据字符过多。

应用部分：主要是反映用户在键盘输入操作时，可能引起的数据类型错误，字符长度超过限制等，使用鼠标或键盘可能引起的操作错误等。

核心库部分：主要反映系统框架中的一些错误。比如数组下标越界，数字超出范围等。

商业层部分：主要反映系统中的一些如权限被拒绝,输入参数错误等。

7.2 出错信息

在软件系统的开发过程中，需要对软件中存在的漏洞立即进行处理，即使一个很小的漏洞，也会造成整个软件开发系统的垃圾资源无限增多，最终可能会导致系统的瘫痪。因此在系统设计中，对拟开发系统要注意数据的约束性。为了进行数据统一收集和规范数据统一格式方便管理者的处理，针对安全隐患，本项目组进行了统一约束规范。同时，项目组也应该解决服务器端的信息安全问题，这样才能更好的运用管理系统，从而减少系统的维修次数，确保系统的安全使用。

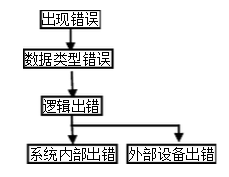


图7.2 出错图

对于客户端不符合系统设计预期的错误输入，应该在客户端给以提示，以避免错误的请求数据对后端产生影响。而对于服务端可能产生的异常或错误，系统应该以有好的方式告知用户，不应直接将系统抛出的异常信息等呈现给用户。

具体如下表所示：

表7.2 错误表

|  |  |  |
| --- | --- | --- |
| 错误层次 | 错误类型 | 解决方案 |
| 客户端错误 | 用户输入与数据类型不相符的数据 | 客户端以有好的方式提示用户对输入数据进行检查 |
| 无网络连接 | 提示用户恢复网络连接 |
| 重复注册或空登陆 | 提示用户用户名不存在或者用户已注册 |
| 用户名和密码出错 | 提示输入不正确，检查信息是否正确，并返回 |
| 验证码输入错误 | 登陆失败，应重新输入 |
| 密码与确认密码不同 | 提示密码与确认密码不同 |
| 提交信息失败 | 提示失败信息，并建议检查错误信息类型 |
| 联系方式具有的格式输入错误 | 提示不是其所拥有的格式，重新输入 |
| 标题为空，或标题大于规定字数 | 提示标题不能为空且应该小于X字，应重新输入 |
| 服务端错误 | 数据库连接失败 | 及时恢复数据库连接 |
| 服务器访问压力过大 | 采取服务器负载均衡技术，同时调整服务器性能 |
| 程序运行中抛出的异常信息 | 将请求数据和错误异常记录到Log日志，同时告知客户端 |
| 不合法的URL请求 | 提示错误，返回登录界面，Struts的请求错误控制 |
| 数据访问量过大 | 返回请求失败信息，针对需求重新选择服务器 |

7.3 补救措施

所有的系统，不管采用哪种开发方式，都不可避免的存在弱点。所以不但在信息管理系统这一级而且在计算中心这一级(如果适用，也包括远程设备)都要审核并提出设计到安全性的问题。通过识别系统的弱点来减少侵犯安全性的危险，以及采取必要的预防措施来提高安全水平，这是用户和信息服务管理部门可做得到的。

（1）后备技术

由于现代计算机科学技术的快速发展，以往的数据备份已经不能满足用户和管理者对现有系统的要求，现在的技术一般是进行系统数据库的实时备份和备份服务器。当数据库瘫痪时，通过快速进行系统库的切换，这样可以有效的减少数据的丢失。当用户量大时，也可以使用备用服务器，两个服务器进行同步数据存储，如果一台服务器出错时，可以转换到另一台进行相同的操作，让用户不需要因瘫痪而面临网页打不开、数据丢失等相关问题。

（2）降效技术

通常情况下系统的管理员都不是专业的计算机人员，对计算机专业的知识了解相对较少。在面对系统瘫痪等错误时，都不知道如何去解决问题。如今，随着计算机科学技术的飞速发展，为了让系统用户能够自行解决运行过程中遇到的问题，大多采取小型服务器作为后备技术。这项技术在现有的开发系统中使用很广泛，当然备用的服务器从价格上考虑，可能不如主服务器，也可能会降低工作效率，但是从数据的完整性和安全出发，可以让维修人员有更多时间进行系统的维护。

（3）恢复及再启动技术

由于采取的是低端服务器，同时进行数据库的实时备份。这样在系统维护好完成后，可以很快的进行数据同步，同时也可以很快的进行系统的运行。

7.4 系统维护设计

信息管理系统不同于其他系统，它并不是一劳永逸的最终系统。在它的运行过程中，还有大量运行管理、系统维护、系统设置等工作要做。交付使用的信息管理系统跟工业产品不同，工业产品可以先生产一个样品，经过试验、改进再正式投入批量生产，而信息管理系统它需要在用户使用过程中不断完善。系统维护包括程序的维护、数据文件的维护、代码的维护等，而且系统的修改，往往会造成系统程序、文件、代码的修改。因此，系统的修改必须通过批准。