**武汉大学计算机学院**

**本科生实验报告**

**活动平台Eventplatform**

专 业 名 称 ：计算机科学与技术

课 程 名 称 : DotNet架构程序设计

团 队 名 称 ：

指 导 教 师 一：贾向阳

团 队 成 员 一：李政民（2018302110388）

团 队 成 员 二：吴文龙（2018302110061）

团 队 成 员 三：颜塨洵（2016301500104）

团 队 成 员 四：孙元锴（2018302080349）

二○二○年十二月

**郑 重 声 明**

本团队呈交的实验报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本实验报告不包含他人享有著作权的内容。对本实验报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本实验报告的知识产权归属于培养单位。

团队成员签名： 李政民、吴文龙、颜塨洵、孙元锴 日期：2020.12.05

摘 要

一、实验目的：

1. 掌握C#语言的基本使用。
2. 熟练掌握.NET集成开发环境

二、遵循原则：由浅入深、由简到繁、逐步递进。

三、实验内容：

搭建一个从注册活动到活动发布、普通用户报名、以及申诉处理的平台，有一个统一的活动平台，放那边用户查看各类活动也能够方便用户归类管理自己的活动档案。当然目前我们这个平台的受众主要是大学以及大学生，大学里的各类社团能够在某一个平台上发布活动，学生也能够在这个平台上找到各类活动信息同时能够方便的查看自己的活动档案，为后来需要的各类评选活动需要提供便利；

在这里面我们使用了.NET Core web MVC框架，使用Razor语法开发前端，使用 C#开发后端，数据库使用的是MySQL。

**关键词：MVC、Razor、C#、MySQL、活动管理平台**

**目录**

[1 项目概述 5](#_Toc58076965)

[1.1 项目目标 5](#_Toc58076966)

[1.2 项目需求 5](#_Toc58076967)

[1.3 主要功能 6](#_Toc58076968)

[1.3 人员分工 13](#_Toc58076969)

[2 项目技术概要 14](#_Toc58076970)

[2.1 后端部分 14](#_Toc58076971)

[2.1.1 管理员模块设计 14](#_Toc58076972)

[2.1.2普通用户（参赛者）模块设计 16](#_Toc58076973)

[2.1.5登录模块 18](#_Toc58076974)

[2.2 1前端部分 20](#_Toc58076975)

[2.2.1 管理员页面 20](#_Toc58076976)

[2.2.2 剪贴板加密解密功能 20](#_Toc58076977)

[2.2.3 文件加密解密功能 20](#_Toc58076978)

[2.2.4 右键快捷加密解密功能 20](#_Toc58076979)

[2.2.5 登录界面 20](#_Toc58076980)

[2.2.6 查看可比赛界面 21](#_Toc58076981)

[2.2.7 查看已参加的比赛界面 21](#_Toc58076982)

[2.3 其他部分 22](#_Toc58076983)

[2.3.1 Web爬虫 22](#_Toc58076984)

[2.3.2 云端服务器 22](#_Toc58076985)

[3 小组自我评价 23](#_Toc58076986)

[3.1 难度 23](#_Toc58076987)

[3.2 代码量 23](#_Toc58076988)

[3.3 亮点 25](#_Toc58076989)

[3.4 收获 27](#_Toc58076990)

# 1 项目概述

## 1.1 项目目标

本实验需要设计一个可以从主办方注册活动，管理员审批活动，主办方发布活动到普通用户可以报名参加活动的这么一个完整闭环的活动平台。

## 1.2 项目需求

表**1.1**项目板块设计

|  |  |
| --- | --- |
| 模块功能 | 功能说明 |
| 管理员 | 编辑个人信息、注销个人账户、查看活动信息、审核活动（分别是待审核、已通过、已拒绝以及需要延期）并做出拒绝或者接受回馈给主办方 |
| 主办方 | 编辑个人信息、注销个人账户、填写活动注册表，向管理员提交活动详情，发布活动公告、筛选报名人员以及公布活动结果 |
| 普通用户 | 编辑个人信息、注销个人账户、查看活动信息、报名参赛以及退赛、查看自己的活动记录 |

主办方sponsor，普通用户participant，活动event、管理员Administrator以及普通用户参赛event\_participant在数据库中都有一张表与其对应；





图**1.1** 项目整体框架类图

## 1.3 主要功能

1.登录界面包含登录以及注册的功能；

2.管理员审核活动具体操作是可以接受活动、拒绝活动、提交建议，修改个人信息，查看活动信息；

3.参赛者可以查看个人信息以及修改个人信息，查看所有准许开展的活动以及查看自己已参加的活动，活动报名和在审核成功后退赛；

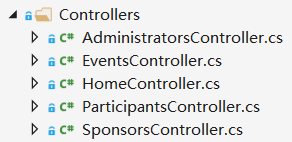
4.主办方可以查看个人信息以及修改个人信息，填写活动注册表，向管理员提交活动详情，发布活动公告，筛选报名人员以及公布活动结果；

**1.4设计思路**

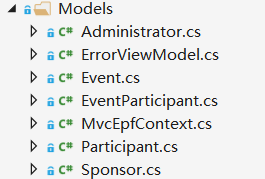
在这个活动平台系统中，出于将前后端分离，能够更好地实现跨平台及项目迁移的目的，本打算采用的是Spring框架，但是后面考虑到目前还不太熟悉前后端开发的完整流程，而且基于眼前的功能来说，这个项目的功能过于简陋，所以我最后放弃了Spring框架，选择了MVC框架，但是并不是完全的单独使用MVC的三个板块，本小组在里面添加Service层，目的是为了在较简单的情况下将我的项目中视图层和逻辑业务层以及数据层分开管理，降低耦合度，事实上对于我们这个项目来说是降低了耦合度。

简单的思路：Controller里面你负责的是接受前端的消息数据（URL请求，每个URL就在后台指明了调用哪个controller，以及调用哪个action，然后带了些什么参数）然后调用Service去执行相关的功能，再由Service调用数据库的资源，依据前端发来的请求去处理，将得到的结果返回给Controller，再由Controller调用View层去显示前端页面。

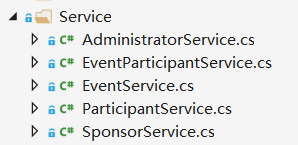
项目划分为Model、Service、Controller以及View四层如下图：



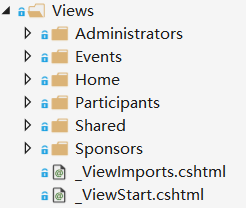
图**1.2** Controllers层结构图



图**1.2** Models层结构图



图**1.2** Service层结构图

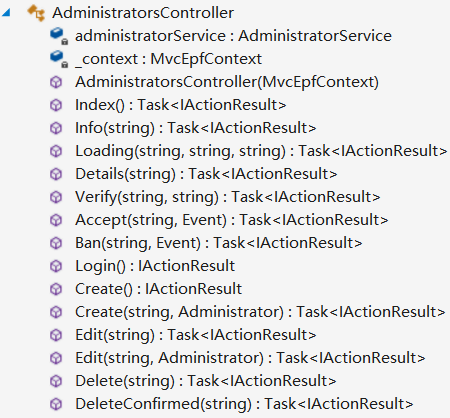


图**1.2** Service层结构图

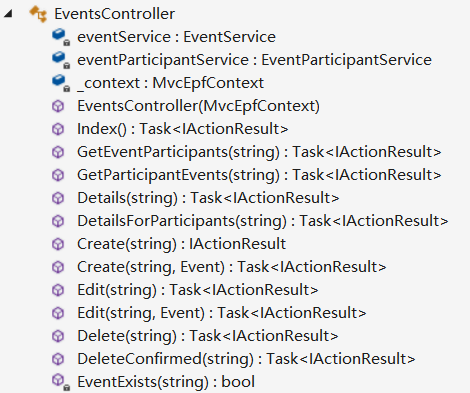
2.每个模块的功能：

Controller里负责的是接受前端的消息数据然后调用Service去执行相关的功能，再由Service调用数据库的资源，依据前端发来的请求去处理，将得到的结果返回给Controller，再由Controller调用View层去显示前端页面。

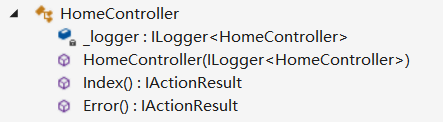
在Controller层里面就是处理各类URL的指定action和接受数据，然后在Service里面是各种action对应的方法的实体，在service里面才能操作数据库。下表是项目的controller层和前端交互需要做出的一些action，包括注册，修改、以及他们对用的一些功能需求。



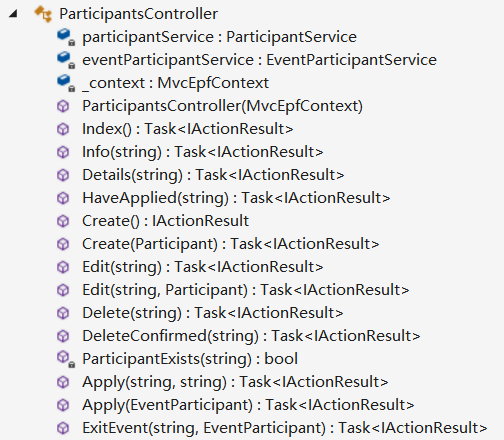
图**1.3** AdministratorsController结构图



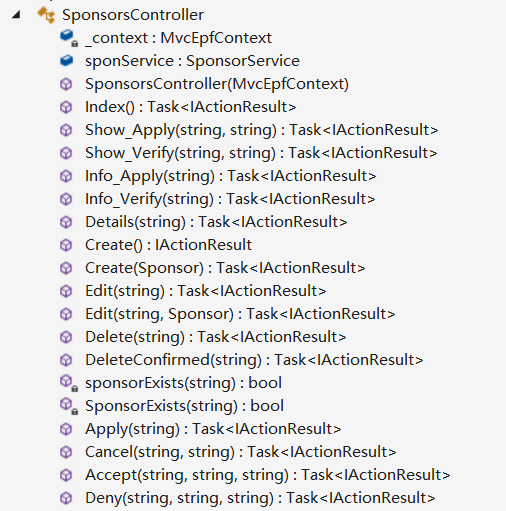
图**1.4** EventsController结构图



图**1.5** HomeController层结构图



图**1.6** ParticipantsController层结构图



图**1.7** SponsorsController层结构图

**1.5 人员分工**

李政民负责举办方模块的前后端设计和数据库设计；

吴文龙负责登录界面相关功能以及管理员模块的前后端设计；

颜塨洵负责参赛者模块的前后端设计；

孙元锴负责EP表设计。

# 2 项目技术概要

## 2.1 后端部分

项目模块主要包括model、controller、service以及view四层的设计，model就是设计管理员实体，在数据库建立对应的表，里面就是需要说明相关的表项。重要的就是在设计service层和controller层，一开始我们小组单纯的仅仅使用controller层是进行前后端的交接操作，简单地以为这就是将前后端分离了，后来在老师的一连串问题下才明白MVC还是的controller层还是将对前后端的操作扭合在一起了，处理前端后操作后端都是在controller里每个action就完成了，这样看起来确实挺方便的，但是考虑到需要平台迁移呢？那我们这个就无法轻松的解决这个问题，最后在老师的指引下我们引入了service层，在controller中主要接受前端传来的数据，在service层就是一句前端传来的指令，数据区队数据库进行操作，将返回的结果通过controller调用前端view层呈现给用户，因此来达到交互的目的，同时这样也能很好的进行项目迁徙。

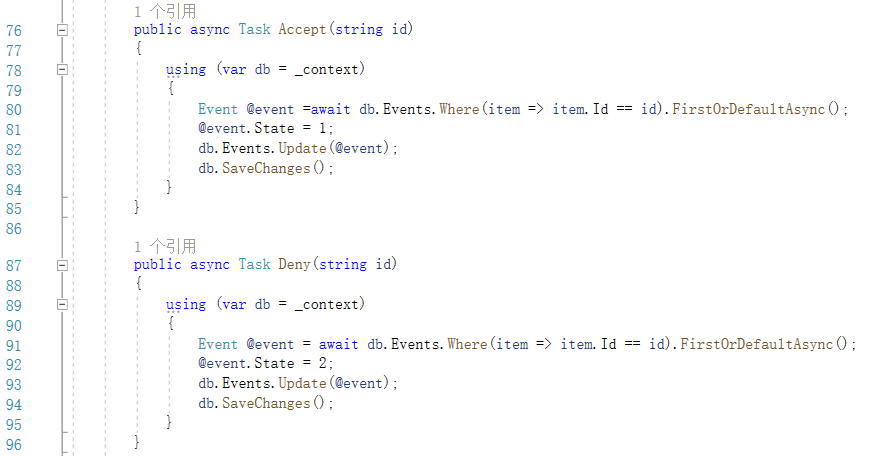
### 2.1.1 管理员模块设计

管理员主要功能就是罗列出所有的活动，按照活动的状态（待审核，已审核）来分区展示在管理员的工作台上，管理员可以查看这些活动的信息，进行拒绝或者同意的操作。

Controller中重要的方法包括修改活动状态的Accept()和ban()，查看其活动信息的Verify()、查看个人的Details()和删除账号的Delete()；

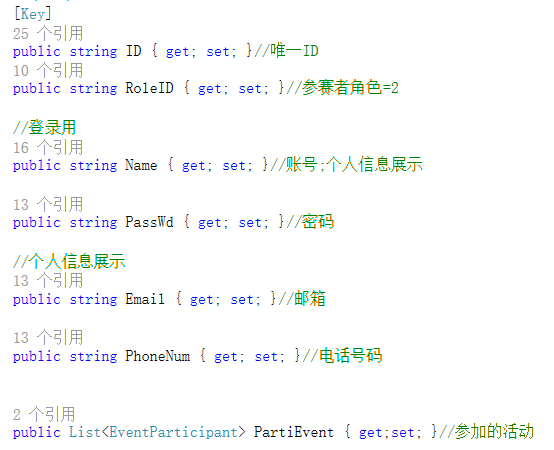
图**2.1** 管理员controller层中的部分操作

Service层是针对controller里每个action对应的操作数据库的方法实现，具体是在从头roller层中声明一个对象service，controller中的action接收到前端管理员的请求后就会调用service对象中的具体方法，最后service通过数据查询数据库，进行对数据库这个表进行添加、修改、删除、查找等具体操作。

图**2.2** 管理员service层中的操作数据库

### 2.1.2普通用户（参赛者）模块设计

普通用户模块主要包括model、controller、service以及view四层的设计，model层的participant类就是设计普通用户实体，包含数据库中participants表所有的字段，以及一个EventParticipant类型的表实现一对多的逻辑。



图**2.3** 普通用户model层中的属性

由于本次大作业体量不大，故将dao层对数据库的操作融入service层，包括对参赛者基本的增删改查，以及报名、查看已参加比赛、退赛。查看可报名的比赛则是直接使用框架封装的方法访问Events表符合条件的字段，故不需service层方法。



图**2.3** 普通用户service层部分方法

Controller层调用service层方法并将参数传入view层或传出，并重写了框架提供的Details、Info、Create、Delete等方法，实现登录后的主界面、个人信息的增删改查。返回值的设置则负责方法使用完后跳转到哪个界面。



图**2.4** 普通用户controller层部分方法

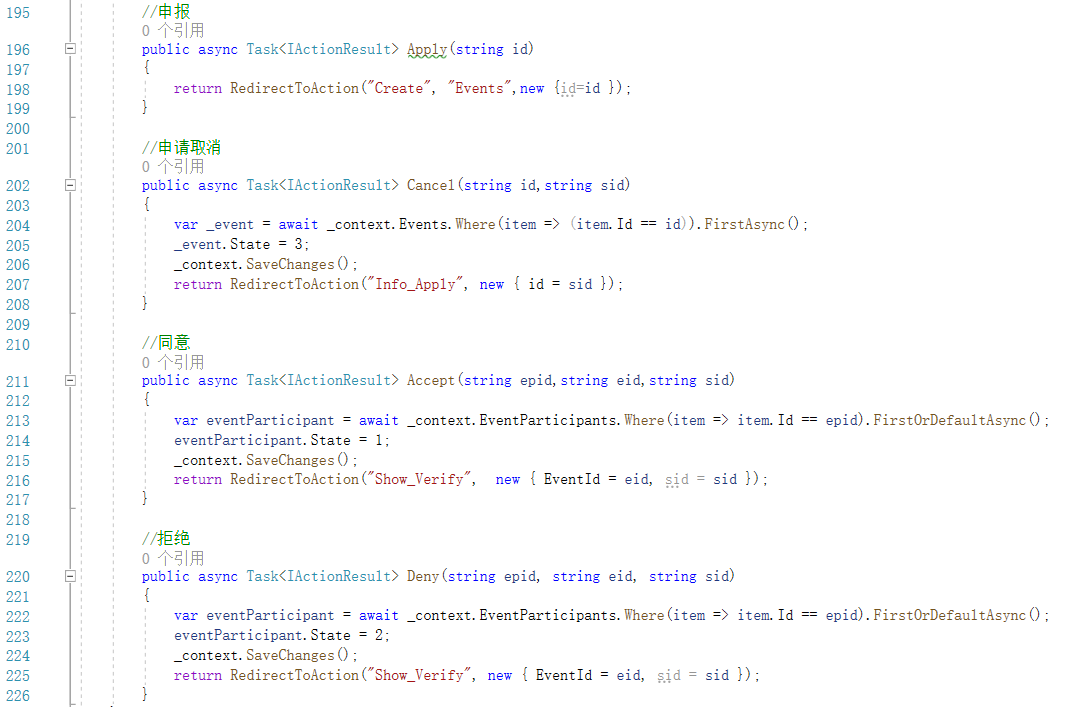
## 

### 2.1.3主办方模块设计

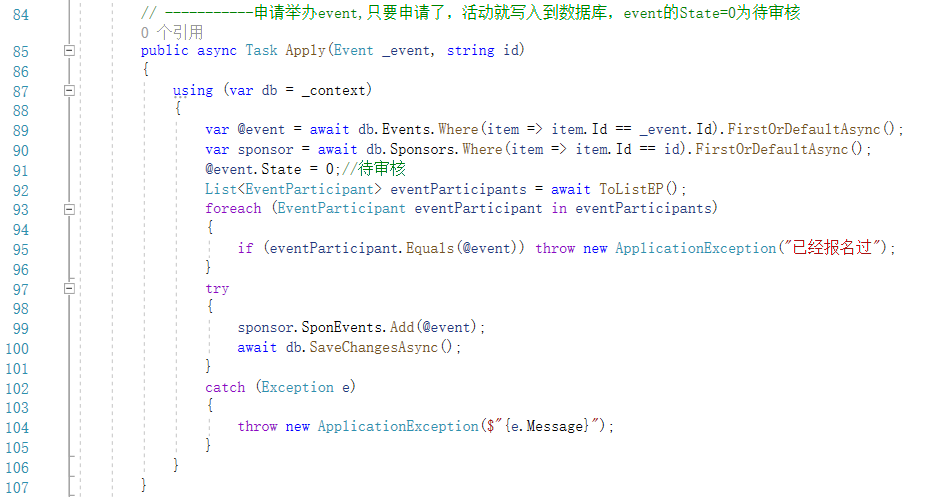
主办方主要功能就是申报活动和查看所有已经申报的活动，按照活动的状态（待审核，已审核）来分区展示在工作台上，举办方可以查看这些活动的用户报名信息并进行审核。



图**2.5** 主办方controller层中的部分操作



图**2.6**  主办方controller层中的部分操作



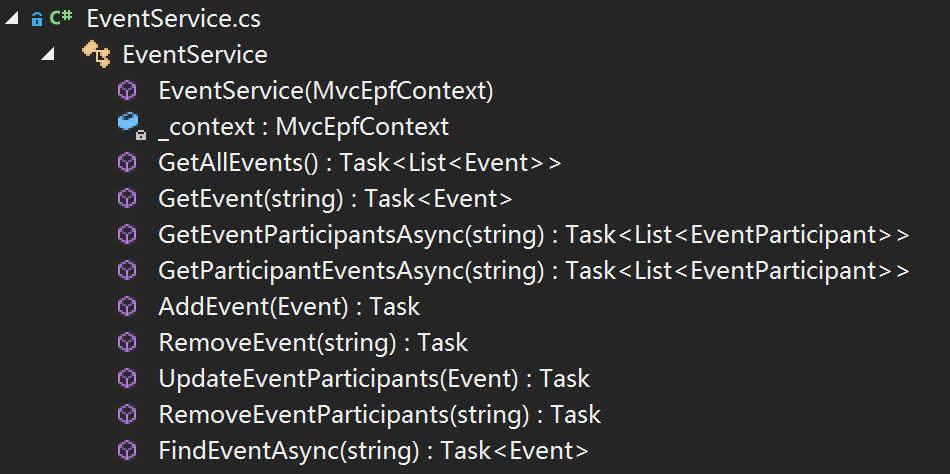
图**2.7**  主办方service层中的部分操作

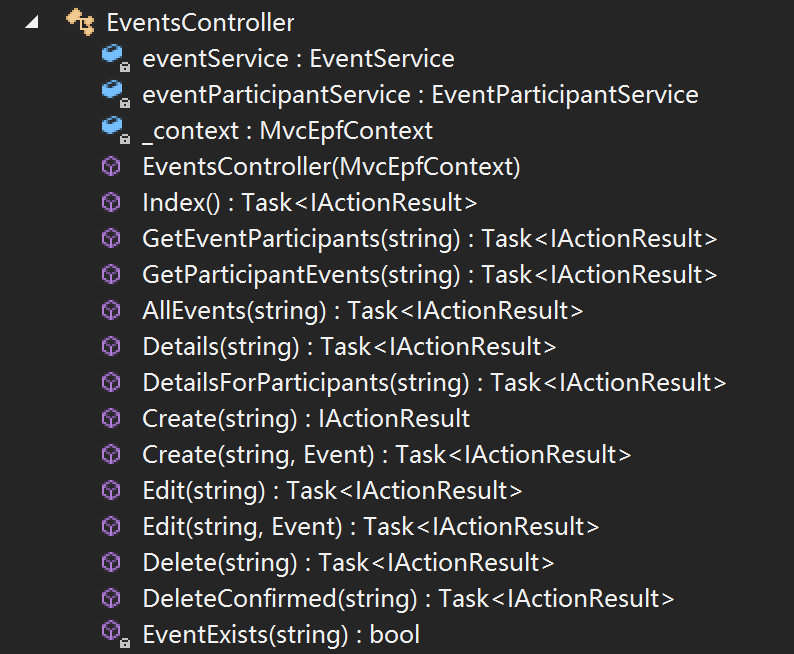


图**2.8**  主办方service层中的部分操作

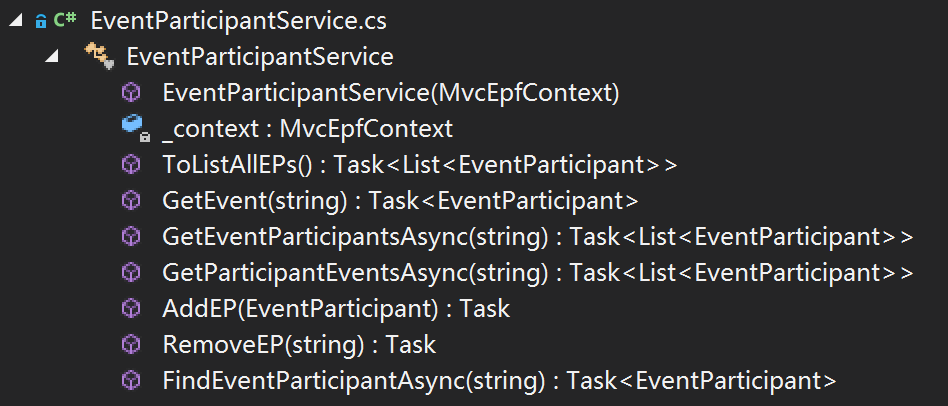
### 2.1.4活动模块设计

活动模块主要包含有Event和EventParticipant两部分。它们分别都包括model、controller、service三层的设计，

相关Event的部分，model层的Event类就是设计活动实体，包含数据库中Events表所有的字段，service层中的EventService类包含有它的服务层面功能实现，主要包括有通过EventId进行查询，查找单个Event中的所有Participant，以及一些基础的增删改查功能。Event部分的Controller层面主要是针对Administer设计的，用于在管理员界面中显示所有活动的具体信息.

图**2.9**  Event service层中的方法

**图2.9**  Event controller层中的方法

相关EventParticipant的部分，该部分对应一个参与者在一项活动中的报名，主要是为了服务Sponsor查看自己举办的event中有哪些参与者，也同时服务于Participant来查看自己所参加的所有event，model层呈现单个event对应一个participant报名所应该具有的字段，在其service中，主要针对event和participant之间关联操作进行设计，包括有通过对EventParticipants表查找返回一个event中所有的participant，也有返回一个participant所参加的所有event。

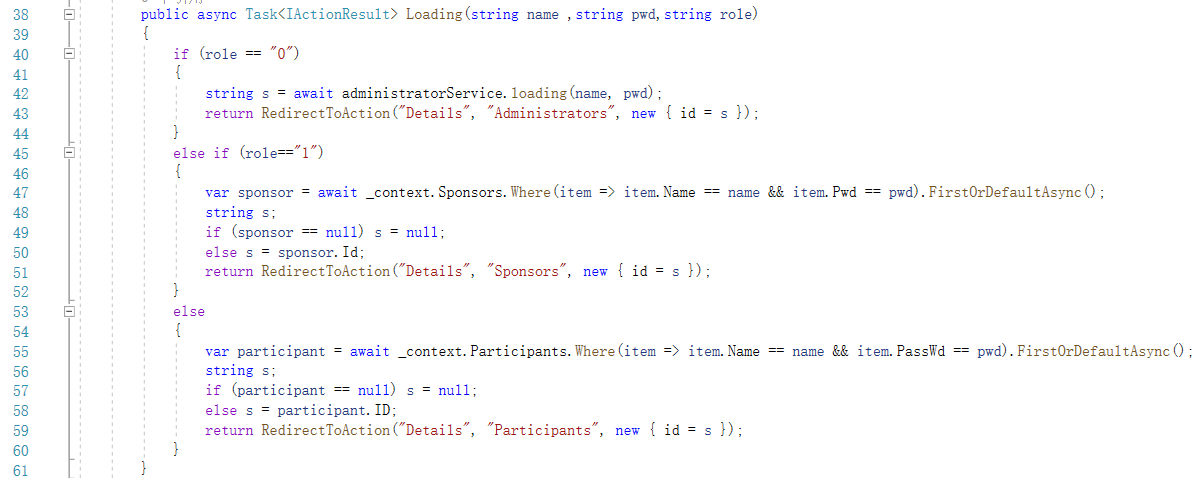
**图2.10**  EventParticipant service层中的方法

### 2.1.5登录模块设计

登录模块的设计思想：包含登录和注册

登录：前端接收到账号和密码后，将数据传给loading()方法，根据前端传来的角色id去各自对应的controller中找，找到了匹配的就返回相应的界面信息，没找到就回到登录界面；

注册：选择角色id后，发送注册的url，将注册界面返回后用户注册；



图**2.9**  登录注册方法

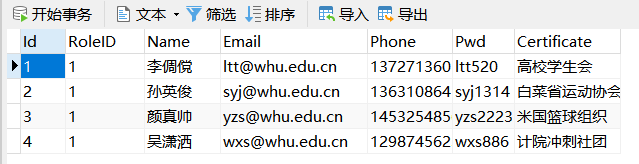


图**2.10**  前端处理注册请求

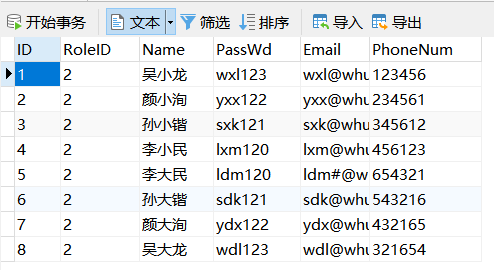
### 2.1.6数据库设计



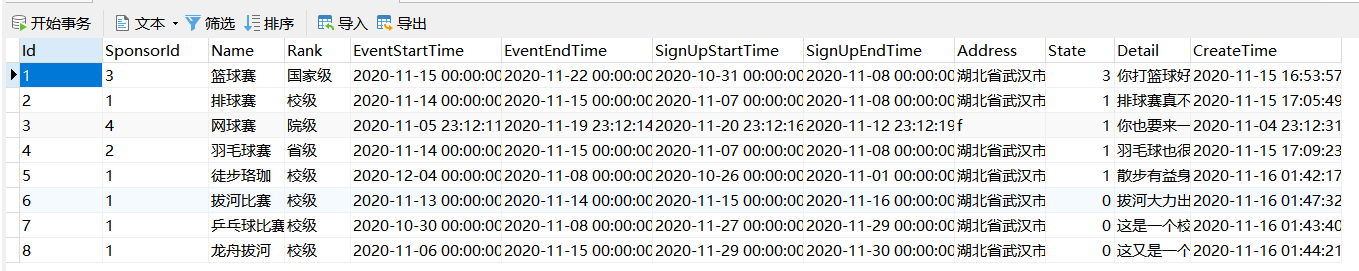
图**2.11** 管理员数据库内容设计



图**2.12** 举办方数据库内容设计



图**2.13** 普通用户数据库内容设计



图**2.14** 活动数据库内容设计

## 2.2 前端部分

### 2.2.1 管理员页面

管理员主要用到的页面有查看个人和活动信息的Details.cshtml和Verify.html，管理员工作台首页面Info.cshtml和修改信息的edit.cshtml；管理员课后台的交互就是在这几个页面表现出来，后台根据管理严发过来的url，在controller中调用service将处理结果传给view，最后展示出来，以此达到交互的目的；

### 2.2.2 普通用户页面

普通用户可以使用的界面除了开始的登录界面外，个人信息方面还有查看个人信息的Details.html和修改个人信息的Edit.html，以及注册使用的Create.html和删除用Delete.html，赛事方面则有查看可参加活动的Info.html，查看已参加比赛的HaveApplied.html，参赛的Apply.html，退赛的ExitEvent.html。

### 2.2.3 主办方页面

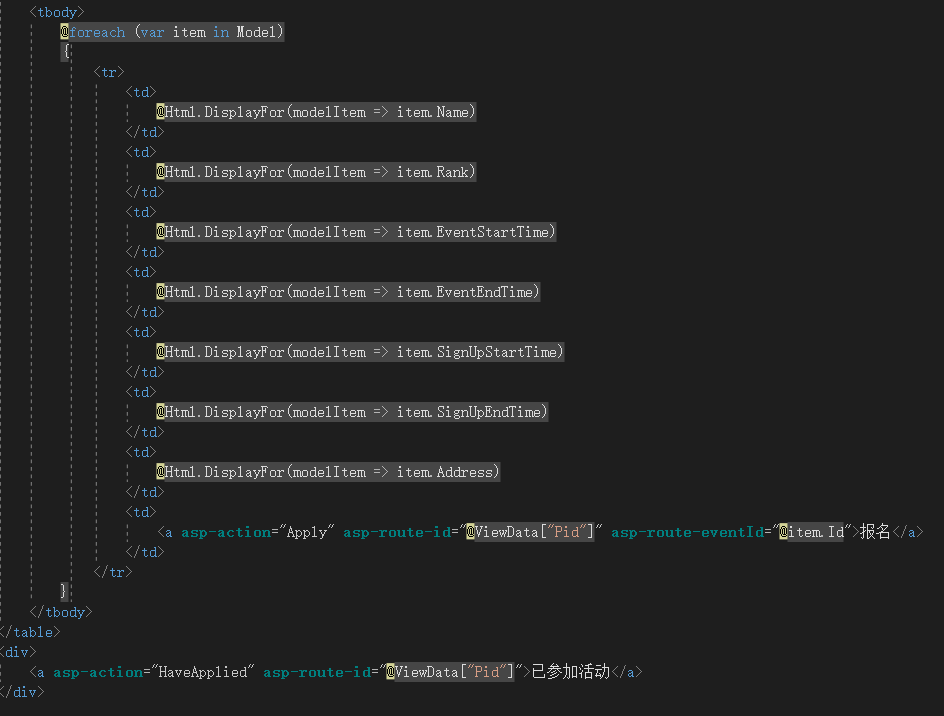
主办方主要用到的页面有查看个人信息页面Details.cshtml，修改信息页面Edit.cshtml，删除现有个人信息页面Delete.cshtml，查看活动页面Info.cshtml，申报活动页面Create.cshtml，审核报名页面Info\_Verify.cshtml，展示活动详细信息页面Show\_Apply.cshtml，展示报名用户详细信息页面Show\_Verify.cshtml。

### 2.2.4 登录界面

根据选择的角色id，以及输入的账号和密码作为一条URL发送给后台：调用相应的登录方法找到匹配的就见结果返回给前台

### 2.2.5 查看可参加比赛界面

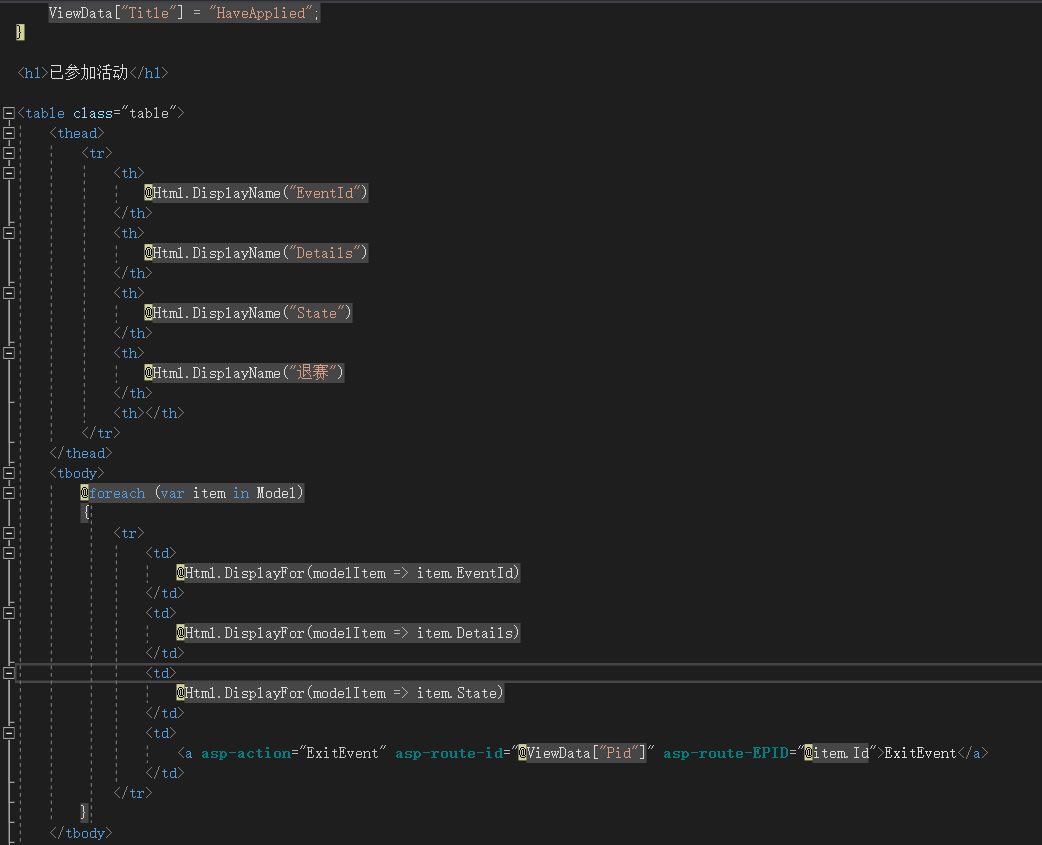
直接在controller层查找Events表里所有管理员已批准的活动显示到前端。且已完成用户的ID传入，可以直接跳转到查找已参加的比赛界面。



普通用户登录后直接跳转到该界面

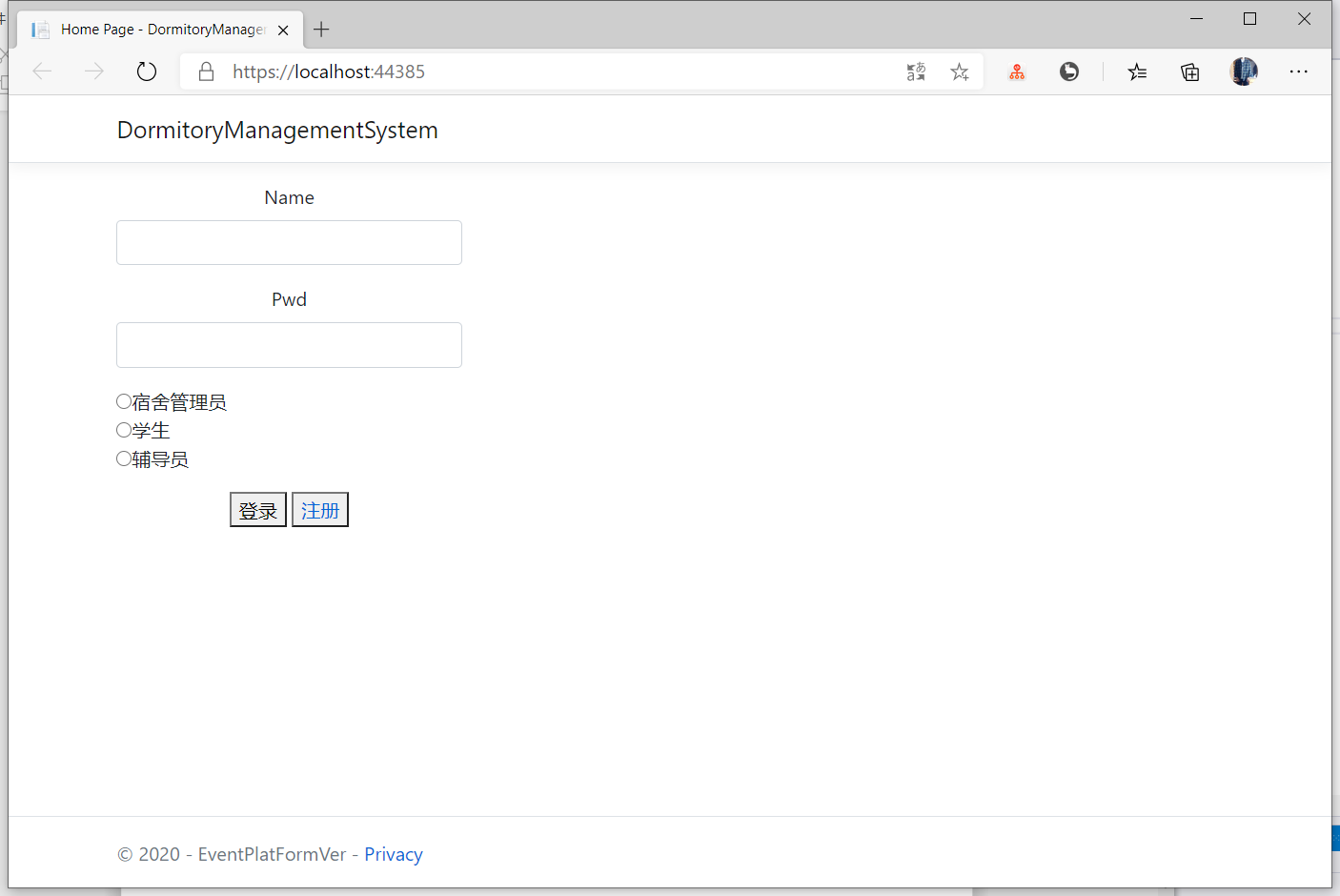
### 2.2.7 查看已参加的比赛界面

将当前使用方法的角色ID作为一条URL发送给后台，调用相应的方法去EP表里寻找所有ParticipantID符合且State不为已退赛的结果返回给前端。

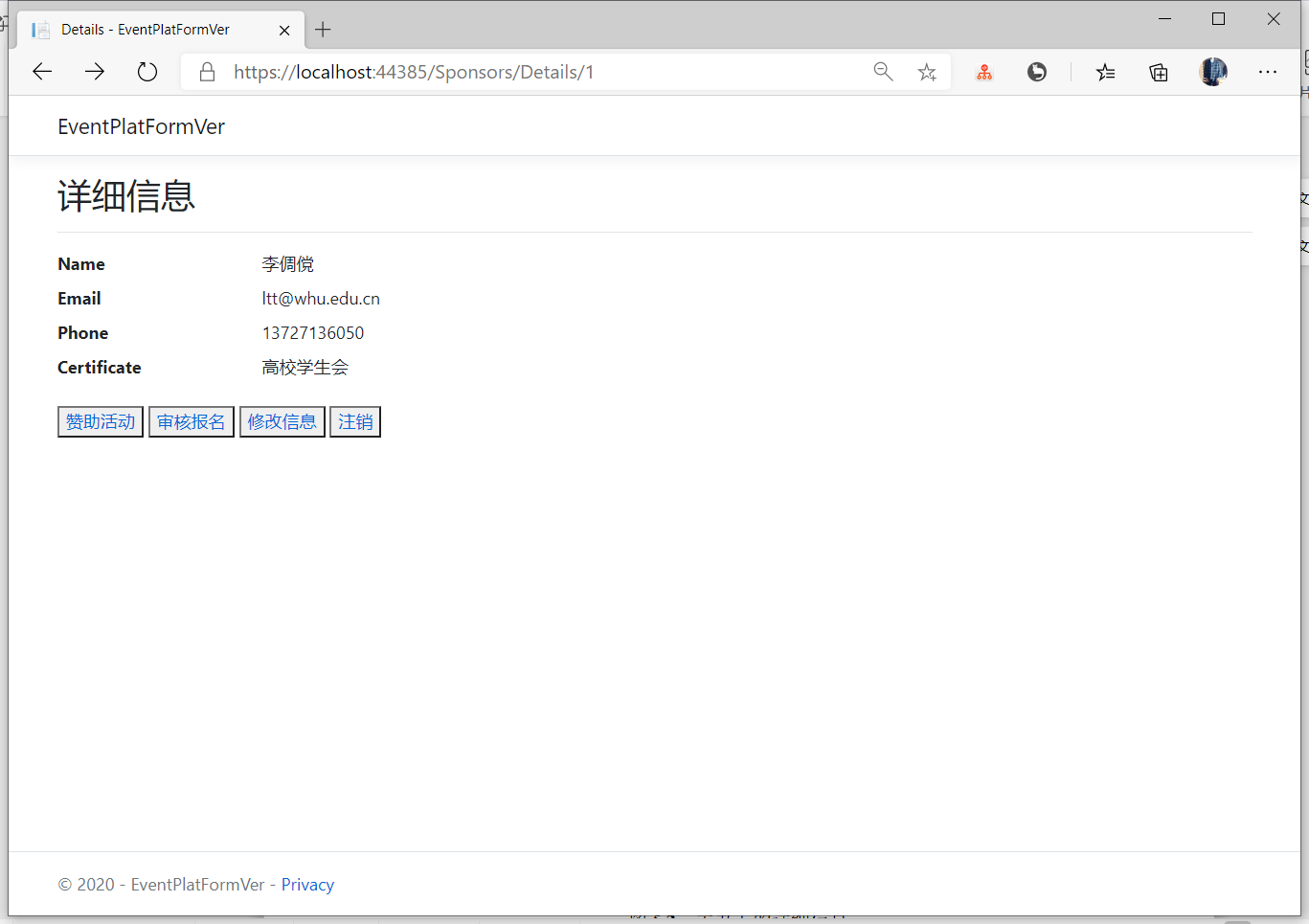


这里为了方便调试显示了几条后端才可见的字段，实际可根据需要调整。

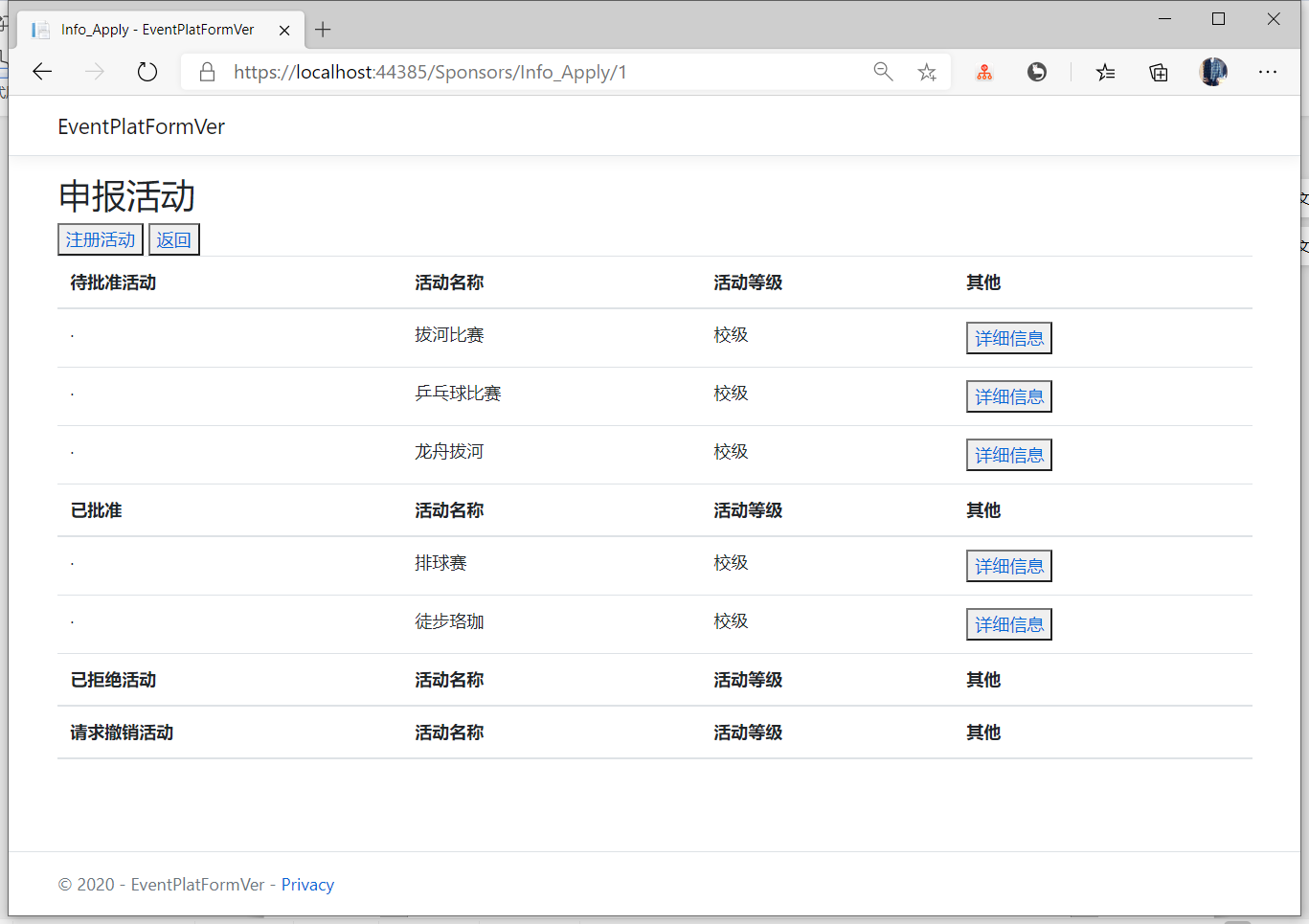
# 3 效果展示



图**3.1** 登录界面



图**3.2** 主办方的详细信息



图**3.3**  主办方查看活动

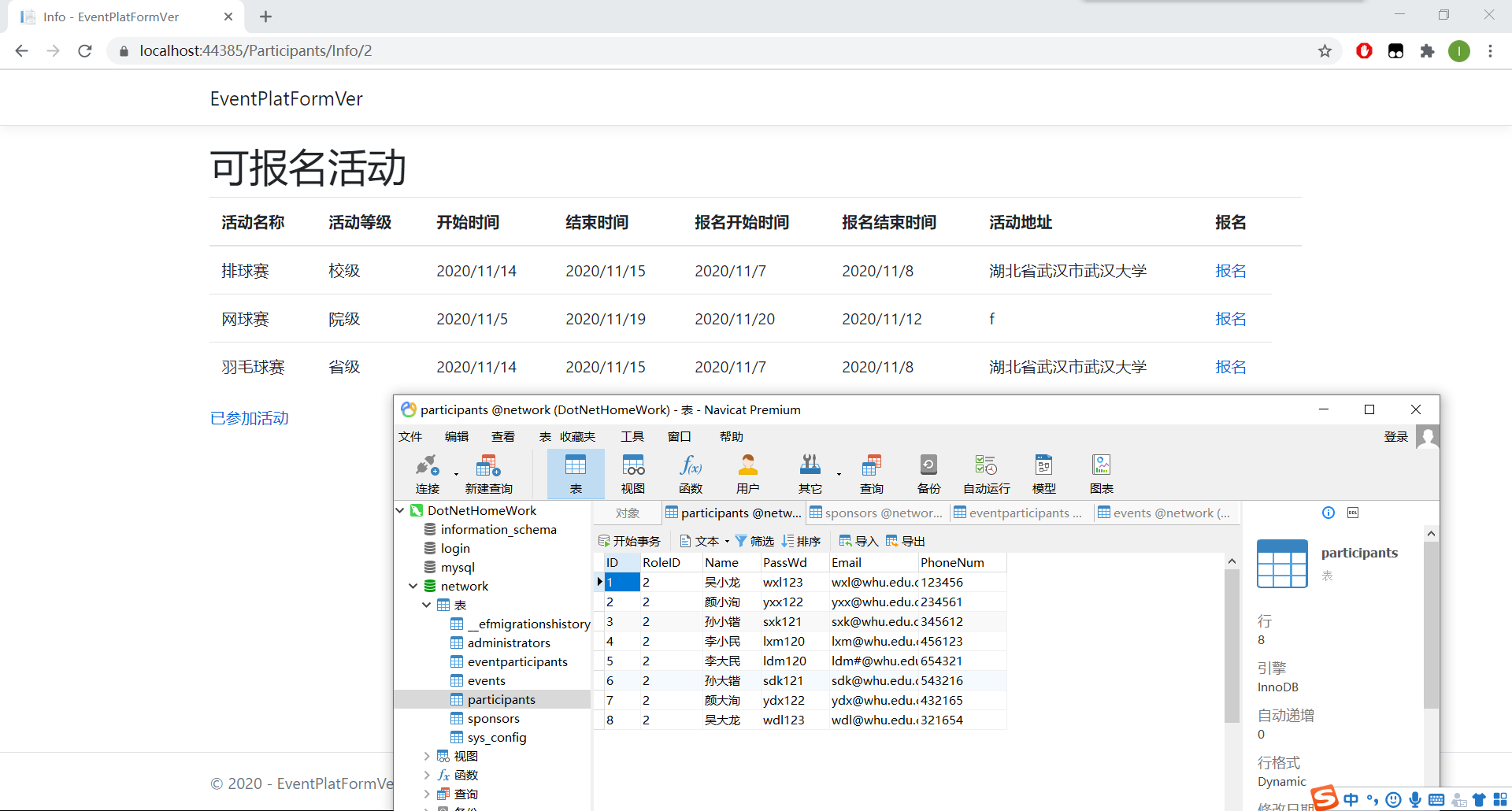


图3.4 使用数据库有的参赛者账号登录，成功进入可报名活动界面

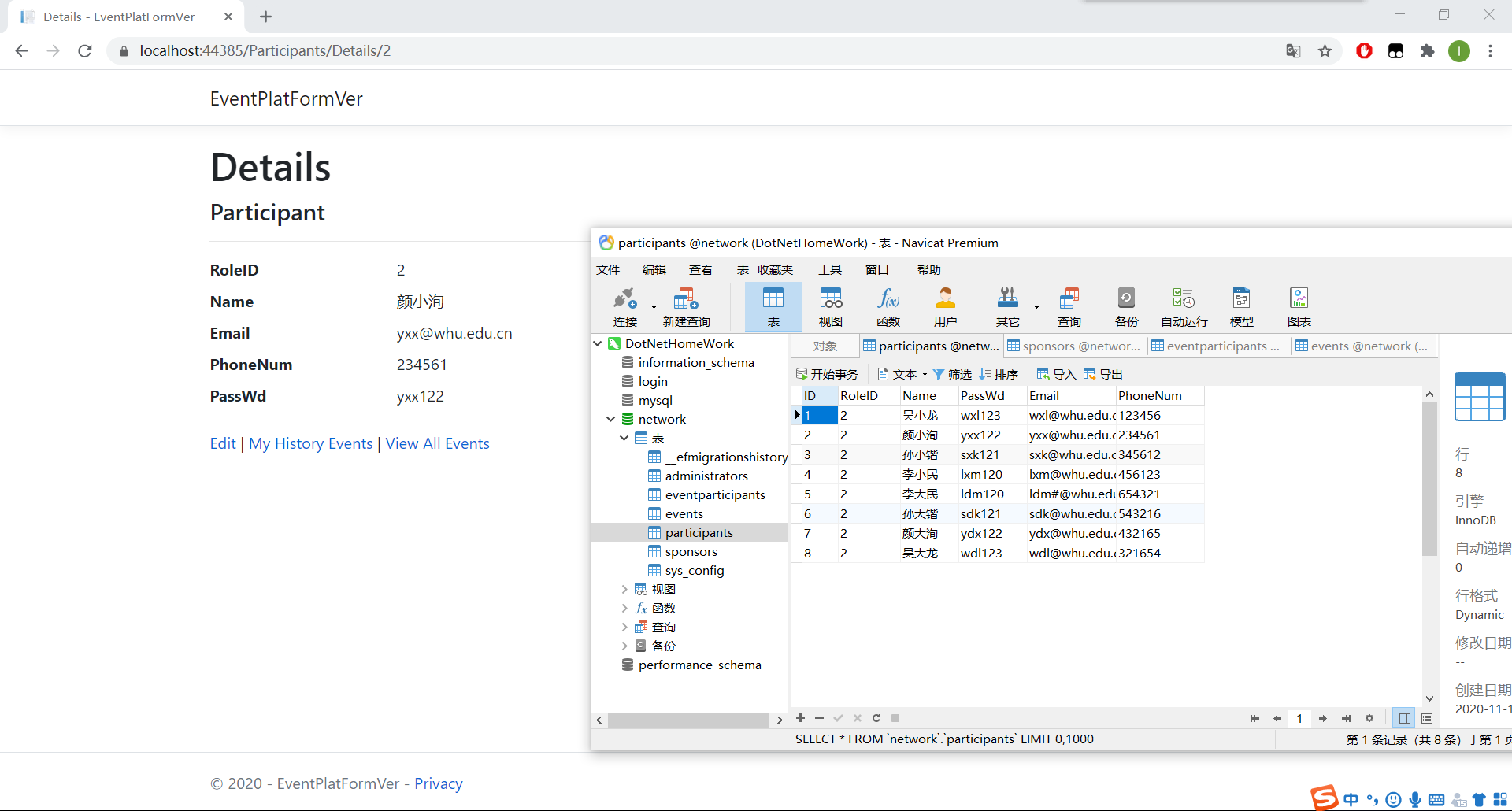


图3.5 个人信息界面

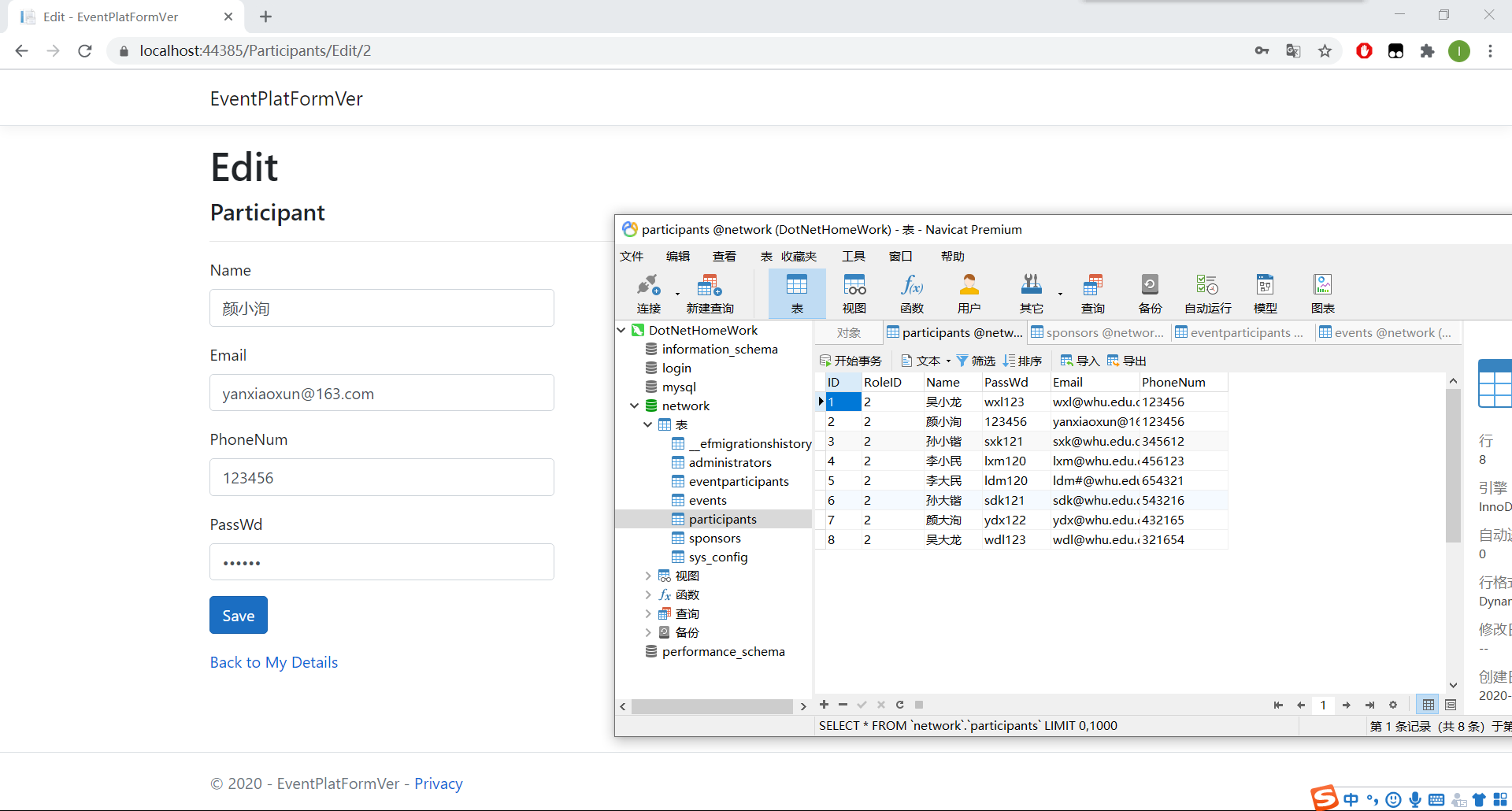


图3.6 修改个人信息界面

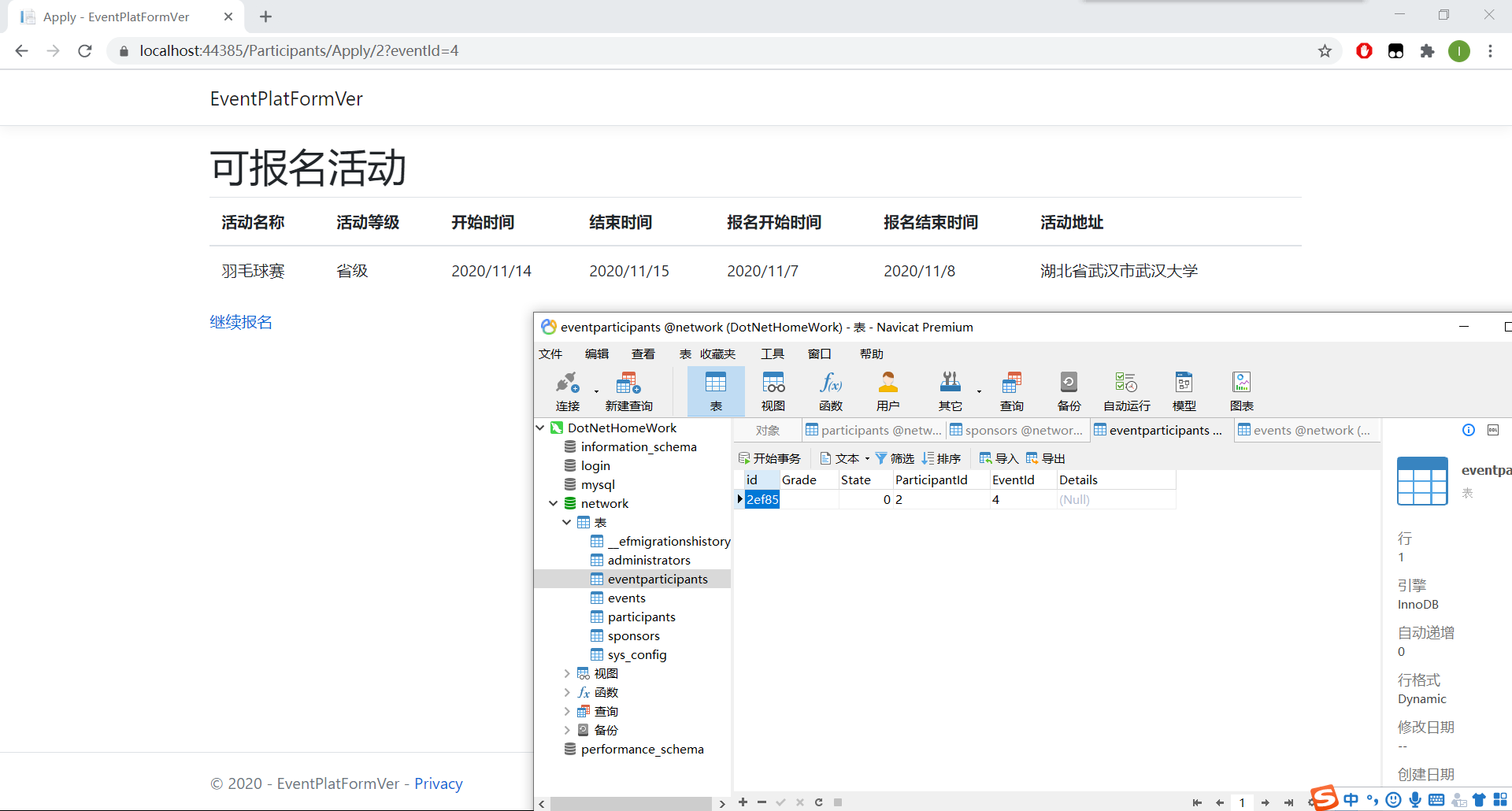


图3.7 点击报名按钮后的界面

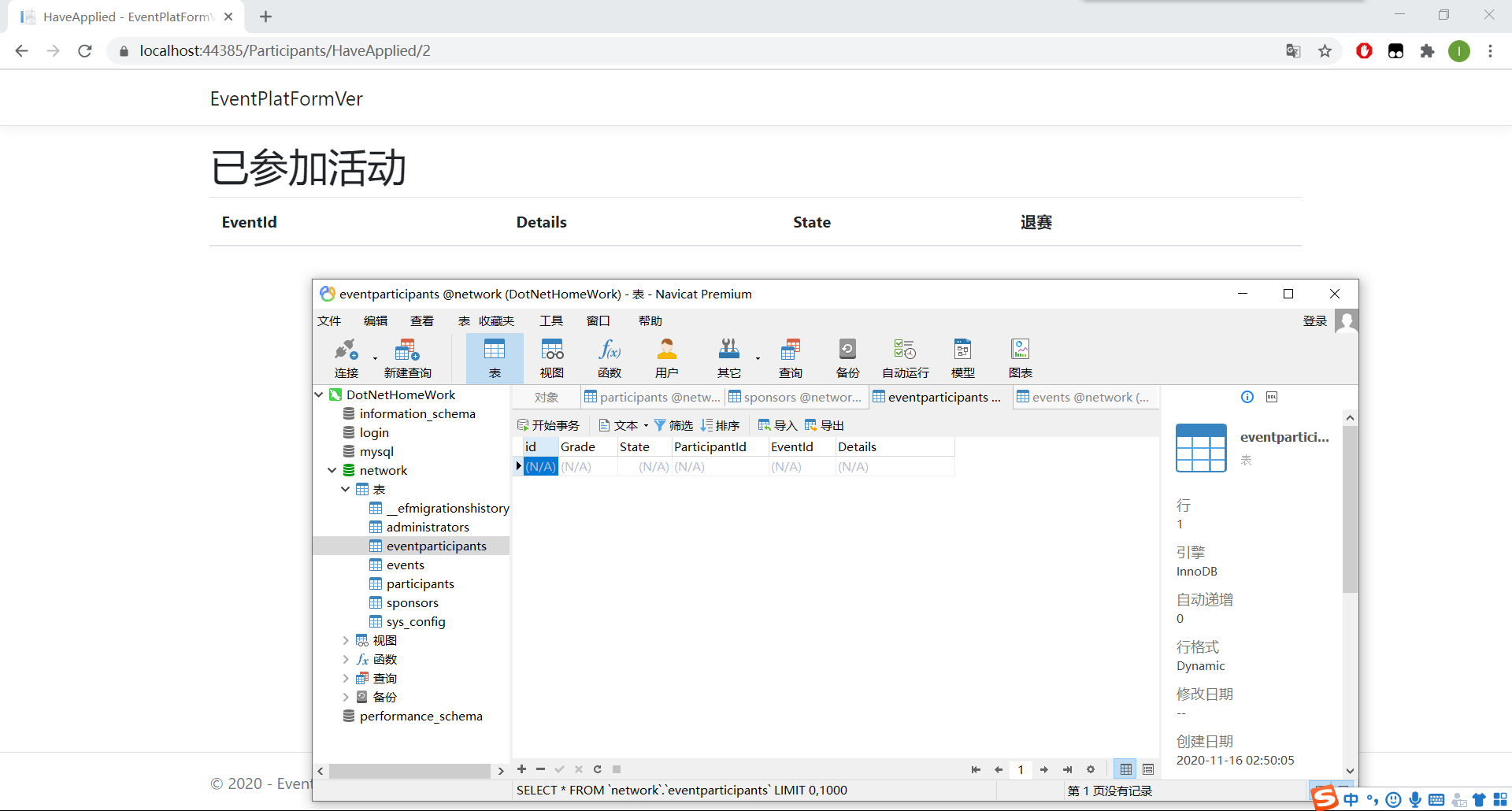


图3.8 点击退赛后的已参加活动界面

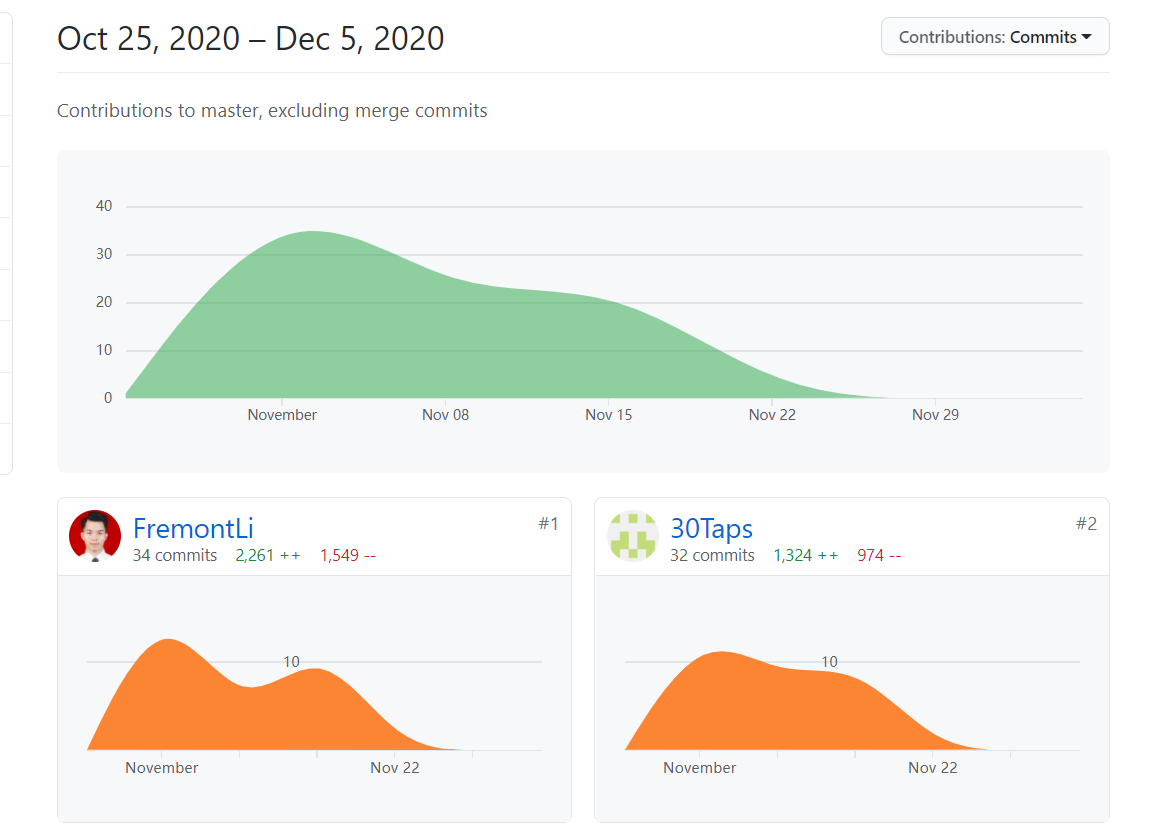
# 4 小组自我评价

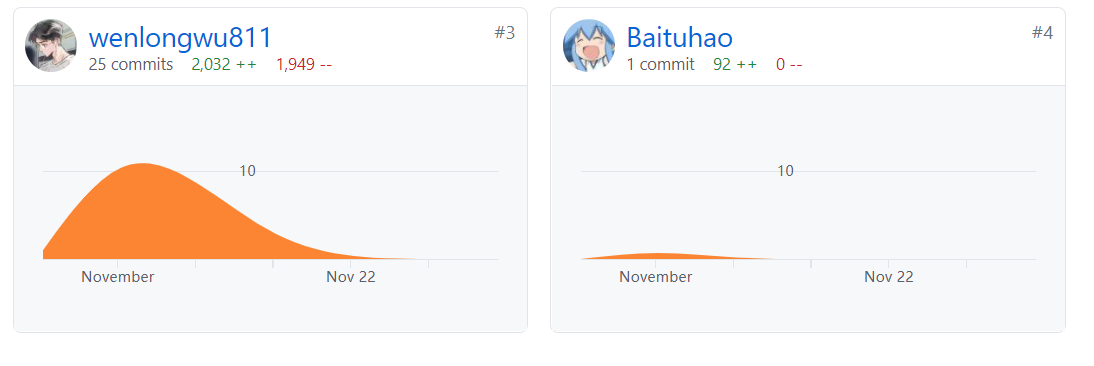
## 4.1 难度

遇到的问题，由于我们的前端开发水平一般，临时学 临时用，前端的页面做的比较简洁（也就是比较丑，基本上就是没有前端！）在功能上面呢，做的也不是很完善，选题有点大，一个人做也比较吃力，所以系统只能跑出一个完整的闭环：从活动的申请到活动审核、发布活动、查看活动信息、报名参加活动、退赛，延期举行、给分。这些基本功能 虽然实现了，但是代码的鲁棒性很差，几乎没有异常处理和权限认证。由于对框架不够熟悉，为了完成功能前后端也写了很多多余的代码。但是我后期会把这个东西给解决掉，希望最终能够部署到学校的服务器上。

## 4.2 代码量

个人贡献（这里关于孙元锴的部分显示并不正确，它不是单纯在自己的分支里提交，期间他是负责帮其他的队友写了不少，在个人报告里会详述。）





## 4.3 亮点

1. 覆盖了课本中大部分知识点，深刻巩固了课程学习内容。

以下为目录知识点Check表（我们跳过了一些目录中的简介小节）：

表**3.1** 知识点Check表

|  |  |  |  |
| --- | --- | --- | --- |
| 章节 | 知识点 | 是否使用 | 备注 |
| 第一章 | 程序中的输入输出及运算 | √ |  |
| 开发工具 | √ | VS和VSCode， |
| 应用程序类型 | √ | Web |
| 面向对象程序设计 | √ |  |
| 第二章 | 数据类型、变量与常量 | √ |  |
| 运算符与表达式 | √ |  |
| 流程控制语句 | √ |  |
| 数组 | √ |  |
| 第三章 | 类、字段、方法 | √ |  |
| 属性、索引器 | √ |  |
| 类的继承 | √ |  |
| 修饰符 | √ |  |
| 接口 | √ |  |
| 结构、枚举 | √ |  |
| 第四章 | 泛型 | √ |  |
| 委托及Lambda表达式 | √ |  |
| 事件 | √ |  |
| 异常处理 | √ |  |
| 命名空间、嵌套类型、程序集 | √ |  |
| 其他成分 | × |  |
| 第五章 | C#语言基础类 | √ |  |
| 字符串 | √ |  |
| 集合类 | √ |  |
| 排序与查找 | × |  |
| Ling | × | 我们使用了更高级的EF框架以及MVC |
| 第六章 | 流及二进制输入/输出 | √ |  |
| 文本输入/输出 | √ |  |
| 文件、目录、注册表 | √ |  |
| 环境参数及事件日志 | √ |  |
| 程序的调试、追踪与测试 | √ |  |
| 第七章 | Windows窗体 | √ |  |
| 常用控件 | √ |  |
| 一些容器类控件 | √ |  |
| 窗体及对话框 | √ |  |
| MDI窗体、菜单、工具栏 | × |  |
| 第八章 | / | × | 这章为绘图及图像，我们主要为应用软件类型的程序，故不需要 |
| 第九章 | 文本及正则表达式 | √ |  |
| XML编程 | √ |  |
| 网络信息获取及编程 | √ |  |
| 几类不同网络信息的处理 | √ |  |
| 第十章 | 线程基础 | √ |  |
| 线程池与计时器 | × |  |
| 集合与Windows程序中的线程 | × |  |
| 并行编程 | × |  |
| 异步编程 | √ |  |
| 第十一章 | ADO.NET数据库编程 | √ |  |
| 使用高级数据工具 | √ | MVC框架，postman以及navicat |
| 网路通信编程 | × |  |
| 互操作与多媒体编程 | × |  |
| 第十二章 | / | / | 课程未作要求 |
| 总计 | / | 80.0% | 知识点覆盖率 |

1. 对书本提到的的前沿技术进行了深入的研究，如EF框架，WPF，WebApi、注册表等。

针对第一次进度汇报中老师提出的质疑进行了针对性的反思，小组决定深入学习

.NET高级特性和前沿技术。

## 4.4 收获

在本门课程的学习中，小组成员认真学习了C#基础语法和.NET中的高级特性。虽然由于疫情在家同学们都是线上交流，但是小组合作的默契并没有因此而降低。本小组为了提高大家合作之间的默契从而顺利地完成最终的项目，每次老师布置的课后作业大家都会在小组群发表自己的意见和想法，互相讨论共同探讨需要注意的细节和难点。在每次的小作业中不断地积累经验，掌握课本知识点，最后将每一个知识点都用在了我们的项目中。

为了提高组员之间合作的效率，李政民同学提前学习了Github网站的使用方法并撰写了一份Github使用指南共享到了小组群，通过线下研修室讨论手把手带领组员掌握了Github网站的使用，小组共享学习的氛围因此而愈来愈浓。组长吴文龙同学积极展开小组的督促学习，小组项目进度进展顺利。

由于我们的项目选用了MVC框架，在第一次向老师汇报小组大作业进度后吸收了老师的意见，增加了Service层以实现前后端更好地分离。为了加强对课本知识的巩固和应用，我们增加了很多应用部分，在这个过程中我们几乎把课本中的知识点都囊括了进来，从而对这学期所教学的内容都有了更深刻的掌握。

此外，在对课本知识进行了全方位地巩固的同时，小组成员也积极探索，通过自学官方文档的方法学习了很多比较前沿但是课本中介绍的不足的技术，比如EF框架、WPF技术和WebApi技术等等。在自学的过程中，组员遇到问题都会主动在小组群提出，知道问题所在的组员就会倾囊而出积极解答，在不同的分工中又做到团队合作以致共赢。

通过两三个月以来密切的交流合作，小组成员也培养了自学能力，学会了如何与同伴沟通合作，为以后的实习和工作打下了坚实的基础。

# 参考文献

[1] 微软WPF参考文档https://docs.microsoft.com/en-us/dotnet/

framework/wpf/

[2] Jonathan Katz，Yehuda Lindell. Introduction to Modern Cryptography, Second Edition [M]. Chapman and Hall/CRC，2014

[3] 潘承洞，潘承彪. 初等数论[M]. 北京大学出版社，2013

教师评语评分

评语：

评分：

评阅人：

年 月 日

（备注：对该实验报告给予优点和不足的评价，并给出百分之评分。）