

2016-11-5

SPORTS

详细设计文档

程翔 141250018

南京大学软件学院

更新历史

时间	修改人	备注
2016-10-24	程翔	创建文档
2016-10-25	程翔	完善文档

目录

更新历史.....	1
1. 引言.....	3
1.1. 编制目的.....	3
1.2. 参考资料.....	3
2. 产品描述.....	3
3. 分层体系结构.....	3
4. 导航设计.....	4
4.1. 导航内容.....	4
4.2. UI 原型界面	4
5. 人机交互设计.....	7
5.1. 人机交互界面.....	7
5.2. 人机交互界面转换.....	8
6. 接口设计.....	8
6.1. 业务逻辑层的分解.....	8
6.1.1. 业务逻辑层模块职责	8
6.1.2. 业务逻辑层模块的接口规范.....	9
6.2. 数据层的分解.....	15
6.2.1. 数据层模块的职责.....	15
6.2.2. 数据层模块的接口规范.....	15
7. 架构与组件设计.....	18
7.1. 系统架构.....	18
7.2. 组件设计.....	19
7.2.1. 图标（采用 font awesome 图标系统）	19
7.2.2. 动态.....	19
7.2.3. 按钮.....	19
7.2.4. 侧边栏.....	20
7.2.5. 图表.....	20
7.2.6. 竞赛.....	20
7.2.7. 分页.....	21
7.2.8. 搜索框.....	21

1.引言

1.1. 编制目的

本文档详细完成对 SPORTS 的概要设计，达到指导开发的目的，同时实现测试人员及用户的沟通。

本文档面向开发人员，测试人员及最终用户编写，是了解系统的导航。

1.2. 参考资料

- 《Web 程序设计 第 8 版》
- 《SPORTS 需求文档》

2.产品描述

参考 SPORTS 需求文档对产品的概述描述

3.分层体系结构

SPORTS 中，选择了分层体系结构的风格，将系统分为三层（展示层、业务逻辑层、数据层，参见 5 中人机交互的界面，参见 6 中接口设计的业务逻辑层和数据层的分解）能够很好的示意整个高层抽象。展示包括 GUI 界面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的读取。分层体系结构的逻辑视角如图 1 所示。



图 1

4. 导航设计

4.1. 导航内容

导航栏	侧边栏
首页	个人运动、竞赛信息、统计数据、我的健康
动态	无
竞赛	全部竞赛、发布竞赛
分析&建议	数据分析、运动推荐
组织	组织信息
个人	个人基本信息、个人运动信息


4.2. UI 原型界面

UI 界面原型图如下图 2-图 6 所示



图 2

The image displays a data visualization dashboard with three panels. The top panel is a bubble chart titled '运动气泡图' (Sports Bubble Chart) showing the relationship between '运动' (Sports) on the y-axis and '年龄' (Age) on the x-axis. Bubbles are colored by year, with red representing 1990 and blue representing 2015. The middle panel is a bar chart titled '运动柱状图' (Sports Bar Chart) showing the same relationship, with bars colored by year (red for 1990, blue for 2015). The bottom panel is another bubble chart titled '运动气泡图' (Sports Bubble Chart), identical to the top panel. The dashboard includes a sidebar on the left with navigation links: '数据' (Data), '分析' (Analysis), '可视化' (Visualization), '仪表盘' (Dashboard), and '设置' (Settings). The top right corner shows a user profile icon and the text 'A 1000000'.



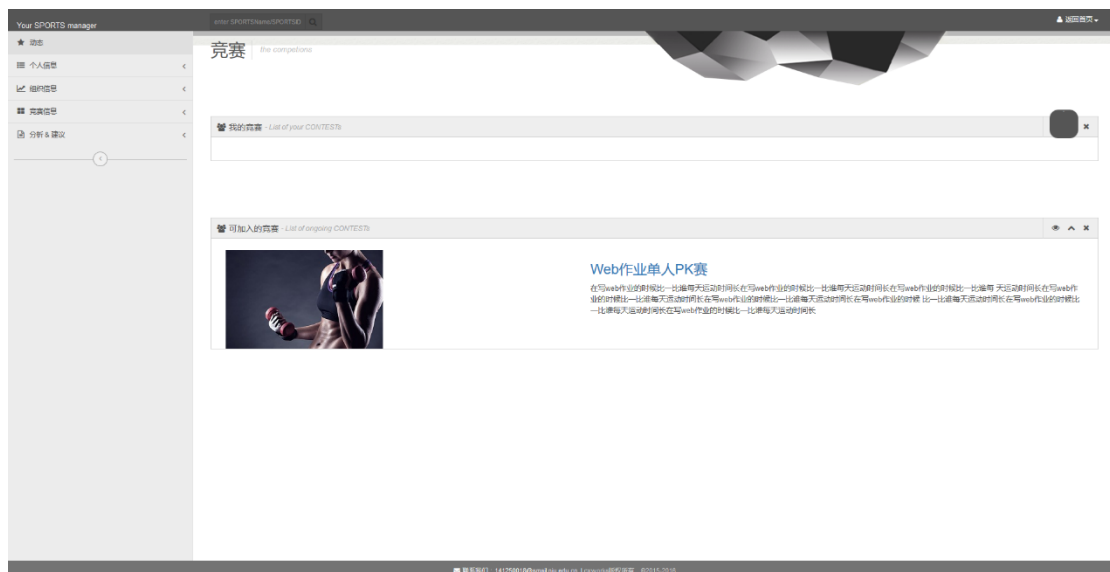


图 5

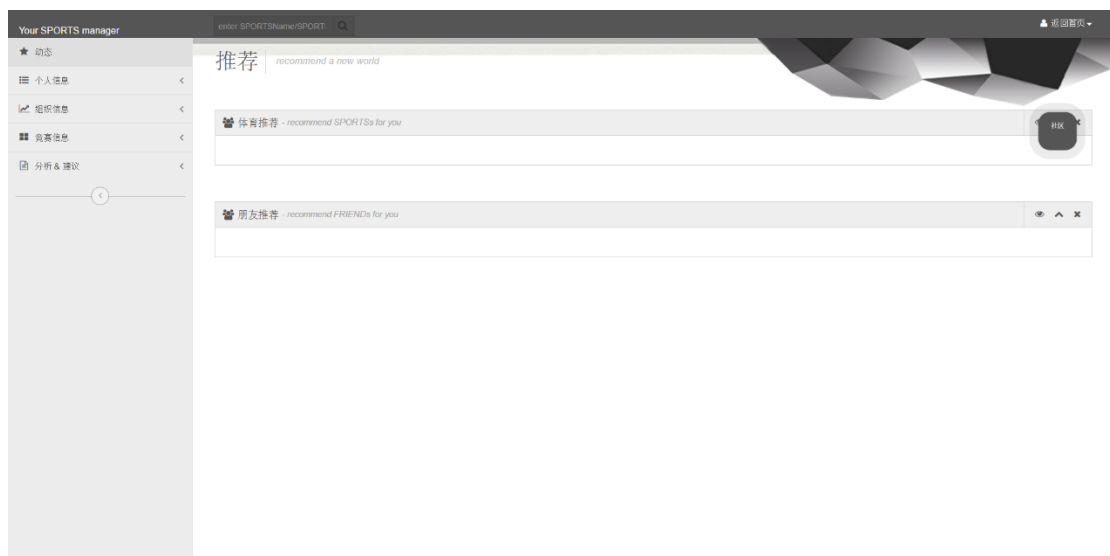


图 6

5.人机交互设计

5.1. 人机交互界面

存在 16 个界面，分别是登陆、注册界面、首页、竞赛（全部竞赛、我参与的、发布竞赛）、动态（好友动态、发布动态、树洞）、个人（基本信息、运动信息）、分析&推荐（数据分析、成长等级、运动推荐）

5.2. 人机交互界面转换

人机交互界面转换如下图：

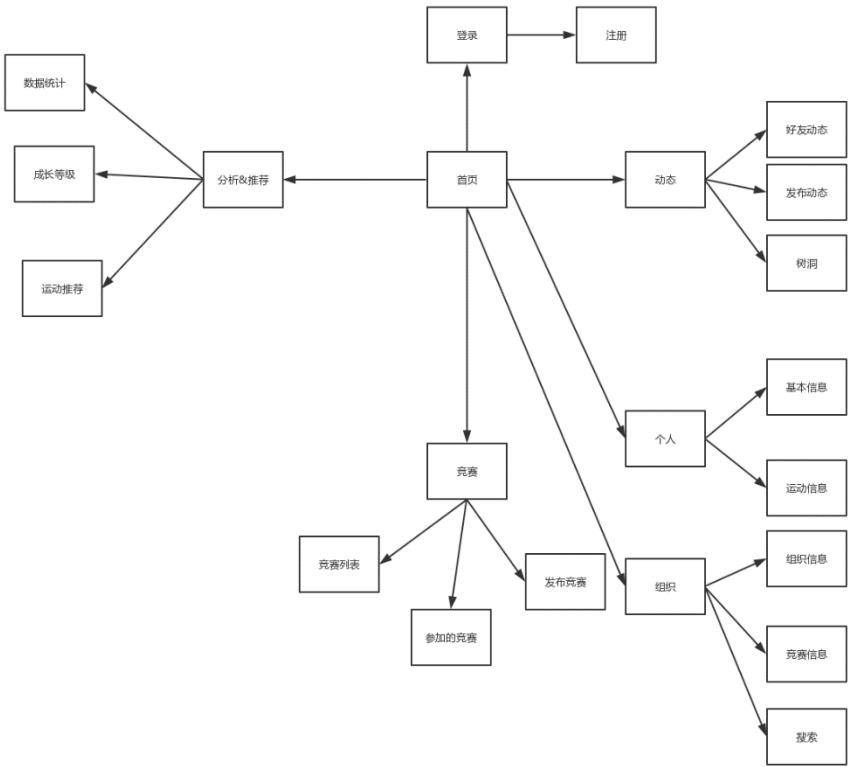


图 7

6.接口设计

6.1. 业务逻辑层的分解

6.1.1. 业务逻辑层模块职责

模块	职责
index	负责首页界面所需要的服务
contest	负责竞赛界面所需要的服务
news	负责动态界面所需要的服务
user	负责个人信息所需要的服务

dataanalyze

负责分析&推荐界面所需要的服务

6.1.2. 业务逻辑层模块的接口规范

6.1.2.1. user 模块

➤ 模块概述

user 模块负责用户的登陆注册，设置个人资料，读取用户账户好友动态，对账户动态好友进行更改。具体功能需求和非功能需求课参见 **SPORTS** 需求文档。

➤ user 接口规范

提供的服务（供接口）		
user.login	语法	function login(\$username, \$password)
	前置条件	username 和 password 不为空
	后置条件	验证用户名和密码是否匹配
user.register	语法	function register(\$user)
	前置条件	用户填写的必要信息不为空，且符合语法规范
	后置条件	记录新用户的数据
user.get_my_news	语法	function get_my_news (\$user_id)
	前置条件	用户 id 必须存在
	后置条件	获得这个用户的全部动态
user.get_info	语法	function get_info (\$user_id)
	前置条件	用户 id 必须存在
	后置条件	获得这个用户的全部基本资料
user.get_friends	语法	function get_friends (\$user_id)
	前置条件	用户 id 和 account_id 必须存在
	后置条件	获得这个用户的全部好友
user.get_account	语法	function get_account(\$user_id, \$account_id)
	前置条件	用户 id 必须存在
	后置条件	获得这个用户的账户
user.update_info	语法	function update_info (\$user_id)
	前置条件	用户 id 必须存在
	后置条件	更新这个用户的基本资料
user.add_friends	语法	function add_friends (\$user_id)
	前置条件	用户 id 必须存在
	后置条件	用户添加一个好友
user.delete_friends	语法	function delete_friends (\$user_id)

	前置条件	用户 id 必须存在
	后置条件	删除这个用户的一个好友
user.get_all_news	语法	function get_all_news ()
	前置条件	无
	后置条件	所有动态信息
user_friends_news	语法	function get_friends_news (\$user_id)
	前置条件	用户已经登录
	后置条件	用户好友的动态信息
user.update_account	语法	function update_account(\$user_id, \$account_id)
	前置条件	用户 id 和 account_id 必须存在
	后置条件	更新此用户的 account

➤ 业务逻辑层的动态模型

如下图显示在用户登录的时候用户逻辑处理的相关对象之间的协作。

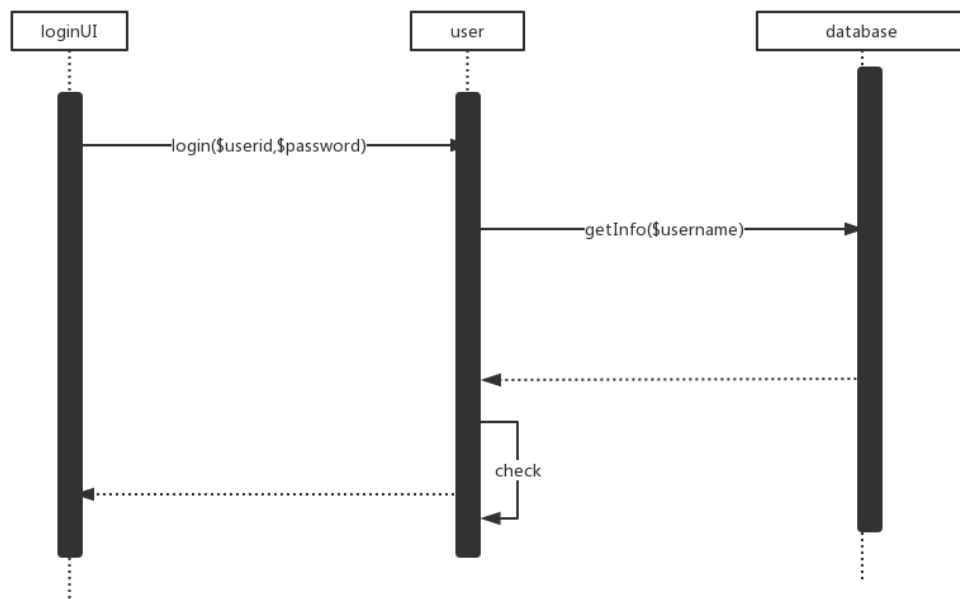


图 8

6.1.2.2. index 模块

➤ 模块概述

提供首页中显示运动信息、统计信息、动态信息，具体功能需求和非功能需求课参见 SPORTS 需求文档

➤ index 接口规范

提供的服务（供接口）		
index.get_user	语法	function get_user()
	前置条件	无
	后置条件	获得注册用户人数
index.get_sports	语法	function get_sports()
	前置条件	无
	后置条件	获取当前运动记录数
index.get_news	语法	function get_news()
	前置条件	无
	后置条件	获取动态总数

➤ 业务逻辑层的动态模型

如下图显示了获得全部动态的时候，业务逻辑处理的相关对象之间的协作。

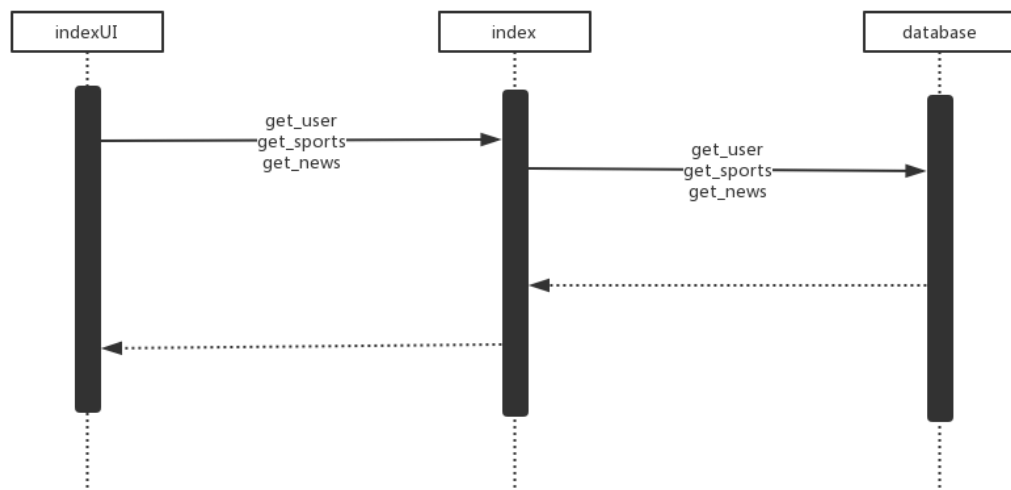


图 9

6.1.2.3. contest 模块

➤ 模块概述

contest 模块提供显示全部竞赛、发起竞赛、我的竞赛功能，具体功能需求和非功能需求课参见 **SPORTS** 需求文档

➤ **contest** 接口规范

提供的服务（供接口）		
contest.show	语法	function show()
	前置条件	无

contest.my_participate	后置条件	提供全部的竞赛
	语法	function my_participate ()
	前置条件	用户成功登陆
contest.add	后置条件	获得用户参与的竞赛
	语法	function add(\$contest)
	前置条件	用户成功登陆，用户输入合法
contest.get_history	后置条件	记录组织发布的竞赛
	语法	function get_history ()
	前置条件	无
contest.find	后置条件	获得已经的历史竞赛
	语法	function find(\$key, \$id)
	前置条件	用户输入的 key 符合条件
	后置条件	根据用户输入或者准确 id 查找比赛
	前置条件	
	语法	

➤ 业务逻辑层的动态模型

如图表明了 SPORTS 中,当添加一次竞赛的时候,业务逻辑处理的相关对象之间的协作。

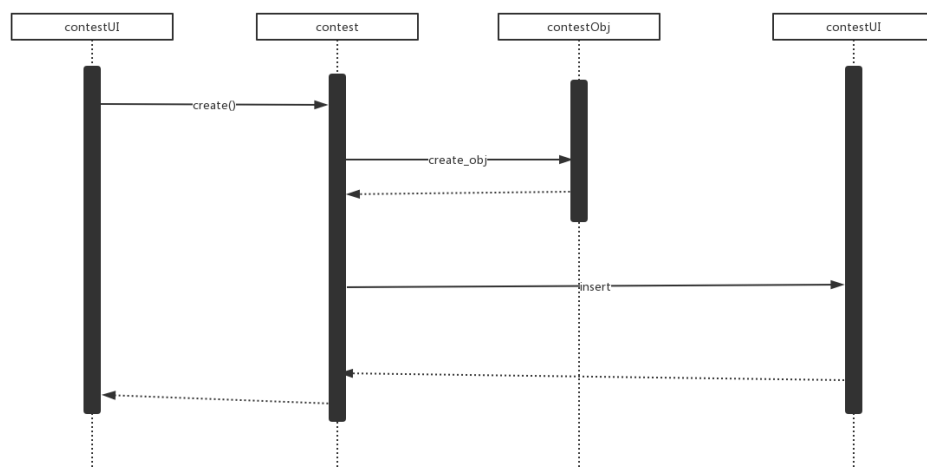


图 10

6.1.2.4. news 模块

➤ 模块概述

news 模块提供显示全部动态、发起动态、删除动态，搜索动态、朋友动态、我的动态功能，具体功能需求和非功能需求课参见 SPORTS 需求文档

➤ news 接口规范

提供的服务（供接口）

news.show	语法	function show()
	前置条件	无
	后置条件	提供全部的动态
news.my_participate	语法	function my_participate ()
	前置条件	用户成功登陆
	后置条件	获得用户参与的动态
news.add	语法	function add (\$news)
	前置条件	管理员成功登陆，用户输入合法
	后置条件	记录用户发布的动态
news.delete	语法	function delete (\$news_id)
	前置条件	用户成功登陆
	后置条件	删除用户发起的竞赛
news.get_history	语法	function get_history ()
	前置条件	无
	后置条件	获得已经过期的历史动态
news.find	语法	function find(\$key, \$id)
	前置条件	用户输入的 key 符合条件
	后置条件	根据用户输入或者准确 id 查找动态
news.friends	语法	function friends(\$username)
	前置条件	用户已登录
	后置条件	显示用户好友的所有动态

➤ 业务逻辑层的动态模型

如图表明了 SPORTS 中,当添加一次动态的时候,业务逻辑处理的相关对象之间的协作。

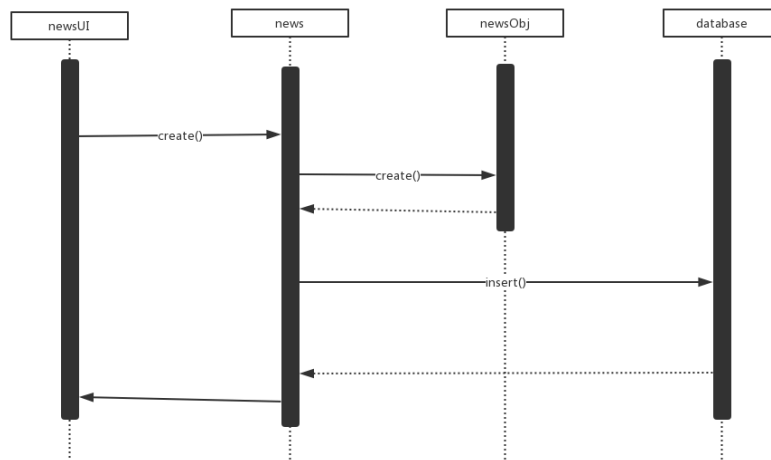


图 11

6.1.2.5. dataanalyze 模块

➤ 模块概述

dataanalyze 模块提供显示全部数据记录、数据分析、成长轨迹、运动推荐等功能，具体功能需求和非功能需求课参见 **SPORTS** 需求文档

➤ dataanalyze 接口规范

提供的服务（供接口）		
dataanalyze.my_history	语法	function my_history (\$username)
	前置条件	用户已经登录
	后置条件	提供全部的运动数据
dataanalyze.my_participate	语法	function my_participate (\$username)
	前置条件	用户成功登陆
	后置条件	获得用户参与的竞赛
dataanalyze.my_level	语法	function my_level (\$username)
	前置条件	用户成功登陆
	后置条件	记录用户的经验记录
dataanalyze.analyze	语法	function analyze (\$username)
	前置条件	用户成功登陆
	后置条件	返回基于用户数据的分析
dataanalyze.recommend	语法	function recommend(\$username)
	前置条件	用户已经登录
	后置条件	根据用户根据用户数据的推荐内容

➤ 业务逻辑层的动态模型

如图表明了 **SPORTS** 中，当添加一次兴趣组的时候，业务逻辑处理的相关对象之间的协作。

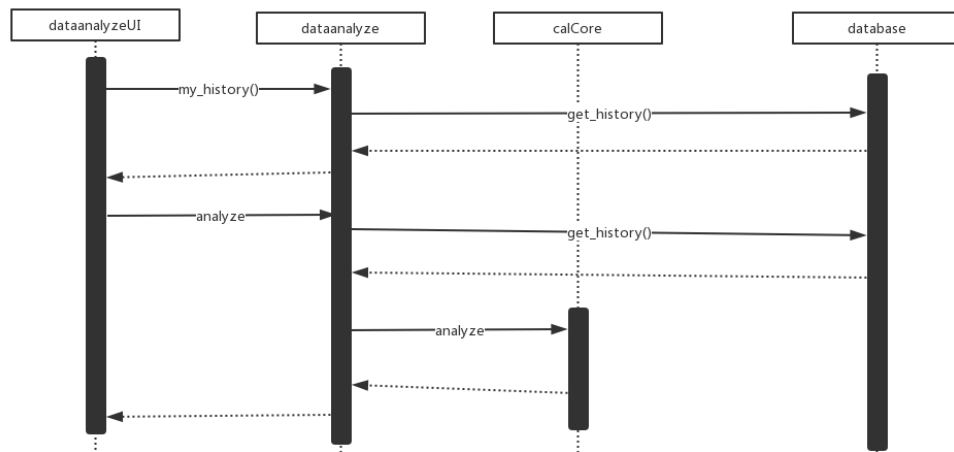


图 12

6.2. 数据层的分解

6.2.1. 数据层模块的职责

模块	职责
userdata	负责读取用户个人资料，好友，账户，通知，动态
sportsdata	负责读取全部运动记录
newsdata	负责读取全部动态，提供增删改查功能
contestdata	负责读取全部竞赛，提供增删改查功能

6.2.2. 数据层模块的接口规范

6.2.2.1. userdata 模块

➤ userdata 接口规范

提供的服务（供接口）		
userdata.show	语法	function show()
	前置条件	无
	后置条件	提供全部的信息

userdata.get_info	语法	function get_info(\$userid)
	前置条件	user_id 存在
	后置条件	提供用户的基本信息
userdata.set_info	语法	function set_info(\$user_id, \$user)
	前置条件	用户成功登陆，输入信息格式规范
	后置条件	修改用户的信息
userdata.get_friends	语法	function get_friends(\$user_id)
	前置条件	user_id 存在
	后置条件	获得用户的好友
userdata.add_friends	语法	function add_friends(\$user_id, \$friend_id)
	前置条件	user_id 和 friend_id 存在
	后置条件	添加好友
userdata.delete_friends	语法	function delete_friends(\$user_id, \$friend_id)
	前置条件	user_id 和 friend_id 存在
	后置条件	删除好友
userdata.get_news	语法	function get_news(\$user_id)
	前置条件	user_id 存在
	后置条件	获得用户的动态
userdata.get_notify	语法	function get_notify(\$user_id)
	前置条件	user_id 存在
	后置条件	获得用户的通知

6.2.2.2. sportsdata 模块

➤ sportsdata 接口规范

提供的服务（供接口）		
sportsdata.show	语法	function show()
	前置条件	无
	后置条件	提供全部的兴趣组
sportsdata.find	语法	function find(\$id)
	前置条件	id 输入合法
	后置条件	根据 id 查找兴趣组
sportsdata.add	语法	function add(\$interest)
	前置条件	用户输入格式正确
	后置条件	插入一个兴趣组到数据库

sportsdata.delete	语法	function delete(\$id)
	前置条件	删除的兴趣组为此用户发起的
	后置条件	根据 id 删除兴趣组
sportsdata.update	语法	function update(\$id)
	前置条件	更改的兴趣组为此用户发起的
	后置条件	根据 id 更新兴趣组

6.2.2.3. newsdata 模块

➤ newsdata 接口规范

提供的服务（供接口）		
newsdata.show	语法	function show()
	前置条件	无
	后置条件	提供全部的活动
newsdata.find	语法	function find(\$id)
	前置条件	id 输入合法
	后置条件	根据 id 查找活动
newsdata.add	语法	function add(\$interest)
	前置条件	用户输入格式正确
	后置条件	插入一个活动到数据库
newsdata.delete	语法	function delete(\$id)
	前置条件	删除的活动为此用户发起的
	后置条件	根据 id 删除活动
newsdata.update	语法	function update(\$id)
	前置条件	更改的活动为此用户发起的
	后置条件	根据 id 更新活动

6.2.2.4. contestdata 模块

➤ contestdata 接口规范

提供的服务（供接口）		
contestdata.show	语法	function show()
	前置条件	无
	后置条件	提供全部的竞赛

contestdata.find	语法	function find(\$id)
	前置条件	id 输入合法
	后置条件	根据 id 查找竞赛
contestdata.add	语法	function add(\$interest)
	前置条件	用户输入格式正确
	后置条件	插入一个竞赛到数据库
contestdata.delete	语法	function delete(\$id)
	前置条件	删除的竞赛为此用户发起的
	后置条件	根据 id 删除活动

7.架构与组件设计

7.1. 系统架构

本系统为动态网站，采取 BS 模式，利用 apache 或 nginx 做服务器，由 php 操纵 sqlite 数据库，获得所需数据。具体工作流程如下：

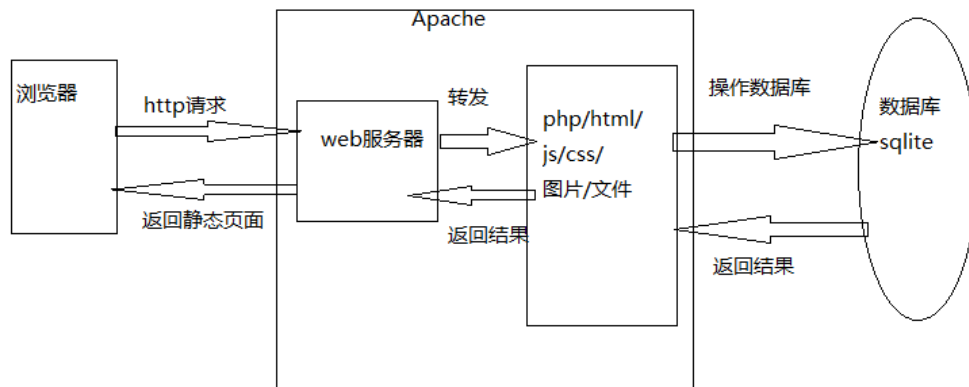


图 13

组件样式采用 bootstrap 中的组件设计，并对其部分组件进行更改，符合系统的风格特性。

7.2. 组件设计

7.2.1. 图标（采用 font awesome 图标系统）



图 14

7.2.2. 动态

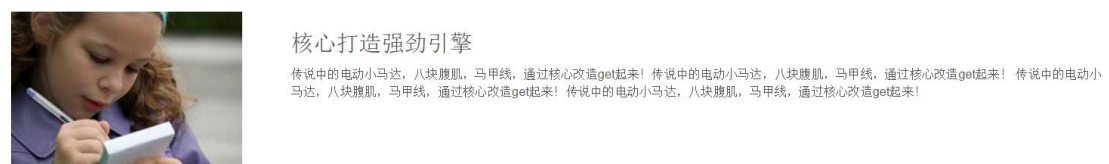


图 15

7.2.3. 按钮

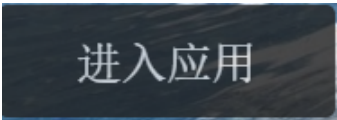


图 16

7.2.7. 分页



图 20

7.2.8. 搜索框

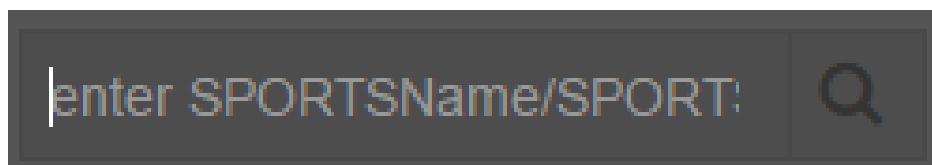


图 21