Developer Manual

Team 9: Chang Liu, Jinhao Wang, Yujie Yi

Requirement:

Environment 1:

System environment: Windows 11

Coding language: Python 3.10

Coding application: PyCharm 2022

Database application: MongoDB 1.32

Environment 2:

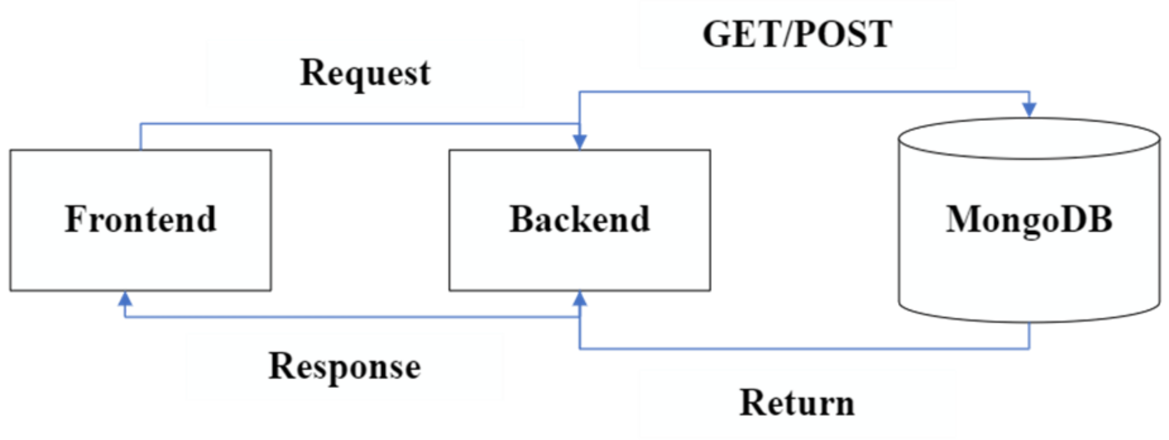
System environment: Mac OS

Coding language: Python 3.10

Coding application: PyCharm 2022

Database application: MongoDB 1.34

Module and relationship:



Frontend:

Class:

Class name	Explanation
MY_GUI	This includes methods for implementing the layout of the GUI interface and the implementation of various events in the GUI. For example, user interaction during use, display of returned results and calling database query methods.

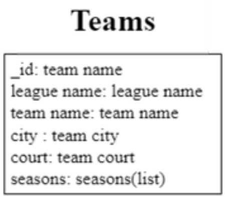
Function:

Function name	Explanation
set_init_window()	Layout methods for main loop calls.
New_league()	Layout of the create new league section. Including Labels, Text Field and Button, etc.
New_team ()	Layout of the create new team section. Including Labels, Text Field and Button, etc.
Query_team ()	Layout of the query team information section. Including Labels, Text Field and Button, etc.
Query_team_record ()	Layout of the query team record section. Including Labels, Text Field and Button, etc.
Query_game_history ()	Layout of the game records of the two teams section. Including Labels, Text Field and Button, etc.
Move_team ()	Layout of the Move team to another league section. Including Labels, Text Field and Button, etc.
Updating_team_rating ()	Layout of the update team ratings section. Including Labels, Text Field and Button, etc.
Update_current_date ()	Layout of the Update time and race automatically section. Including Labels, Text Field and Button, etc.
Query_season()	Layout of the query the season ranking section. Including Labels, Text Field and Button, etc.
update_game()	Layout of the update game score section. Including Labels, Text Field and Button, etc.
submitTeam()	Trigger event when " Submit Team " button is clicked.
submitLeague()	Trigger event when " Submit League " button is clicked.
All_clear()	Trigger event when " All Clear " button is clicked. Clear all information on the page.
query_record()	Trigger event when " Query team " button is clicked. Given a team, return its basic information (name, city, home court, current rating).
query_league()	Trigger event when " Query league " button is clicked. Return basic information of a league
query_champion()	Trigger event when " Query champion " button is clicked. given a league, return the team with the highest point each season (if there is a tie, then print all champions). Print the year, champions, and its record.

Function name	Explanation
moveTeam()	Trigger event when " Move Team" button is clicked. Teams can be moved from one league to another between seasons. However, both the league that is it moving from and to cannot be in the middle of a season.
Update()	Trigger event when " Update " (Updating Team Rating) button is clicked. Update team ratings.
Update_current()	Trigger event when " Update " (Update Current date) button is clicked. System to randomly generate the schedule.
query_season()	Trigger event when " Query season " button is clicked. Given a league and a season, print the standing of teams, ordered.
showNotice()	Error message when submitting the output.
showNotice2()	Error message when output query.
showNotice3()	Clear all error messages on the page.
change_query_league()	Trigger event when " Change league " button is clicked. Convert Interface to Query league basic information.
change_champion_query()	Trigger event when " Change champion " button is clicked. Convert Interface to Query Champions.

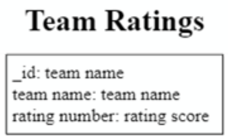
Backend:

Relationship:



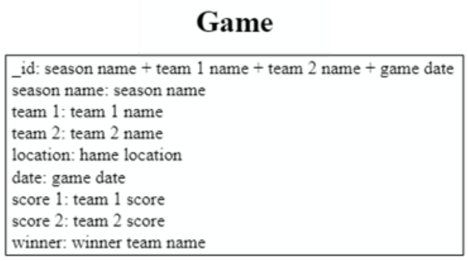
Teams

```
_id: team name
league name: league name
team name: team name
city : team city
court: team court
seasons: seasons(list)
```



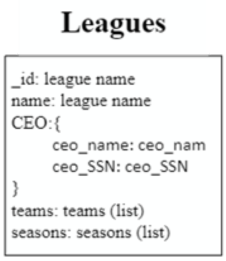
Team Ratings

```
_id: team name
team name: team name
rating number: rating score
```



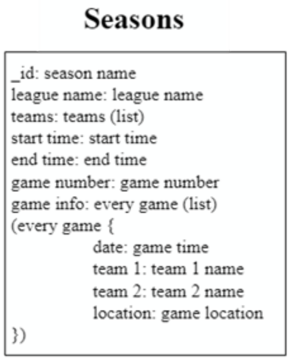
Game

```
_id: season name + team 1 name + team 2 name + game date
season name: season name
team 1: team 1 name
team 2: team 2 name
location: hame location
date: game date
score 1: team 1 score
score 2: team 2 score
winner: winner team name
```



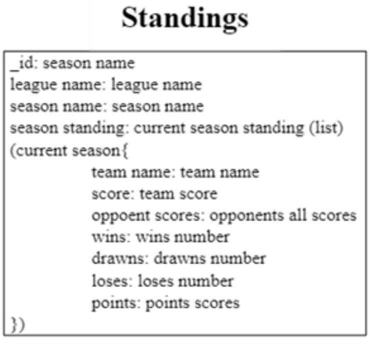
Leagues

```
_id: league name
name: league name
CEO:{
        ceo_name: ceo_nam
        ceo_SSN: ceo_SSN
}
teams: teams (list)
seasons: seasons (list)
```



Seasons

```
_id: season name
league name: league name
teams: teams (list)
start time: start time
end time: end time
game number: game number
game info: every game (list)
(every game {
                date: game time
                team 1: team 1 name
                team 2: team 2 name
                location: game location
})
```



Standings

```
_id: season name
league name: league name
season name: season name
season standing: current season standing (list)
(current season{
                team name: team name
                score: team score
                oppoent scores: opponents all scores
                wins: wins number
                drawns: drawns number
                loses: loses number
                points: points scores
})
```

Class:

Class name	Explanation
SportLeagueDatabase	It is a backend class. It implements the database link and the establishment of each sets, including leagues, seasons, teams, standings, ratings, and game. Besides, it implements the database to add, delete, change, and query.

Function:

Function name	Explanation
InputLeague(GET, league_name=None, ceo_name=None, ceo_SSN=None, start_time=None, end_time=None, team_name=None, city=None, court=None)	If GET is 'Enter a new league', create a new league and initialize a season with the league. If GET is 'Enter a new team', create a new team.
UpdateRating(team_name, new_rating)	Update the team's evaluation score.
UpdateCurrentDate(league_name, new_season_date)	Complete all games of the current season, and initialize a new season.

Function name	**Explanation**
MovingTeam(team_name, old_league_name, new_league_name, moving_date)	Move a team from a league to another league.
LeagueQuery(GET, league_name)	If GET is 1, get basic information of a league. If GET is 2, get the team with the highest point each season of the league.
UpdateGameScore(season_name, team_1, team_2, score_1, score_2)	Update two teams game scores.
TeamQuery(GET, team_name)	If GET is 1, get the team's basic information. If GET is 2, get the teams' record for each season, including games played, number of wins, number of draws, number of loses, sum of scores for its games, sum of its opponent scores’ in games, and the total number of points.
GameQuery(team_1, team_2)	Get all games that two teams played, and score. And get the date and the league where they are competing also.
SeasonQuery(league_name, season_name)	Get the standing of teams, ordered of the league with the season.