

Python 编程项目报告与系统使用介绍

基于 Python-Socket 库实现的 2022 年卡塔尔世界杯赛事管理系统



兰州大学信息科学与工程学院

2021 级计算机科学与技术（数据科学方向）一班

姓名： 周忠泉

学号： 320210941131

邮箱： zhouzhq2021@lzu.edu.cn

日期： 2023/1/15

目录

一、引言	3
1. 编写背景	3
2. 编写目的	3
二、系统概述	3
1. 系统简介	3
1.1. 运行环境	3
1.2. 系统文件说明	4
1.3. 运行步骤	4
2. 系统界面设计	5
3. 系统功能演示	6
3.1. 用户登陆与注册功能	6
3.2. 普通用户赛事查询功能	7
3.3. 管理员用户赛事查询功能	9
3.4. 管理员用户赛事管理功能	10
4. 系统数据库设计	13
4.1. 赛事管理数据库设计	13
4.2. 用户管理数据库设计	14
三、编程思路与实现	14
1. 客户端	14
2. 服务器端	16
3. 赛事数据库构建程序	18
四、项目亮点与特色	18
1. 编程方面	18
2. 需求实现方面	18
3. 系统的二次开发	19
五、总结	19

一、引言

1. 编写背景

国际足联世界杯(FIFA World Cup)，简称“世界杯”，是一项象征着足球世界的最高荣誉的足球赛事，每四年举办一次具有极大的知名度和影响力，有来自世界各地的国家级团队参与。

2022 年卡塔尔世界杯（FIFA World Cup Qatar 2022）是第 22 届国际足联世界杯，该届赛事于 2022 年 11 月 20 日至 12 月 18 日在卡塔尔境内 8 座球场举行，是历史上首次在卡塔尔和中东国家境内举行、也是第二次在亚洲举行的世界杯足球赛，卡塔尔世界杯还是首次在北半球冬季举行、首次由从未进过世界杯决赛圈的国家举办的世界杯足球赛。该届赛事有 32 支球队参加，除东道主卡塔尔本国足协代表队外，另有来自五个大洲足球联合会的 31 支球队拥有该届世界杯决赛圈参赛资格，各大洲足联各自举办预选赛以决定最终出线的球队。

2. 编写目的

本次卡塔尔世界杯包括众多赛事信息，例如：赛程安排、国家队名称、比赛时间、比分、加时罚时等等，为了实现对这些赛事信息的有效管理以及对用户查询赛事信息的简洁展示，我根据任务要求，设计一款基于 Python-Socket 库的卡塔尔赛事查询系统，该系统具有客户端和服务端，并采用 Sqlite3 模块设计的数据库进行数据管理。

二、系统概述

1. 系统简介

2022 卡塔尔世界杯赛事管理系统可以接受客户端用户输入的信息，并传输给服务器端，服务器端连接有数据库，查询信息在服务器端处理后将查询结果返回给客户端。系统实现了普通用户的登陆与注册功能、管理员用户登陆与赛事管理功能、根据用户输入的关键信息进行查找功能等，同时使用了 GUI 界面对系统操作进行了可视化以及查询结果的简洁展示。

系统整体设计时语言采用了全英文，即交互界面和提示信息为英文，但是由于不同计算机中系统语言的不同，tkinter 设计出来的提示窗按钮可能弹出中文，这并不影响系统的使用。

1.1. 运行环境

系统基于 Python 3.9 版本进行了设计，.py 文件中调用了一系列 Python 内置库：socket 模块、tkinter 模块、re 模块、json 模块、sys 模块、sqlite3 模块，以及外置的 Pandas 数据分析包，Pandas 包在使用前可以运行下面的命令进行安装（建议同时安装 NumPy 库）：

1. pip install numpy
2. pip install pandas

其余内置库如果 Python 环境中缺少也可以在终端中使用 pip 命令进行安装。

1.2. 系统文件说明

初始状态系统文件夹中含有一下五个文件，分别是存储世界杯的赛事数据的 CSV 文件、系统 icon 图标文件、系统客户端.py 文件、系统服务器端.py 文件、赛事数据库建立.py 文件。在运行完赛事数据库建立.py 文件会生成一个存储有赛事数据的.db 文件，在首次运行服务器端后，还会生成一个用户数据管理的.db 文件。






	2022 FIFA World Cup Tournament Da...	2022-12-30 20:30	Microsoft Excel ...	4 KB
	FIFA_icon	2023-01-12 16:18	PNG 文件	12 KB
	System_Client	2023-01-12 12:29	JetBrains PyCharm	37 KB
	System_Server	2023-01-12 12:31	JetBrains PyCharm	10 KB
	Tournamentdb_Establish	2023-01-12 12:32	JetBrains PyCharm	1 KB

图 1. 初始状态系统文件

1.3. 运行步骤

首先要注意系统文件必须放在同一个文件夹里并且运行环境要满足程序需求，下面我们打开 Tournamentdb_Establish.py 文件并运行，该程序会生成一个 2022_FIFA_WorldCup.db 文件，注意运行一次后不需要重新运行。

然后打开 System_Client.py 文件与 System_Server.py 文件，先运行 System_Client.py 文件提起服务器，然后运行 System_Server.py 文件输入 GUI 操作界面。这两个文件运行后会生成一个 System_users.db 文件，该数据库文件中提前存入了一个 ID 为 A320210941131 (A + 本人学号)，密码为 admin123 的管理员用户，用于后续系统演示。系统设计时的 Socket 模块默认与本地 (localhost) 建立通讯连接。

	2022 FIFA World Cup Tournament Da...	2022-12-30 20:30	Microsoft Excel ...	4 KB
	2022_FIFA_WorldCup	2023-01-12 19:28	Data Base File	8 KB
	FIFA_icon	2023-01-12 16:18	PNG 文件	12 KB
	System_Client	2023-01-12 12:29	JetBrains PyCharm	37 KB
	System_Server	2023-01-12 12:31	JetBrains PyCharm	10 KB
	System_users	2023-01-12 19:31	Data Base File	12 KB
	Tournamentdb_Establish	2023-01-12 12:32	JetBrains PyCharm	1 KB

图 2. 运行后完整的系统文件

至此系统开始运行，可以进行继续操作，如果系统被关闭，还可以继续重复运行服务器端再运行客户端的步骤重新启动系统。如果在修改数据库中的数据后想要初始化数据库（恢复至初始状态数据库中的数据），只需要在文件夹中删除这两个数据库然后重复最开始的步骤即可。

2. 系统界面设计

系统的交互界面全程使用GUI界面进行展示，系统运行起来后首先是用户登陆界面，界面中有操作提示信息与信息输入框，同时为了美观我也修改了tkinter的icon图片以贴合现实。

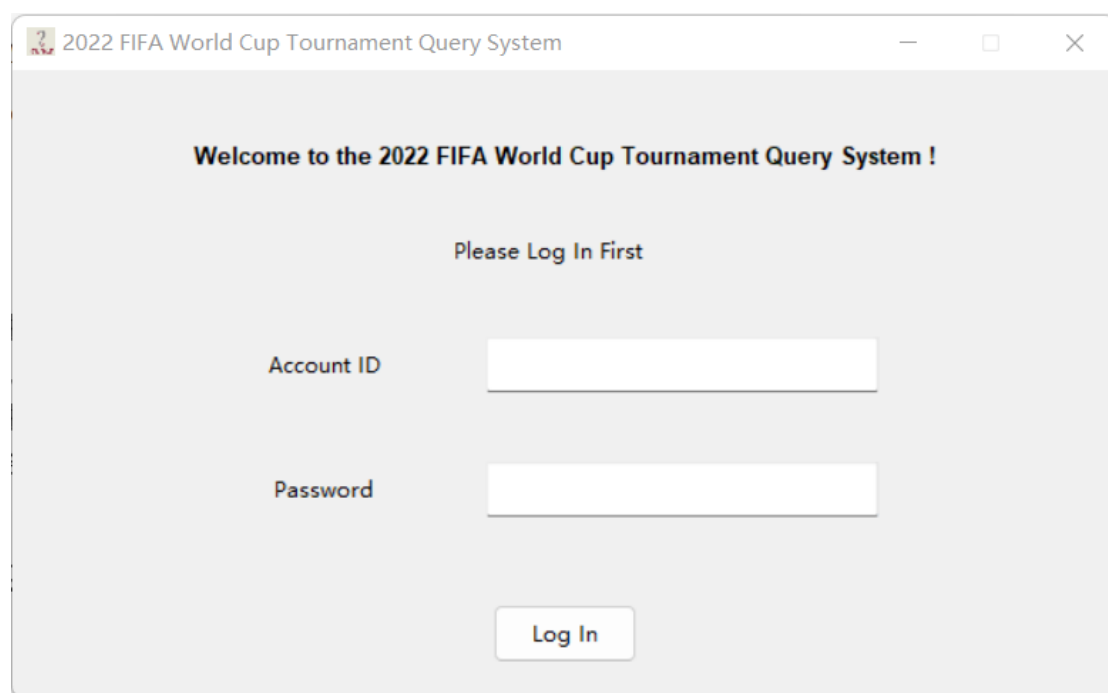


图 3. 系统登陆界面



图 4. FIFA_icon.png 图标文件

其余系统界面设计将在下面系统功能介绍时进行展示。

3. 系统功能演示

3.1. 用户登陆与注册功能

首先，用户在输入框中输入信息后点击“Log In”按钮即可以登陆，如果是新用户首次登陆（用户数据库中没有检索到用户信息）则系统会提示用户进行注册，在初始状态下，使用该系统均需要先注册，可以先输入自己设计的 ID 与密码，然后点击登陆就好弹出注册提示信息：

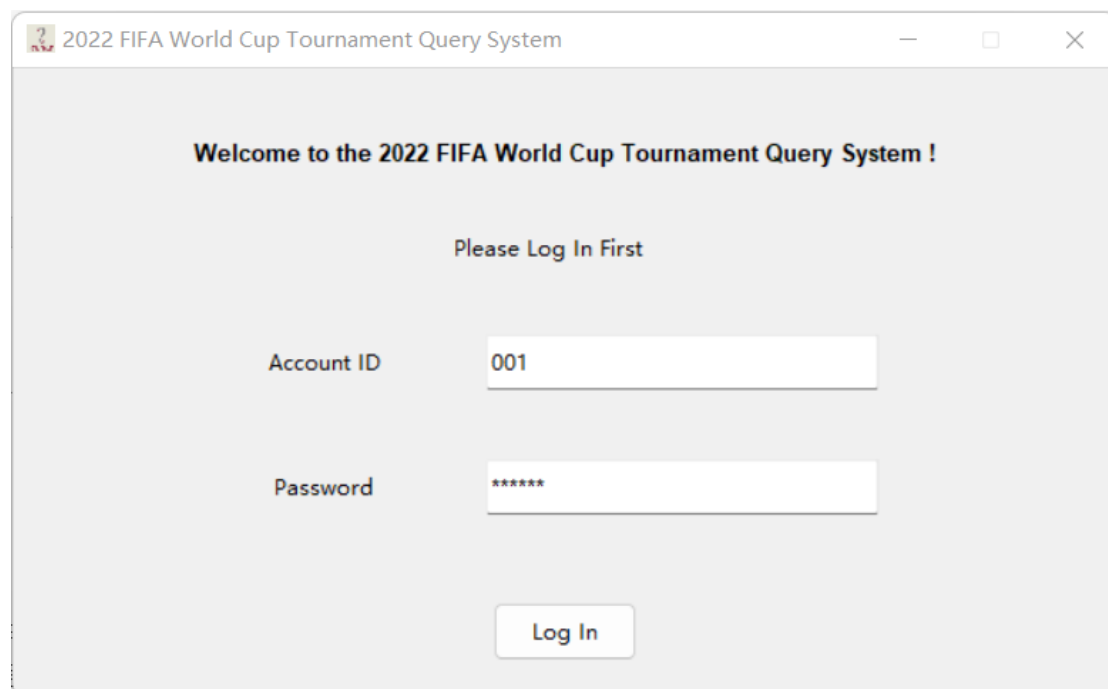


图 5. 登陆界面

在这里我输入一个新用户 001，密码为 123456，系统会提示您为新用户，引导用户进行注册（这里按钮为中文是因为 tkinter 识别计算机系统语言为中文，其实在系统设计时为全英文，这并不影响系统的正常使用）。

注册时需要注意密码长度必须大于等于 6，系统对密码长度小于 6 的输入会进行提示，并且注册时两次输入的密码要相等，否则也会进行报错提示。

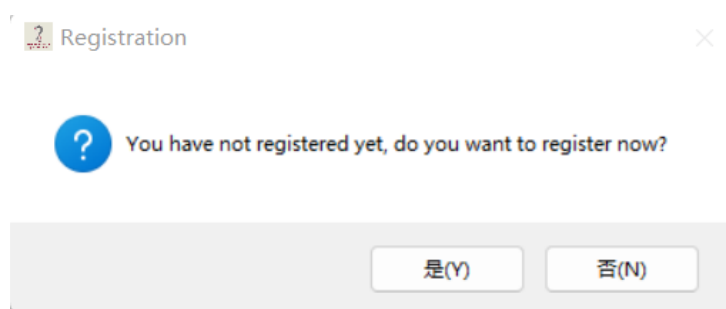


图 6. 注册引导

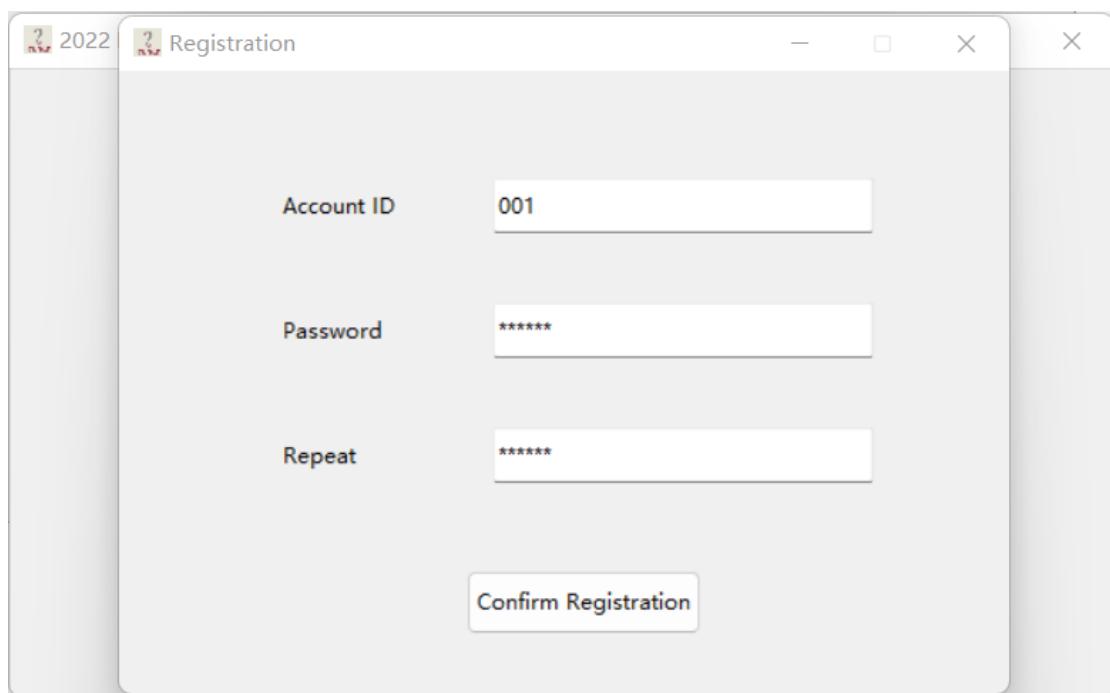


图 7. 进行注册

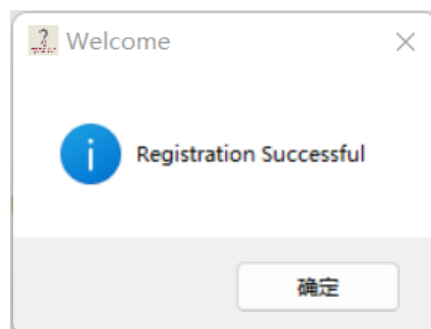


图 8. 注册成功

注册成功后会回到登陆界面，我们重新登陆即可。

3.2. 普通用户赛事查询功能

登陆成功后会进入普通用户的查询界面，界面上有信息提示，提示用户如何进行操作。并且还有“展示全部比赛”按钮可以直接查看所有比赛以及系统退出按钮。

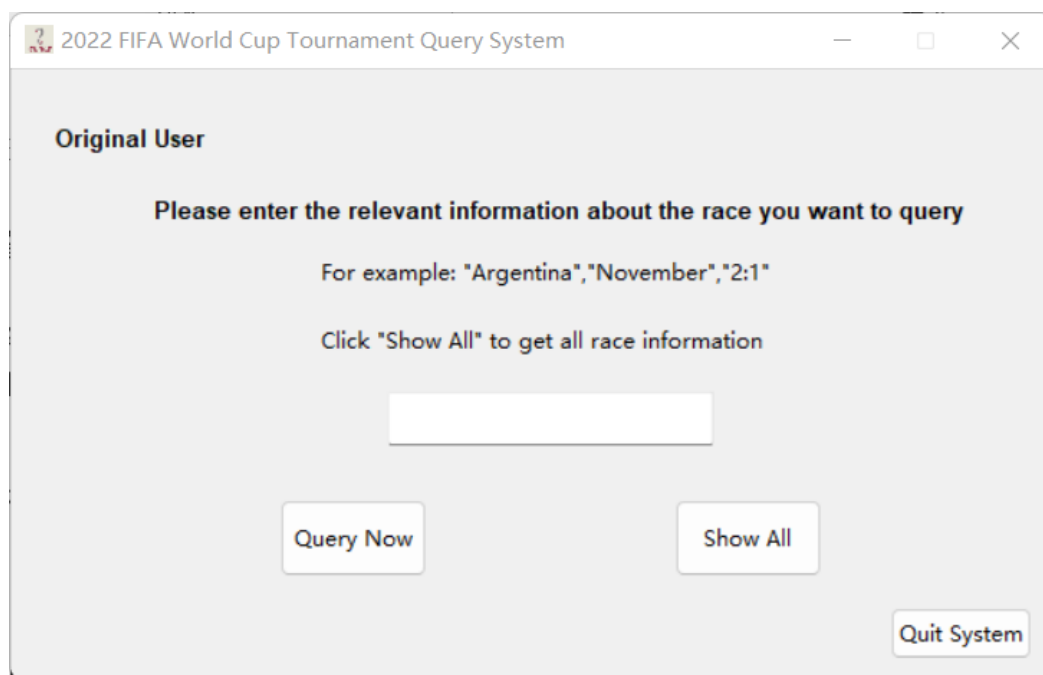


图 9. 普通用户查询界面

3.2.1. 根据关键词进行查找

赛事数据库中存储了完整 2022 年卡塔尔世界杯的 64 场赛事数据(比赛均已完结), 具体数据库设计思路会在下文进行详细介绍。

用户可以输入提示信息中的关键词, 例如: “Argentina”、“November”、“2:1” 或者具体的日期天数比如 27 等, 输入信息后点击 “Query Now” 按钮就可以查看相应信息。这里支持的查询关键词有: 月份、队名、比分、日期天数, 基本能满足用户的所有查询需求。这里以 “Argentina” 为例进行展示:

ID	Date	Time	Team1	Points	Team2	Status	Penalty Kick
13	November 22	18:00	Argentina	1:2	Saudi Arabia	Ended	—
16	November 27	3:00	Argentina	2:0	Mexico	Ended	—
17	December 1	3:00	Poland	0:2	Argentina	Ended	—
50	December 4	3:00	Argentina	2:1	Australia	Ended	—
57	December 10	3:00	Netherlands	2:2	Argentina	Ended	3:4
61	December 14	3:00	Argentina	3:0	Croatia	Ended	—
64	December 18	23:00	Argentina	3:3	France	Ended	4:2

图 10. 查询结果展示

赛事信息以表格的形式进行了展示，除任务要求的几项信息进行展示以外还加入了罚球数（Penalty Kick），使赛事信息更加详细。用户可以多次在查询界面进行查询。

3.2.2. 实现模糊查找

在系统设计时考虑到了用户对模糊搜索的需要，在信息输入框中不需要输入完整的队名信息或者月份信息，比如只输入“Argen”或“Nove”。并且不严格要求首字母大写，系统会对这些信息进行处理，这样可以为用户的查找提供便利。当然对于具体天数和比分要求进行完整的输入。

3.3. 管理员用户赛事查询功能

在系统设计时，默认客户端不能注册管理员账户，管理员账户只能通过提前存入或者由系统设计人员自行添加，客户端只能进行普通用户的注册，这样可以保证系统的安全性。在这里我们向数据库中提前存入了一个管理员账户（ID:A320210941131, 密码：admin123）进行相关系统功能的演示。

同样通过上述的登陆方法进行登陆，这样我就会来到管理员交互界面，可以看到，管理员用户在登陆后主页面会自动显示赛事数据库中的所有赛事信息（64 场比赛），包括赛事的 ID、日期、时间、队名、比分等，这样设计的目的是方便管理员用户对赛事数据的管理。

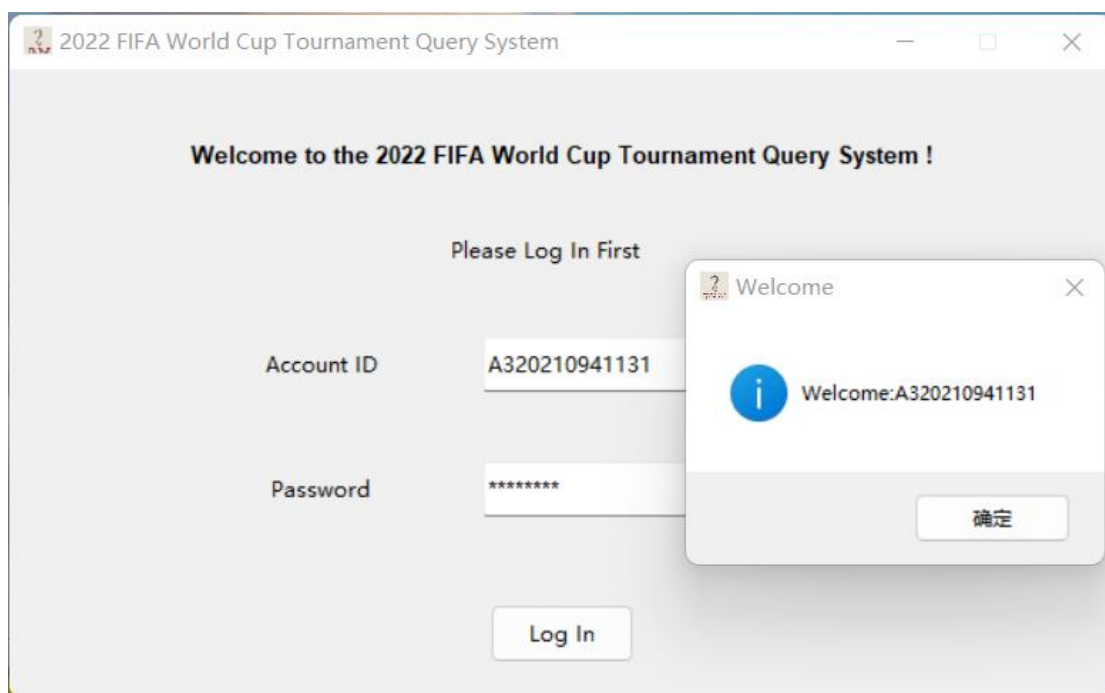


图 11. 管理员用户登陆

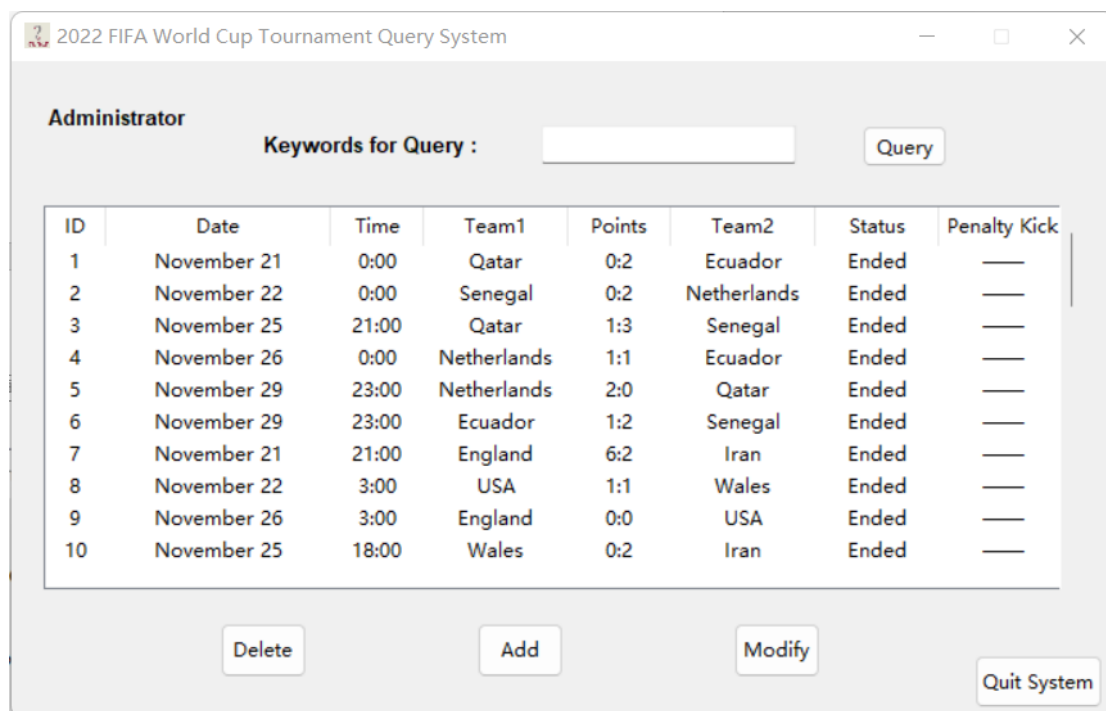


图 12. 管理员用户交互界面

在管理员用户交互界面，设有 5 个按钮，作用分别为查询、添加赛事、删除赛事、修改赛事以及退出系统，在赛事展示表格中还设计了滚动条使界面更加美观。在这里的查询输入框中输入的关键词要求与普通用户查询功能一致，支持的查询关键词有：月份、队名、比分、日期天数。

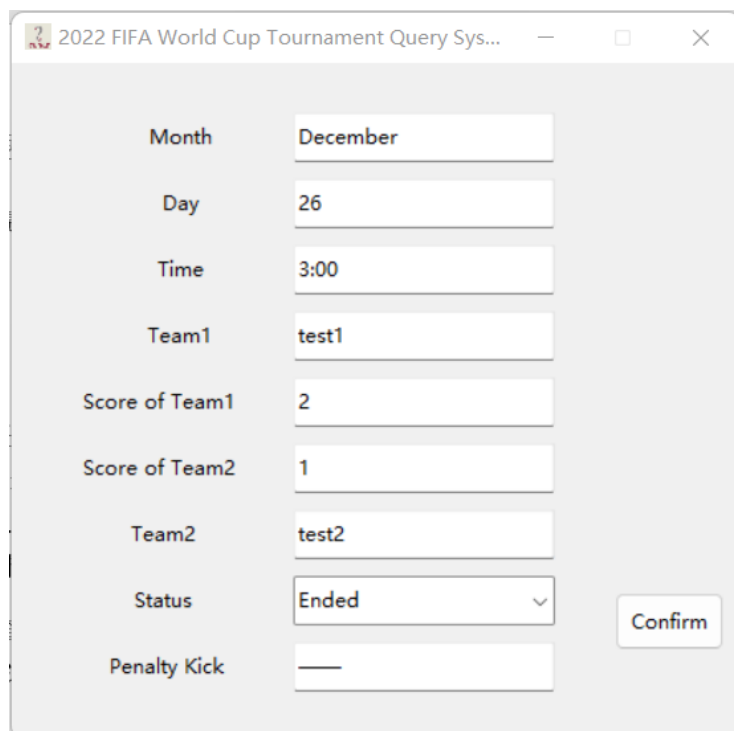
3.4. 管理员用户赛事管理功能

总共分为三类：赛事添加，即对世界杯中额外的赛事（假设的赛事）进行补录；赛事删除，即对存在某些问题的比赛进行直接删除；赛事修改，即对比赛中某些信息进行修改，如修改比分、修改比赛时间等。

3.4.1. 赛事添加

赛事添加对应系统的“Add”按钮，点击后会弹出一个新的信息输入界面，在这个界面中依次输入月份、日期、时间等信息后点击“Confirm”按钮确认提交。其中新添加的这场比赛 ID 由服务器端自动处理生成，一般按照录入次序，ID 依次递增，“Status”框在界面设计时设计为列表框，可以下拉列表选择两种状态：完结与未开赛，方便管理员进行添加。

添加成功后系统会提示：重启系统后可以直接查看到新添加的赛事信息。或者可以根据关键词进行查询，在不重启系统的情况下也可以看到新的比赛信息。



The screenshot shows a window titled "2022 FIFA World Cup Tournament Query Sys...". Inside, there is a form with the following fields and values:

Field	Value
Month	December
Day	26
Time	3:00
Team1	test1
Score of Team1	2
Score of Team2	1
Team2	test2
Status	Ended
Penalty Kick	—

A "Confirm" button is located at the bottom right of the form.

图 13. 赛事添加界面

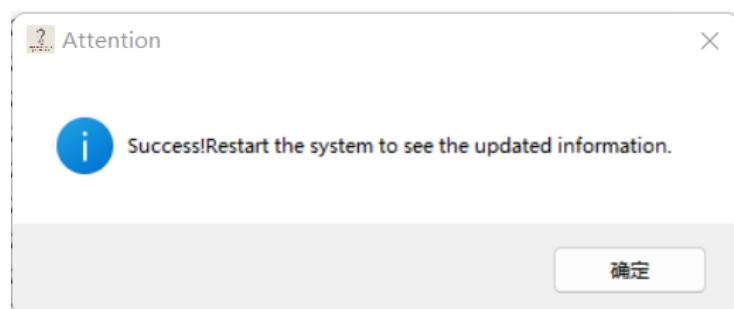


图 14. 操作成功提示

3.4.2. 赛事删除

赛事删除的原理是根据管理员用户提供的比赛 ID, 在数据库中删除与这条 ID 对应比赛的比赛信息, 需要删除的比赛对应的 ID 可以直接在主界面进行查看, 在输入框中输入 ID 后点击确认按钮即可删除, 删除成功后会有系统提示。

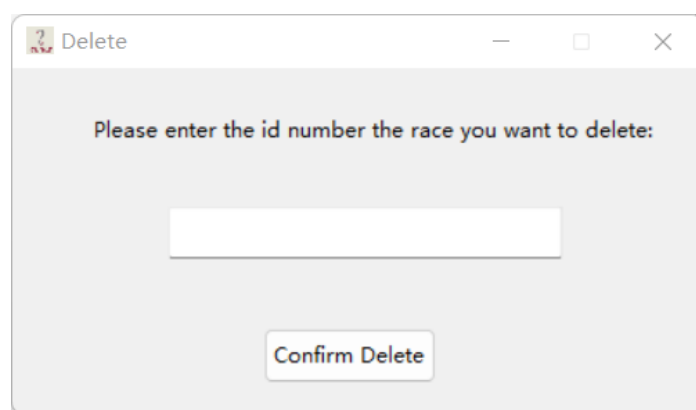


图 15. 赛事删除界面

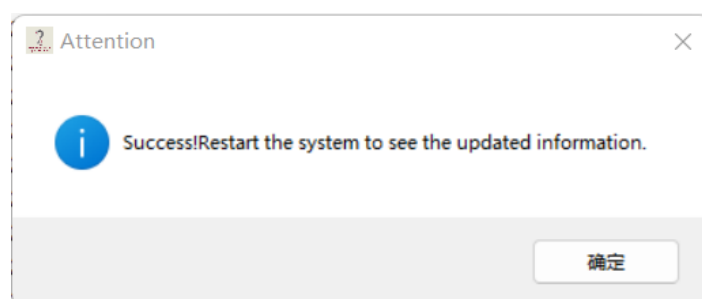
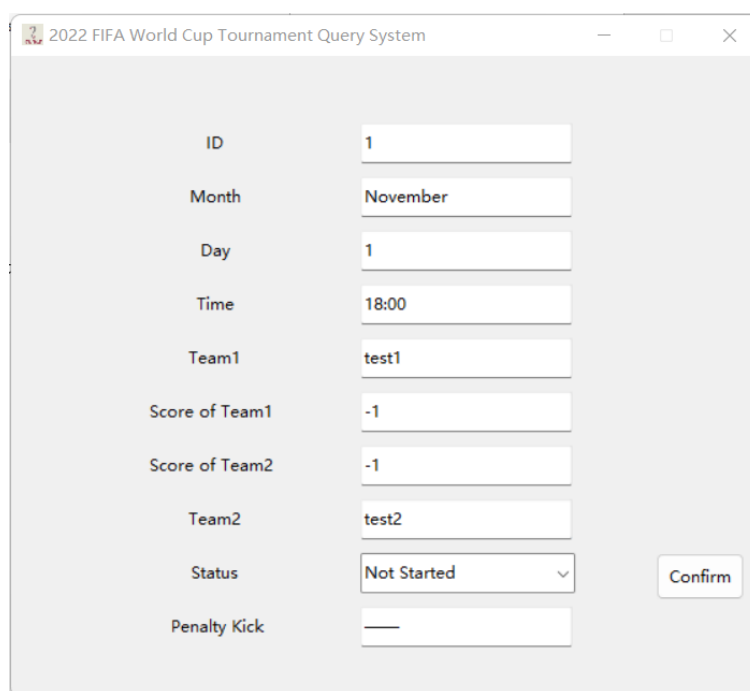


图 16. 操作成功提示

3.4.3. 赛事修改

赛事修改的原理是获取管理员用户输入的比赛 ID，修改对应 ID 的比赛信息，需要修改的信息由管理员用户进行直接输入。在这里我以 ID 为 1 的比赛进行演示，修改其比赛时间、队名、比赛状态。

在这里由于我将比赛状态设置为未开赛，所以对应的比分我需要设为-1，代表没有比分，这样能够保证数据库的数据一致性，方便对比赛数据进行管理，否则数据库系统将由于无法处理非数字型的比分从而出现报错。



ID	1
Month	November
Day	1
Time	18:00
Team1	test1
Score of Team1	-1
Score of Team2	-1
Team2	test2
Status	Not Started
Penalty Kick	—

Confirm

图 17. 赛事修改界面

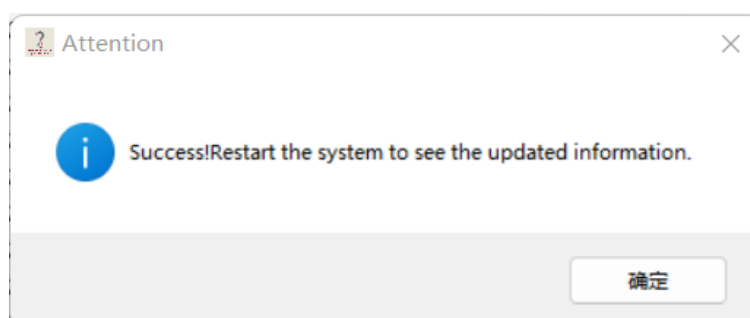


图 18. 操作成功提示

4. 系统数据库设计

4.1. 赛事管理数据库设计

首先赛事数据由本人通过世界杯官网手动收集整理而成，64 场比赛数据为真实的完结的 2022 卡塔尔世界杯赛事数据，将这些数据先存入了 CSV 文件中，然后再通过程序（Tournamentdb_Establish.py 文件）统一导入数据库，选择将所有数据都进行录入的原因是为了让系统更加贴合现实，让有相关需求的用户可以直接查询对应关键信息的比赛数据。

其次就是数据库表属性的设置，下面用表格的形式进行展示，其中“Penalty Kick”为罚球数，未开赛的赛事中比分为需设为-1：

ID	Month	Day	Time	Team1
INTEGER PRIMARY KEY	Varchar(15)	INTEGER	Varchar(15)	Varchar(50)
Score1	Score2	Team2	Status	Penalty Kick
INTEGER	INTEGER	Varchar(50)	Varchar(15)	Varchar(15)

4.2. 用户管理数据库设计

首先我们给出用户数据库的表设计，用表格进行属性展示：

id	password	is_admin
Varchar(25) UNIQUE NOT NULL	Varchar(255) NOT NULL	Boolean NOT NULL

其中 id 和 password 为用户注册时输入的信息，在 is_admin 属性中，普通用户和新注册用户为 0，管理员用户为 1，用于标识普通用户和管理员用户，在用户登陆时根据这个标识跳转到不同的交互界面。

用户数据库对应的 System_users.db 文件在服务器端首次运行时自动生成，并在表中导入一位管理员，初始状态下该数据库中仅有 1 名用户，初次使用该系统均需要先注册再登陆。

三、编程思路与实现

该系统主要程序包括以下三个方面：客户端、服务器端、数据库构建程序。具体的代码实现本人在 .py 文件中做了较为详细的代码注释，注释大部分为英文，重要函数部分做了中文注释，代码结构清晰，方便阅读。下面主要介绍思路与层次结构。

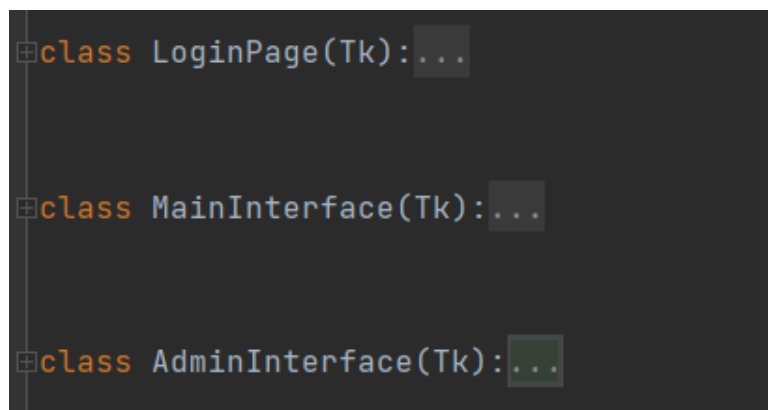
1. 客户端

客户端主要实现了两大类功能：使用 socket 模块与服务器端进行本地连接、使用 tkinter 模块实现 GUI 界面交互，首先介绍一下客户端中 import 的不同模块的作用：

```
1. from tkinter import *
2. from tkinter.ttk import *
3. import tkinter.messagebox
```

```
4. import socket
5. import json
6. import sys
7. import re
```

其中 `tkinter` 相关组件用于实现可视化，客户端中总共设计了 3 个主界面，分别是用户登陆主界面、普通用户交互界面、管理员用户交互界面，这三个界面的实现对应下图中的三个继承自 `Tk` 的 `Class` 类，在这三个类中通过编写 `tkinter` 组件函数，比如按钮、输入框、标签等实现界面的可视化：



```
class LoginPage(Tk):...
class MainInterface(Tk):...
class AdminInterface(Tk):...
```

图 19. 客户端主要的三个类

三个主界面的调用层次为：首先运行 `LoginPage` 类生成登陆界面，登陆成功后根据用户是否为管理员，分别调起 `MainInterface` 类（普通用户交互界面）和 `AdminInterface` 类（管理员用户交互界面），每个类中又有着自己的函数方法实现不同的功能。

`Socket` 模块用于和服务端建立连接，客户端向服务端发送消息，服务端返回一个响应。使用 `socket.AF_INET` 和 `socket.SOCK_STREAM` 常量来创建一个 `TCP socket`。接下来，我们绑定 `socket` 到本地地址和端口，并使用 `listen()` 方法开始监听连接请求。然后使用 `accept()` 方法可以用来接受来自客户端的连接请求，这个方法会返回一个新的 `socket` 和客户端的地址信息。`with` 语句来自动处理 `socket` 的打开和关闭。`send()` 方法向服务端发送数据，`recv()` 方法接收服务端的响应。

其次是 `json` 模块和 `re` 模块，`json` 模块用于对列表进行字符串化，结合 `encode()` 方法和 `decode()` 方法使我们可以与服务器进行 `list` 列表的传输。`re` 模块用于对用户输入的查询信息进行正则匹配，判断该信息输入月份、日期、队名等中的哪一类。

最后是 `sys` 模块，该模块中的方法 `sys.exit()` 用于正常退出系统。

接下来介绍一下客户端代码的函数功能与主体函数的层次问题，我在这里使用树图的方式对代码结构进行一下梳理（这里只梳理了重要的类和函数体，更细节的代码可以在程序中直接查看）：

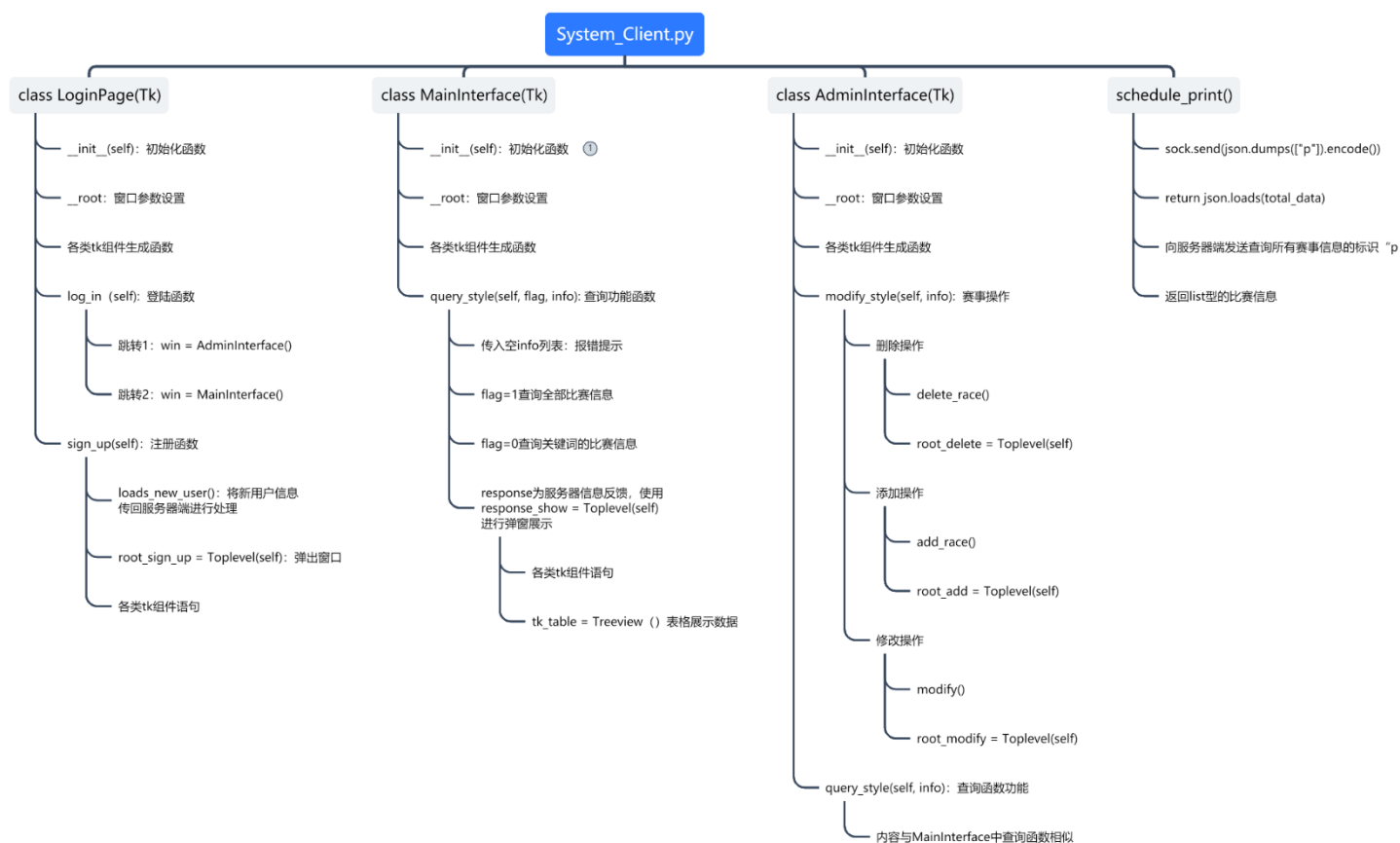


图 20. 客户端代码结构梳理

在这里客户端运行后首先与服务器端建立连接，成功建立连接后会打印成功信息，然后开始调用 `LogInPage` 的主体进行 `mainloop()`，然后根据上面客户端代码结构图进行调用运行，其中包含与服务器端数据的传输与 `tkinter` 模块的可视化语句。并且我也实用了 `try-except` 语句对客户端可能出现的错误进行处理，这样就完成了整个客户端的设计与实现。

2. 服务器端

服务器端主要实现了三大类功能：与客户端之间的数据传输、正确处理客户端的数据并返回结果、对数据库进行数据管理，下面我首先介绍一下 `import` 的模块的作用：

```

1. import socket
2. import sqlite3
3. import json
4. import sys
5. import re
  
```


Python 的 `sqlite3` 库可以很方便地创建和操作 SQLite 数据库。它可以用于创建数据库，与数据库进行交互以及存储数据。可以将 SQL 语句内置到函数中，根据用户的不同需求执行不同的函数，调用不同的 SQL 语句并执行，从而实现对数据的简单管理，并将结果返回给客户端进行显示。

Socket 模块用于和服务端建立连接，客户端向服务端发送消息，服务端返回一个响应。使用 `socket.AF_INET` 和 `socket.SOCK_STREAM` 常量来创建一个 TCP socket。`send()` 方法向客户端发送数据，`recv()` 方法接收客户端的数据。与客户端相同，`json` 模块用于对列表进行字符串化，结合 `encode()` 方法和 `decode()` 方法使我们可以与服务端进行 list 列表的传输。`re` 模块用于对用户输入的查询信息进行正则匹配，判断该信息输入月份、日期、队名等中的哪一类，然后针对不同关键词使用不同的 SQL 语句对数据库进行查询，将查询结果在通过 `socket` 返回给客户端。

接下来我同样使用树图的方式对服务器端的函数层次以及代码结构进行介绍：



图 21. 服务器端代码结构梳理

服务器端主要作用为接受客户端消息，根据信息调用不同函数，完成数据处理，然后返回处理结果，相对于客户端代码量较小，结构也更简单一些。

3. 赛事数据库构建程序

该程序的思路较为简单，首先导入 pandas 包和 sqlite3 模块，读取整理好的存储有赛事信息的 CSV 文件，将其中的数据使用循环语句直接导入建立好的数据库，这样就生成了一个初始状态的数据库，该数据库中在初始状态存有 64 场完整的世界杯比赛数据。

该程序运行一次后不需要多次运行，当数据库丢失或者需要初始化数据时再重新运行即可，下面直接附上关键部分的代码：

```
1. import pandas as pd
2. import sqlite3
3.
4. df = pd.read_csv("2022 FIFA World Cup Tournament Data.csv")
5.
6. conn = sqlite3.connect("2022_FIFA_WorldCup.db")

1. for i in range(len(df.index)):
2.     sql_text = "INSERT INTO Tournaments VALUES "
3.     cur.execute(sql_text + str(tuple(df.values[i])))
4.     conn.commit()
```

四、项目亮点与特色

1. 编程方面

- 系统由客户端和服务端之分，客户端与服务端通过 socket 模块进行通讯，客户端不能直接调用服务端的功能，使用户不能直接访问赛事数据库，保障了系统数据的安全性。
- 客户端采用 GUI 界面进行交互，界面设计完整，操作简单，满足了本次任务的所有需求，相较于命令行交互更简洁、更贴合实际。
- 客户端与服务端尽可能使用 Python 内置库，不需要额外配置环境。并且代码结构清晰，容错性高，稳健性好，对于可能出现的错误操作可以进行处理并提示。
- 赛事数据库的设计合理，赛事数据符合实际要求。

2. 需求实现方面

- 实现了系统对不同用户的管理功能，区分普通用户与管理员用户，对两类用户提供不同的交互界面，对管理员用户提供更多的功能。
- 满足用户对查询赛事的需求，实现了关键词的模糊查找，支持多种类型的关键词，

比如月份、日期、队名、比分。并且采用简洁的表格视图对数据进行展示，方便用户阅读。

- 满足管理员用户对赛事的添加、删除和修改的功能。
- 在常规的比赛信息基础上，额外加入了一项“罚球数”（即点球大战中的进球数），使赛事信息更加完整。
- 撰写了详细的编程报告，对系统的功能以及如何使用进行了介绍。

3. 系统的二次开发

本次编程项目代码结构清晰，附有详细的介绍，因此后续还可以进行二次开发。例如：系统在基础的功能上还可以加入额外的功能；针对使用的 `socket` 通讯模块，该系统还可以将服务器端以及相关文件迁移到云服务器，变为 IP 访问；另外客户端的 GUI 界面还可以进行优化，使系统更加美观。

五、总结

本次编程项目以 2022 年卡塔尔世界杯为实际背景，以客户端-服务器端的方式开发出一个贴近实际问题的赛事管理系统，满足用户对赛事的查询、管理功能。通过完成本次编程项目，我更加系统地复习了 Python 的各类模块和库，使用编程解决了现实生活中的实际问题，进一步锻炼了我的编程能力，提高了编程思想水平和解决问题的能力。

以上就是本次编程报告与系统介绍的全部内容，如有不当之处还请老师批评指正，最后感谢助教老师的辛勤批改与姚爱萍老师一学期的教导，祝身体健康，新年快乐！