
运动社交web应用

系统设计文档

刘璇琳 - 2016年11月1日



1.引言

1.1编制目的

本报告详细完成对运动社交网络应用的概要设计，达到指导详细设计和开发的目的，同时实现和测试人员及用户的沟通。

本报告面向开房人员、测试人员及最终用户而编写，是了解系统的导航。

1.2词汇表

词汇名称	词汇含义
KeepFit	健康管理网络应用
Root	提供服务的url的根

2.产品概述

参考运动社交网络应用需求规格说明中对产品的概括描述。

3.逻辑视角

KeepFit应用中，选择了分层体系结构风格，将系统分为3层（展示层，信息逻辑层，数据层）示意整个高层抽象。展示层包含网页UI的实现，信息逻辑层包含信息逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视角和逻辑设计方案如图1和图2所示。

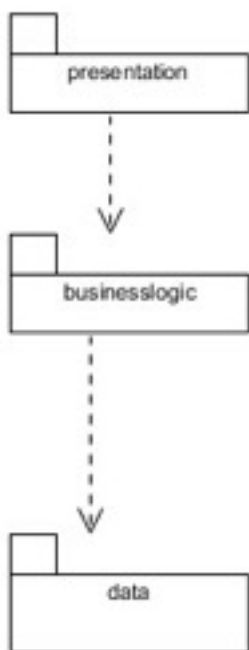


图1 参照体系结构风格的包图表达逻辑视角

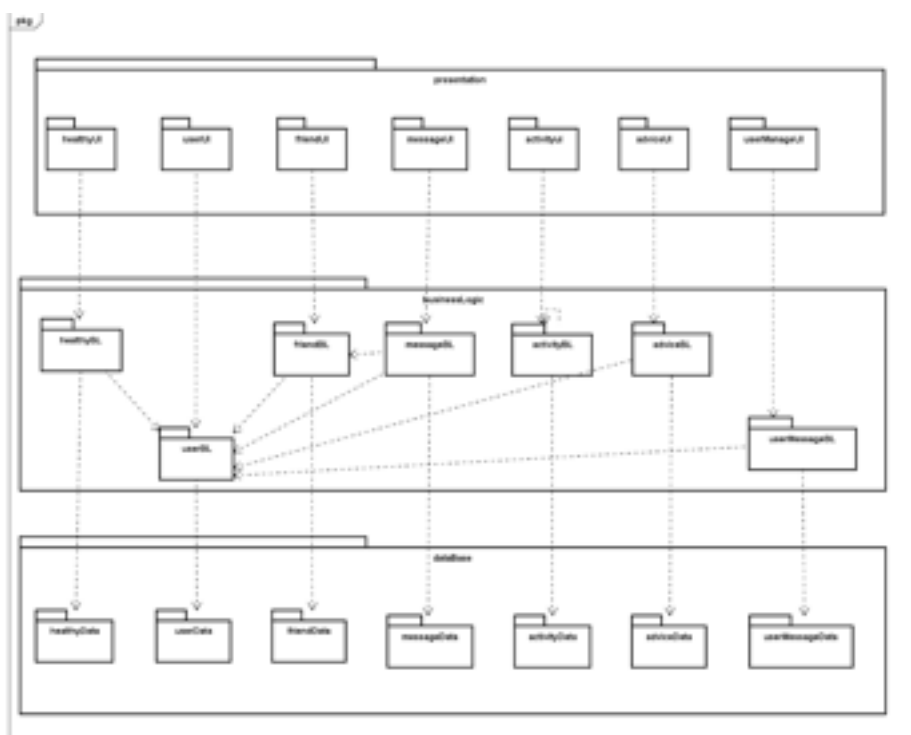


图2 软件体系结构逻辑设计方案

4.组合视角

4.1运行时进程

在KeepFit应用中，会有多个客户端进程和一个服务器端进程，其进程图如图3所示。

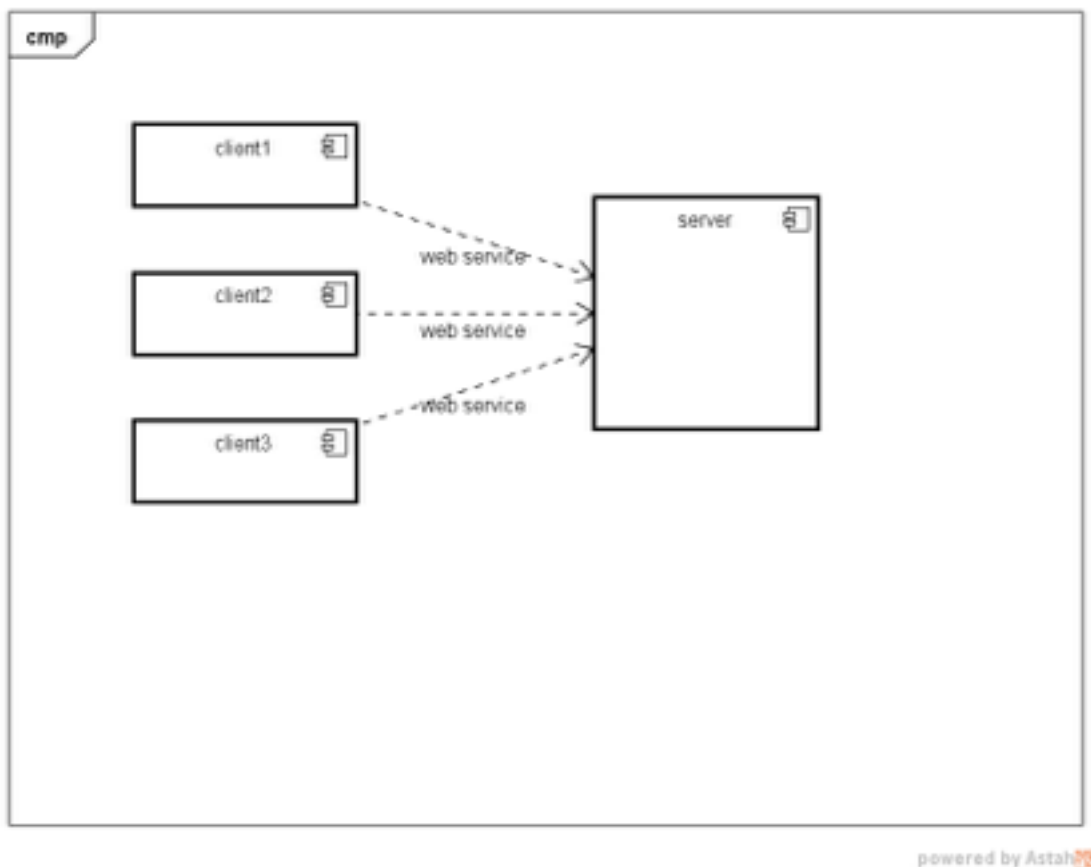


图3

结合部署图，客户端进程是在客户端机器上运行，服务器端进程在服务器端机器上运行。

4.2物理部署

KeepFit应用中客户端构件为浏览器，置于客户端机器上，服务器端构件是放在服务器端机器上。在客户端节点上，只需安装浏览器，连上网络，即可使用该应用，无需其他部署。

5.接口视角

5.1模块的职责

客户端模块和服务端模块视图分别如下所示。客户端各层和服务端各层的职责分别如表2和表3所示。

表2 客户端各层职责

层	职责
启动模块	负责通过网络连接服务器，取得初始数据，加载用户界面
用户界面层	基于浏览器窗口的KeepFit应用用户界面
业务逻辑层	对于用户界面的输入进行相应并进行业务处理逻辑
客户端网络模块	利用网络与服务端进行交互

表3 服务器端各层职责

层	职责
启动模块	负责初始化网络通信机制
数据层	负责数据的持久化及数据访问接口
服务器端网络模块	利用网络与客户端进行交互

5.2 用户界面层的分解

根据需求，应用存在11个相互交互的用户界面：登录界面、普通用户主界面、专业用户主界面、账户管理界面、活动管理界面、通讯主界面、好友管理界面、个人健康界面、建议管理界面、用户管理界面
界面跳转如图4所示。

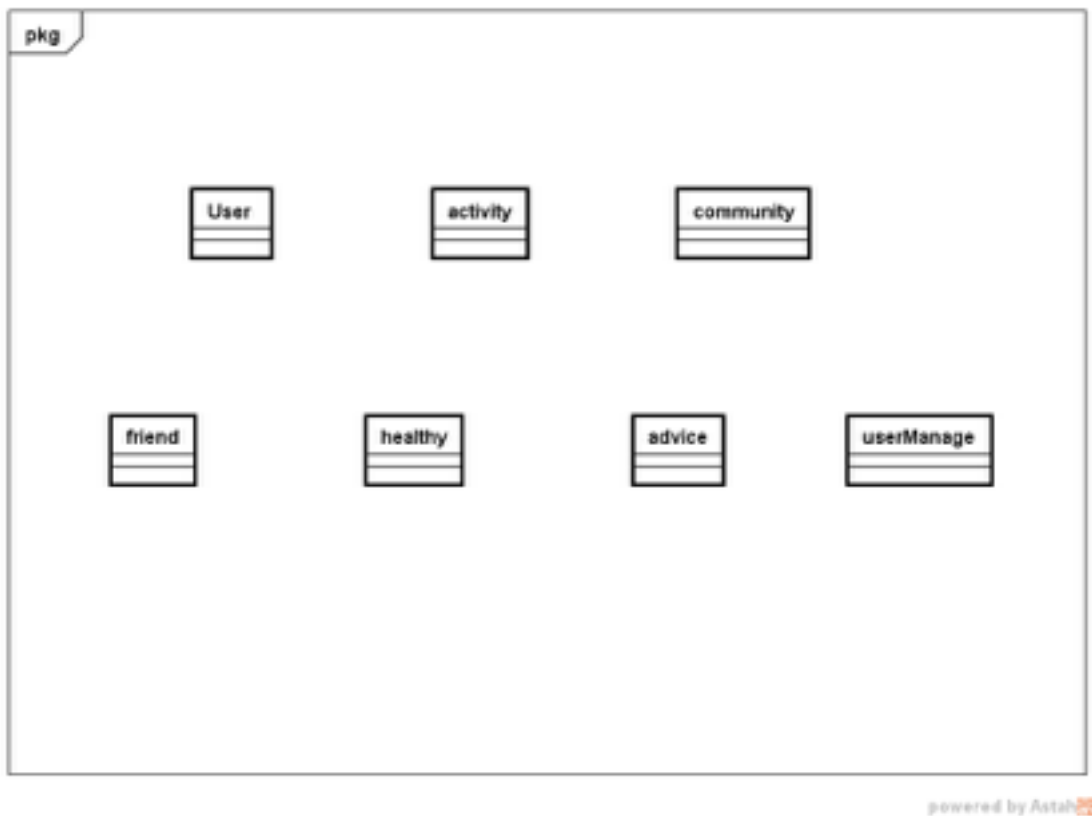


图4 用户界面层的分解

5.3业务逻辑层的分解

业务逻辑层包括多个针对界面的业务逻辑处理对象。例如，User对象负责处理登录界面和账户管理的业务逻辑;activity对象负责活动管理界面的业务逻辑。业务逻辑层的设计如图5所示。

图5 业务逻辑层的设计



5.3.1 业务逻辑层模块的职责

业务逻辑层的职责如表4所示。

模 块	职 责
userbl	负责实现对应与登录界面和账户管理所需要的服务。
Activitybl	负责实现与活动管理有关界面所需要的服务。
communitybl	负责实现与通讯有关界面所需要的服务。
friendbl	负责实现与好友管理有关界面所需要的服务
healthybl	负责实现与个人健康有关界面所需要的服务
advicebl	负责实现与建议管理有关界面所需要的服务
userManger	负责实现与用户管理有关界面所需要的服务

表4 业务逻辑层模块的职责

5.3.2 业务逻辑层模块的接口规范

表5 userbl模块的接口规范

提供的服务（供接口）		
User.loginUser	语法	function login(\$userName, \$password)
	前置条件	用户正确输入密码和用户名
	后置条件	系统进入用户界面
User.buildUser	语法	function buildUser()
	前置条件	启动了用户自我管理回合
	后置条件	系统显示应该输入的信息
User .updUser	语法	function updUser()
	前置条件	启动了用户自我管理回合
	后置条件	更新用户名、用户密码等信息
User .getInfo	语法	function getInfo()
	前置条件	用户申请新账户或修改用户信息
	后置条件	从外部系统取得数据并传输给调用程序
User.getinit	语法	function getinit()
	前置条件	新用户已经输入必备信息
	后置条件	系统自动生成用户等级、身份等信息

需要的服务（需接口）	
服务名	服务
URL=ROOT+/User	提供与用户相关的数据服务

表6 activitybl模块的接口规范

Activity.findActivity	语法	function findActivity (\$id)
	前置条件	启动一个活动管理回合
	后置条件	显示某一活动具体信息
Activity.showActivity	语法	function showActivity()
	前置条件	进入活动界面
	后置条件	显示所有活动信息
Activity.addActivity.	语法	function addActivity()
	前置条件	启动一个活动管理回合
	后置条件	在数据库中新增一个活动的信息
Activity.delActivity	语法	function delActivity(\$id)
	前置条件	启动一个活动管理回合
	后置条件	删除数据库中的某个活动信息
Activity.updActivity	语法	function updActivity(int id)
	前置条件	启动一个活动管理回合
	后置条件	修改数据库中修改一个活动的信息
Activity.showHotActivity	语法	function showHotActivity()
	前置条件	进入活动界面
	后置条件	从数据库中得到热门活动的信息
Activity.getinit	语法	function getinit ()
	前置条件	已经输入所新增的活动的部分信息
	后置条件	初始化活动的创建时间、创建人等
需要的服务（需接口）		
服务名	服务	
URL=ROOT+/User	提供与用户相关的数据服务	
URL=ROOT+/Friend	提供与活动相关的数据服务	

表7 communitybl模块的接口规范

Community.showMessage	语法	function showMessage ()
	前置条件	进入消息查看界面
	后置条件	显示所有未读信息
Community.sendMessage.	语法	function sendMessage ()
	前置条件	启动一个通讯回合
	后置条件	在数据库中新增一条消息的信息
Community.delCommunity	语法	function delCommunity ()
	前置条件	启动一个通讯管理回合
	后置条件	删除数据库中的某个通讯信息
Community.getinit	语法	function getinit ()
	前置条件	已经输入所要发送的消息信息
	后置条件	初始化消息的发送时间、发送人等
需要的服务（需接口）		
服务名	服务	
URL=ROOT+/User	提供与用户相关的数据服务	
URL=ROOT+/Community	提供与通讯相关的数据服务	

表8 friendbl模块的接口规范

Friend.showFriend	语法	function showFriend ()
	前置条件	进入好友界面
	后置条件	显示所有好友信息
Friend.addFriend.	语法	function addFriend ()
	前置条件	启动添加好友回合
	后置条件	在数据库中新增一条好友申请的信息
Friend.applyManage	语法	function applyManage ()
	前置条件	启动好友申请管理回合
	后置条件	在数据库中新增一条好友申请结果的信息

Friend.updFriend	语法	function updFriend ()
	前置条件	启动好友管理回合
	后置条件	在数据库中更新一条好友信息
Friend.delFriend	语法	function delFriend (\$id)
	前置条件	启动一个好友管理回合
	后置条件	删除数据库中的某个好友信息
需要的服务（需接口）		
服务名	服务	
URL=ROOT+/User	提供与用户相关的数据服务	
URL=ROOT+/Friend	提供与好友相关的数据服务	

表9 healthybl模块的接口规范

Healthy.showHealthyData	语法	function showHealthyData ()
	前置条件	进入个人健康界面
	后置条件	显示个人健康信息及分析
Healthy.updHealthy	语法	function updHealthy ()
	前置条件	进入健康管理界面
	后置条件	更新个人健康界面显示的条目
需要的服务（需接口）		
服务名	服务	
URL=ROOT+/User	提供与用户相关的数据服务	
URL=ROOT+/Healthy	提供与个人健康相关的数据服务	

表10 advicebl模块的接口规范

Advice.showAdvice	语法	function showAdvice ()
	前置条件	进入建议界面
	后置条件	显示所有推送的建议信息
Advice.addAdvice	语法	function addAdvice ()
	前置条件	启动一个建议管理回合
	后置条件	在数据库中新增一条建议的信息

Advice.delAdvice	语法	function delAdvice (\$id)
	前置条件	启动一个建议管理回合
	后置条件	删除数据库中的某条建议信息
Advice.showHotAdvice	语法	function showHotAdvice ()
	前置条件	进入建议界面
	后置条件	从数据库中得到热门推送建议的信息
Advice.getinit	语法	function getinit ()
	前置条件	已经输入所新增的建议的部分信息
	后置条件	初始化活动的创建时间、创建人等
需要的服务（需接口）		
服务名	服务	
URL=ROOT+/User	提供与用户相关的数据服务	
URL=ROOT+/Advice	提供与活动相关的数据服务	

表11 userManagebl模块的接口规范

userManage.forbiddenUser	语法	function forbiddenUser (\$id,\$time)
	前置条件	进入用户界面
	后置条件	更新数据库中用户信息
userManage.delUser	语法	function delUser (\$id)
	前置条件	进入用户界面
	后置条件	删除数据库中的一条用户信息
需要的服务（需接口）		
服务名	服务	
URL=ROOT+/User	提供与用户相关的数据服务	

5.4数据层的分解

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、查、改。本系统数据采用 数据库的格式存储。

5.4.1数据层模块的职责

数据层模块的职责如表24所示。

表24 数据层模块的职责

模 块	职 责
URL=ROOT+/User	基于xml和json和SQLite文件的持久化数据文件的接口，提供用户数据集体载入、集体保存、增删改查服务
URL=ROOT+/Advice	基于xml和json和SQLite文件的持久化数据文件的接口，提供建议信息集体载入、集体保存、增删改查服务
URL=ROOT+/Healthy	基于xml和json和SQLite文件的持久化数据文件的接口，提供个人健康数据集体载入、集体保存、增删改查服务
URL=ROOT+/Friend	基于xml和json和SQLite文件的持久化数据文件的接口，提供好友信息集体载入、集体保存、增删改查服务
URL=ROOT+/Activity	基于xml和json和SQLite文件的持久化数据文件的接口，提供活动数据集体载入、集体保存、增删改查服务

6.信息视角

6.1数据持久化对象

数据存储于数据库中，部分数据类型暂定如下

·User_info包含用户的用户名、密码属性和身份等属性：

```
<user_info>
    <user_id>number</user_id>
    <user_name>String</user_name>
    <password>String</password>
    <identify>number</identify>
</user_info>
```

·User_detil包含用户的基本信息，如头像，身高，好友数等

```
<user_detil>
    <user_id>number</user_id>    <!--用户ID -->
    <user_name>string</user_name>    <!--用户名 -->
    <get_icon>string</get_icon>    <!--头像 -->
```

```

    <friends>number</friends>      <!--好友数 -->
    <weight>number</weight>        <!--体重 -->
    <height>number</height>        <!--身高 -->
    <realname>string</realname>    <!--真实姓名 -->
    <birthday>
        <year>number</year>
        <month>number</month>
        <day>number</day>
    </birthday>
    <address>string</address>      <!--地址 -->
    <gender>number</gender>        <!--性别 -->
    <hobby>string</hobby>          <!--爱好 -->
    <descroption>string</descroption> <!--描述-->
</user_detil>
·Body_info包含用户的身体数据，包括血压、心率、体脂率、体重等:
<body_info>
    <date>
        <year>number</year>
        <month>number</month>
        <day>number</day>
        <time>string</time>
    </date>
    <blood_pressure>number</blood_pressure>
    <heart_rate>number</heart_rate>
    <weight>number</weight>
    <body_fat_rate>number</body_fat_rate>
    <weight_taeget>number</weight_taeget>
</body_info>
·spoot_summary包含用户按天汇总的运动数据
<spoot_summary>
    <status>string</status> <!--运动状态，null不动 low轻微 proper适量 over
过于-->
    <calories>number</calories> <!--消耗的卡路里-->
    <meters>number</meters>      <!--运动距离-->
    <activity>number</activity> <!--运动量（卡路里/体重）-->
    <steps>number</steps>        <!--步数-->
    <minutes>number</minutes>    <!--运动时长-->
</spoot_summary>
·Activity_info包括活动信息
<Activity_info>
    <start_date>date</start_date>
    <end_date>date</end_date>

```

```
<user_id>number</user_id>
<content>string</content>
<theme>string</theme>
<picture>string</picture>
</Activity_info>
·advice中包含专业用户面对广泛用户提出的无差别建议
<advice>
    <time>date</time>
    <user_id>number</user_id>
    <content>string</content>
    <theme>string</theme>
</advice>
·friends中包含用户的好友及分组信息
<friends>
    <user_id>number</user_id>
    <friend_id>number</friend_id>
    <group_name>string</group_name>
</friends>
·message中包含用户交流信息
<message>
    <time>date</time>
    <from_id>number</from_id>
    <to_id>number</to_id>
    <content>string</content>
</message>
```

6.2 数据库表

数据库中包含User表、Activity表、Advice表、UserHealthy表、friend表、message表等。

7 性能优化

7.1 尽量减少HTTP请求

相关的多个CSS文件合并，相关的多个JS文件合并

7.2 使用浏览器缓存

尽量强制浏览器缓存文件，并设置过期时间

7.3 图片、JS文件等预载入

7.4 合理安排脚本文件、样式文件的位置

1、脚本文件置于底部

2、样式文件置于顶部

7.5 JS、CSS文件均使用外联方式

7.6精简各类文件

8 原型设计

个人活动主页登录界面



普通用户个人主页

主页活动消息登出



路人甲

运动追踪

我的好友

今日推送

我的设置

体重

身高

体重

保存

目标体重

更改

记录日期

体重

身高

血压

收缩压(高压)

舒张压(低压)

保存

专业用户个人主页

主页活动消息登出



路人甲

我的文章

我的健康

我的好友

我的设置

写一篇新文章

标题:

详情:

创建

清空

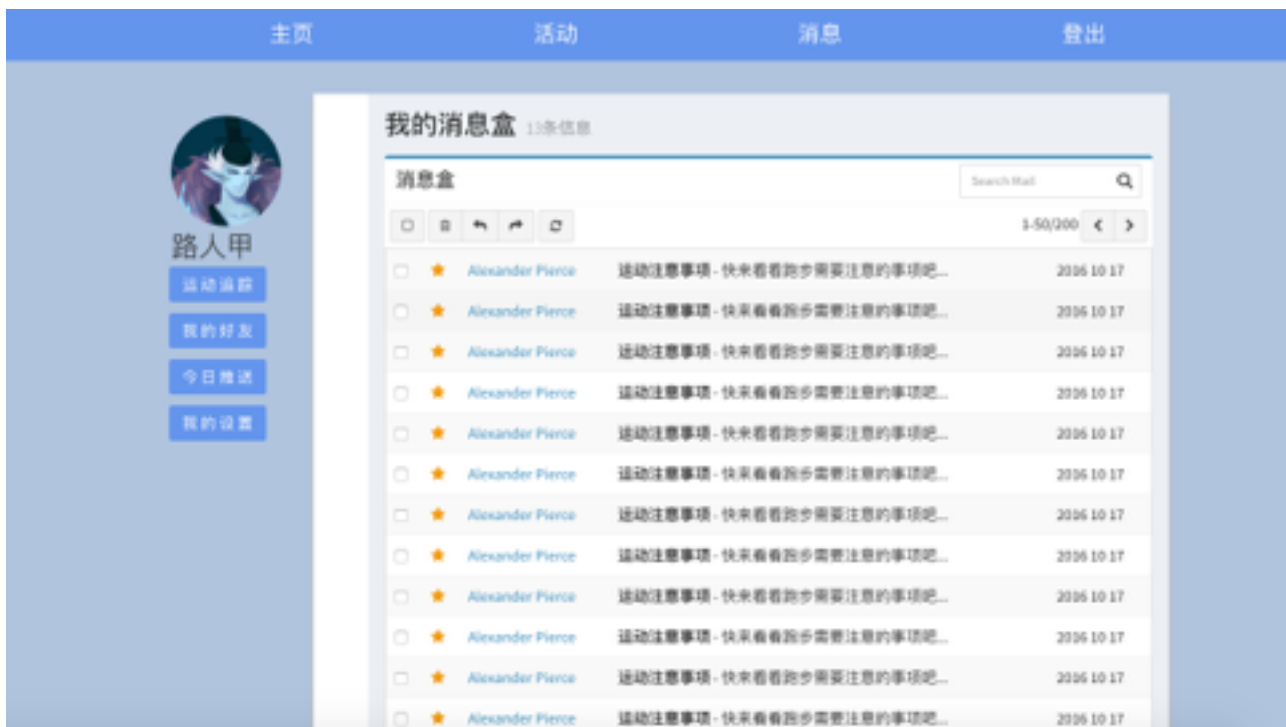
运动社交WEB应用－系统设计文档

16

管理员个人主页



邮箱界面




读邮件界面



写邮件界面



登陆界面



The image shows a login interface for an application named 'Keeper'. The interface is centered on a light blue background. It consists of a blue rounded rectangle containing the following elements:

- The word 'Keeper' in white text at the top left.
- A white input field with the placeholder text '请输入用户名' (Please enter username).
- A white input field with the placeholder text '密码: 至少6位' (Password: at least 6 characters).
- A link '忘记密码' (Forgot password) and a link '注册' (Register) in white text below the password field.
- A white button with the text '登录' (Login) in black text at the bottom.