Bilkent University

CS 353

Database Systems

Project Design Report

Football Database System

Şekip Kaan EKİN 21401795 Section 1

Ekinsu BOZDAĞ 21604089 Section 3

Tanay AKGÜL 21400778 Section 3

Koray GÜRSES 21401254 Section 3

1. Revised E/R Model

According to our feedback and during the design process we changed our E/R model to have more suitable structure for our database:

· We removed the weak entity that was Dependent entity because it was not relevant for our context.

· We removed player id, agent id and we changed person id name in Person entity to id.

· We added weak Goal Entity to Game entity. The new Goal entity has time and player attributes.

· We added team attribute to Player entity.

· We changed the name of “season” to “date” in League entity.

· We removed number_of_cups attribute from Coach entity.

· We added new e-mail and password attributes to Person entity.

· We renamed of “Match” entity to Game entity.

· We changed the name of the relation between Player and Club entities from plays for to “Offer”. We added role to Offer.

· We changed the cardinality of the relation between Player and Club. We removed one to many and added total participation to many.

· We removed Budget Entity and we added its attributes to Club entity.

· We removed Foreign Keys that were player id from E/r model because they were unnecessary.

· We corrected aggregation that was wrong. We removed wrong aggregation and we added new aggregation between Player – Club – Agent. We added new relation “Set” between Aggregation box and Agent with role wage demand.

· We changed the relation between Agent entity and Player entity to “Consult” relation and we changed the cardinality to total participation.

· We added team attribute to Player entity.

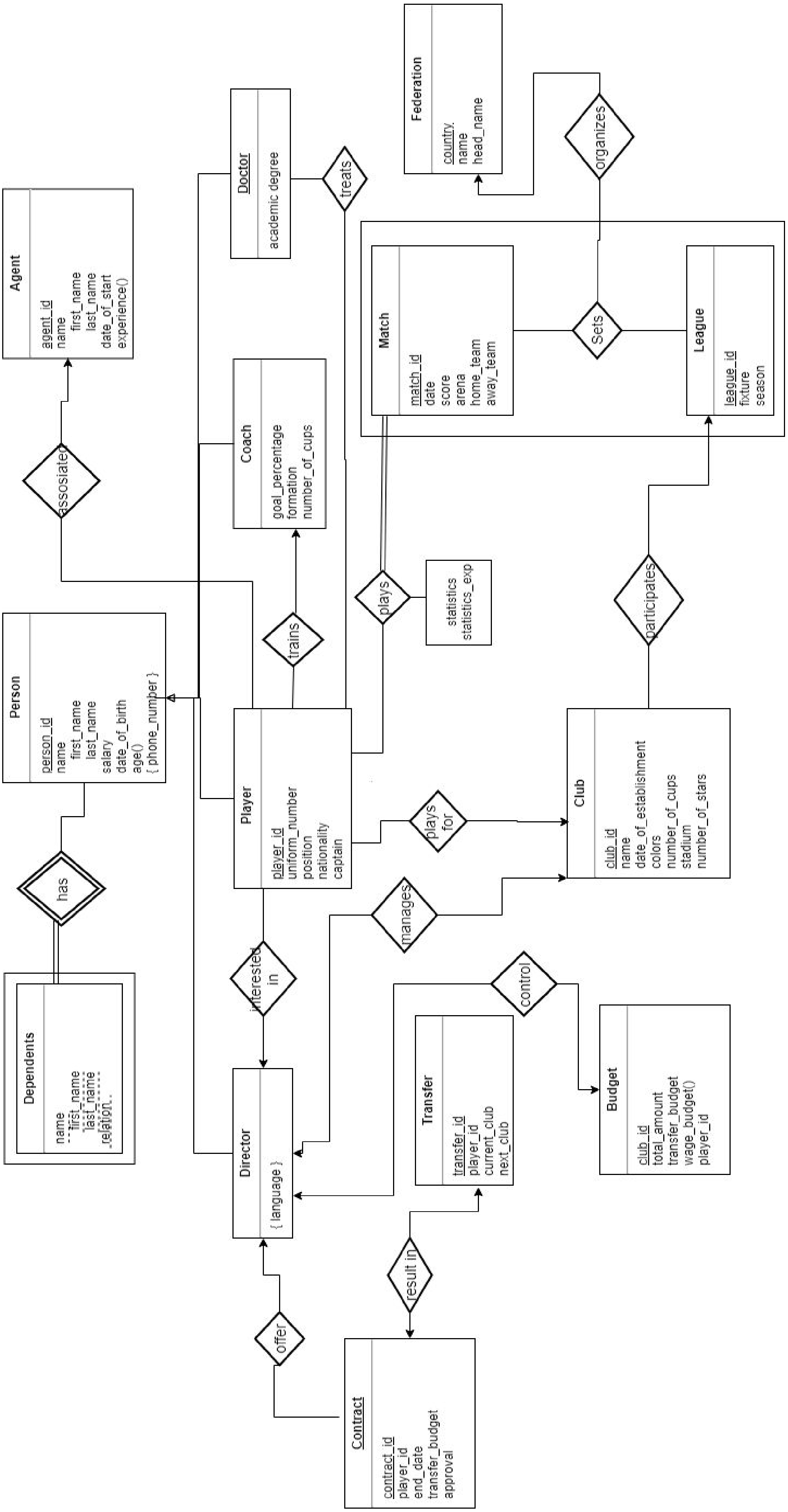
· We changed the cardinality of relation “Manages” between Director and Club to total participation.

· We changed the name of the relation between Game and League from “Sets” to “Arranges” because of repetition of relations and we changed the cardinality to total participation to many relations.

· We changed the cardinality of relation between League and Federation to total participation.

· We removed Contract entity instead of the entity we used “Contract” relation between Player and Club with role.

· We added Fan entity with fan id and favorite team attributes and we used “is a” relation with Person entity.


Agent
agent_id
name
first_name
last_name
date_of_start
experience()
assosiated
Person
person_id
name
first_name
last_name
salary
date_of_birth
age()
{ phone_number }
has
Dependents
name
first_name
last_name
relation
Doctor
academic degree
treats
Coach
goal_percentage
formation
number_of_cups
trains
Player
player_id
uniform_number
position
nationality
captain
plays
statistics
statistics_exp
plays for
interested in
Director
{ language }
manages
offer
Contract
contract_id
player_id
end_date
transfer_budget
approval
result in
Transfer
transfer_id
player_id
current_club
next_club
control
Budget
club_id
total_amount
transfer_budget
wage_budget()
player_id
Club
club_id
name
date_of_establishment
colors
number_of_cups
stadium
number_of_stars
participates
Match
match_id
date
score
arena
home_team
away_team
Sets
League
league_id
fixture
season
organizes
Federation
country
name
head_name

2. Relation Schemas

2.1. Person

Relational Model:

Person (person_id, first_name, last_name, e-mail, password, salary, date_of_birth, age(), phone_number)

Functional Dependencies:

person_id -> first_name, last_name, e-mail, password, salary, date_of_birth, age(), phone_number

e-mail -> person_id, first_name, last_name, password, salary, date_of_birth, age(), phone_number

Candidate Keys:

{ (person_id), (e-mail)}

Normal Form:

Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE Person(
person_id            INT PRIMARY KEY AUTO_INCREMENT,
first_name           VARCHAR(24) NOT NULL,
last_name            VARCHAR(24),
e-mail               VARCHAR(32) NOT NULL UNIQUE,
password             VARCHAR(32) NOT NULL,
salary               INT,
date_of_birth        DATETIME,
age                  INT,
phone_number         INT);
```

2.2. Fan

Relational Model:

Fan (fan_id, favorite_team)

Functional Dependencies:

fan_id-> favorite_team

Candidate Keys:
{ (fan_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE Fan(
fan_id                  INT PRIMARY KEY,
favourite_team          INT,
FOREIGN KEY (fan_id) REFERENCES Person(person_id),
FOREIGN KEY (favourite_team) REFERENCES Club(club_id));
```

2.3. Doctor

Relational Model:
Doctor(doctor_id,academic degree)

Candidate Keys:
{(doctor_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Doctor(
doctor_id                    INT PRIMARY KEY,
academic degree              VARCHAR(32) NOT NULL,
FOREIGN KEY (doctor_id) REFERENCES Person(person_id));
```

2.4. Agent

Relational Model:
Agent (agent_id,date_of_start,experience())

Candidate Keys:
{ (agent_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE Agent(
agent_id            INT PRIMARY KEY,
date_of_start       DATE,
experience          INT,
FOREIGN KEY (agent_id) REFERENCES Person(person_id));
```

2.5. Coach

Relational Model:
Coach (coach_id, goal_percentage, formation)

Functional Dependencies:
coach_id -> goal_percentage, formation

Candidate Keys:
{ (coach_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE Coach(
coach_id            INT PRIMARY KEY,
goal_percentage     INT,
formation           INT,
FOREIGN KEY (coach_id) REFERENCES Person(person_id));
```

2.6. Director

Relational Model:
Director(director_id,language)

Functional Dependencies:
director_id -> language

Candidate Keys:
{(director_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE  Director(
director_id              INT PRIMARY KEY,
language                 VARCHAR(32) NOT NULL,
FOREIGN KEY (director_id) REFERENCES Person(person_id));
```

2.7. Player

Relational Model:

Player (player_id, team,uniform_number, position, nationality, captain,agent,doctor,coach)

Functional Dependencies:
player_id -> team, uniform_number, position, nationality, captain,agent,doctor,coach
team, uniform_number -> player_id, position, nationality, captain,agent,doctor,coach

Candidate Keys:
{ (player_id), (team,uniform_number)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE Player(
player_id                INT,
team                     INT,
uniform_number           INT,
nationality              INT,
captain                 BIT(1),
agent                    INT,
doctor                   INT,
coach                    INT,
position ENUM('GK', 'SW', 'CB', 'RB', 'LB', 'LWB', 'RWB', 'DM', 'CM', 'AM',
'RM', 'LM', 'LW', 'SS', 'RW', 'CF'),
UNIQUE(team,uniform_number),
FOREIGN KEY (agent) REFERENCES Agent(agent_id),
FOREIGN KEY (doctor) REFERENCES Doctor(doctor_id),
FOREIGN KEY (coach) REFERENCES Coach(coach_id),
FOREIGN KEY (player_id) REFERENCES Person(person_id),
FOREIGN KEY (team) REFERENCES Club(club_id));
```

2.8. Club

Relational Model:

Club (club_id, name, date_of_establishment, colors, number_of_cups, stadium, number_stars, transfer_budget, wage_budget, total_amount)

Functional Dependencies:

club_id -> name, date_of_establishment, colors, number_of_cups, stadium, number_of_stars, transfer_budget, wage_budget, total_amount

Candidate Keys:

{(club_id)}

Normal Form:

Boyce-Codd Normal Form (BCNF)

SQL Table :

```
CREATE TABLE Club(
club_id                  INT PRIMARY KEY,
name                     VARCHAR(32) NOT NULL,
colors                   VARCHAR(32) NOT NULL,
date_of_establishment    DATETIME, NOT NULL,
stadium                  VARCHAR(32),
number_of_cups           INT,
number_of_stars          INT,
transfer_budget          INT,
wage_budget              INT,
total_amount             INT);
```

2.9. Game

Relational Model:
Game (Game_id, date,score,arena, home_team, away_team,league_number)

Functional Dependencies:
Game_id-> date,score,arena, home_team, away_team,league_number

Candidate Keys:
{(Game_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Game(
Game_id             INT PRIMARY KEY,
date                DATETIME,
score               INT,
arena               VARCHAR(32),
league_number       INT,
home_team           VARCHAR(32),
away_team           VARCHAR(32),
FOREIGN KEY (studium) REFERENCES Club(stadium),
FOREIGN KEY (league_number) REFERENCES League(league_id),
FOREIGN KEY (home_team) REFERENCES Club(club_id),
FOREIGN KEY (away_team) REFERENCES Club(club_id));
```

2.10 Goal

Relational Model:
Goal (Game_id, time, player)

Functional Dependencies:
Game_id, time -> player

Candidate Keys:

{(Game_id,time)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Goal(
 Game_id        INT PRIMARY KEY,
time            TIME,
player          INT,
PRIMARY KEY (Game_id, time),
FOREIGN KEY (Game_id) REFERENCES Game(game_id),
FOREIGN KEY (player) REFERENCES Player(player _id));
```

2.11. League

Relational Model:
League(league_id,date)

Functional Dependencies:
league_id-> date

Candidate Keys:
{(league_id)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE League(
league_id       INT PRIMARY KEY,
date                DATE);
```

2.12 Federation

Relational Model:
Federation(country, name, head_name)

Functional Dependencies:
country-> name, head_name

Candidate Keys:
{(country)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Federation(
country                 VARCHAR(32) PRIMARY KEY,
name                    VARCHAR(32),
head_name               VARCHAR(32));
```

2.13. Trains

Relational Model:
Trains (coach, player)

Candidate Keys:
{(coach, player)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Trains(
coach           INT,
player          INT,
PRIMARY KEY (coach, player),
FOREIGN KEY (player) REFERENCES Player(player_id),
FOREIGN KEY (coach) REFERENCES coach(coach_id));
```

2.14. Offer

Relational Model:
Offer (player, team_prev, team_next, date, transfer_fee)

Functional Dependencies:
player, team_prev, team_next, date-> transfer

Candidate Keys:
{(player, team_prev, team_next, date)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Offer(
player          INT,
team_sell       INT,
team_buy        INT,
date            DATE,
transfer_fee   INT,
team_prev        INT,
team_next       INT,
PRIMARY KEY (player, team_sell, team_buy, date),
FOREIGN KEY (player) REFERENCES Player(player_id),
FOREIGN KEY (team_prev) REFERENCES Club(club_id),
FOREIGN KEY (team_next) REFERENCES Club(club_id));
```

2.15. Sets

Relational Model:
Sets(player, team_prev, team_next, date, wage_demand)

Functional Dependencies:
player, team_prev, team_next, date -> wage_demand

Candidate Keys:
{(player, team_prev, team_next, date)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Sets(
player INT,
team_sell INT,
team_buy INT,
date DATE,
wage_demand INT,
team_prev INT,
team_next INT,
PRIMARY KEY (player, team_prev, team_next, date),
FOREIGN KEY (player, team_prev, team_next, date) REFERENCES Offer));
```

2.16. Contract

Relational Model:
Contract (player, team_prev, team_next,start_date,end_date, transfer_fee, wage, termination_fee)

Functional Dependencies:
player, team_prev, team_next,start_date,end_date -> wage

Candidate Keys:
{(player, team_prev, team_next, start_date), (player, team_prev, team_next, end_date)}

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Contract(
player              INT,
team_prev           INT,
team_next           INT,
start_date          DATE,
contract_end        DATE,
ransfer_fee         INT,
wage                INT,
termination_fee     INT,
PRIMARY KEY (player, team_prev, team_next,start_date),
UNIQUE (player, team_prev, team_next, contract_end),
FOREIGN KEY (player) REFERENCES Player(player_id),
FOREIGN KEY (team_prev) REFERENCES Club(club_id),
FOREIGN KEY (team_next) REFERENCES Club(club_id));
```

2.17. Organizes

Relational Model:
Organizes(league,federation)

Normal Form:
Boyce-Codd Normal Form (BCNF)

SQL Table:

```
CREATE TABLE Organizes(
leauge              INT,
federation          INT,
PRIMARY KEY (league, federation),
FOREIGN KEY (league) REFERENCES League(league_id),
FOREIGN KEY (federation) REFERENCES Federation(country));
```

3. Functional Dependencies and Normalization of Table

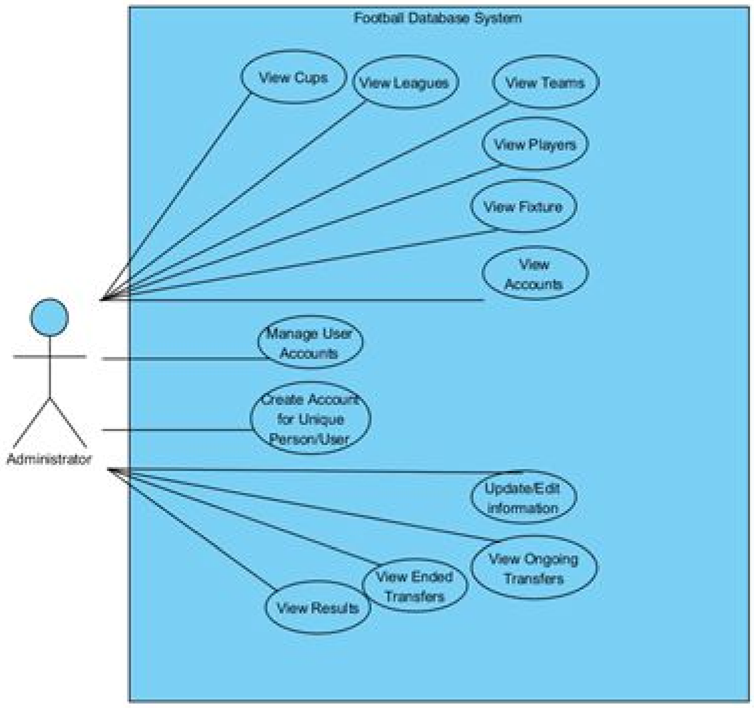
All functional dependencies and normal forms are indicated in Relation Schemas of this Project Design Report. The relations are all in Boyce-Code Normal Form. We concluded that there is no need for any decomposition or normalization.

4. Functional Components

4.1. Use Cases / Scenarios

Football Database System is providing information to two kinds of user. There are regular users like fans, and there are unique users that includes players, agents, club directors, doctors, coaches and administrators. There are common services provided for both regular users and unique users however, there are some functionalities that can be accessed by specific type of user in the system.

4.1.1. Administrator

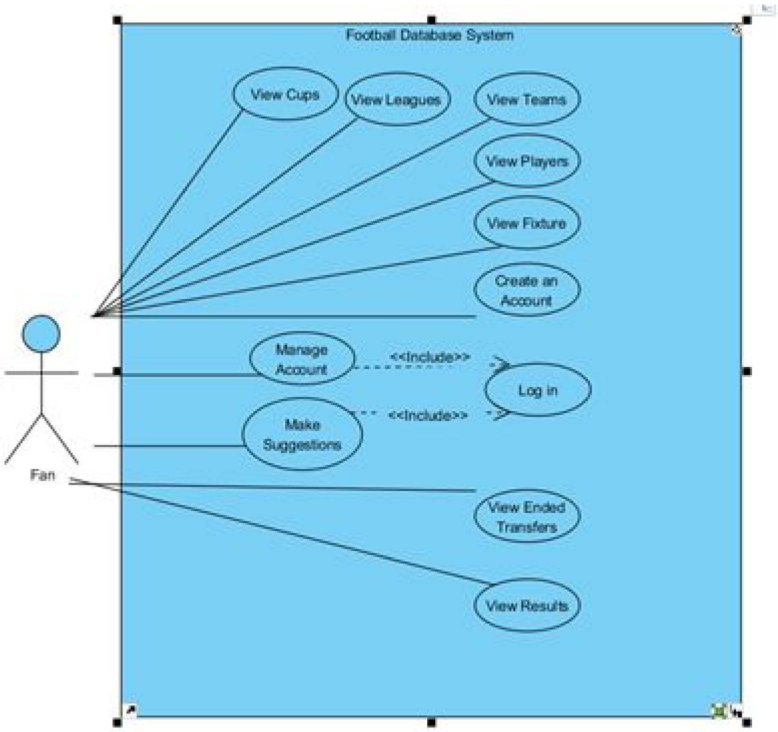


· **Create Account for Unique Person:** Administrators or admins can create accounts for unique users (players, agents, club directors…). They can determine the username and passwords for that specific account, but also, they can give permission to the account owner to change their username and password

· **Manage Users Accounts:** If there is a security issue detected in the account, administrators can manage the user account.

- **View Accounts:** Admins can view all the accounts. They can also detect the user type of the account whether it is a fan or is a doctor, etc.
- **Update/Edit Information:** Admins can update the information if necessary whether there is a suggestion or not.
- **View Teams:** Admin can search for teams and Admin can select the specific team to see team’s detailed information. The teams are also selectable from the league and cups.
- **View Players:** Admins can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.
- **View Fixture:** Admins can see the fixture by selecting the league.
- **View Ongoing Transfers:** Admins can see the ongoing transfer offers, and can see the background information about that transfer such as the team that got offered transfer, and the team that offered the transfer, the transfer fee, contract length etc.
- **View Ended Transfers:** Admins can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.
- **View Results:** Admins can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.
- **View Cups:** Admins can view general and detailed information about the cups, such as previous winners, match results and the statistics
- **View Leagues:** Admins can view general and detailed information about leagues such as the previous winners, score table of the league including previous years’, fixtures and results.

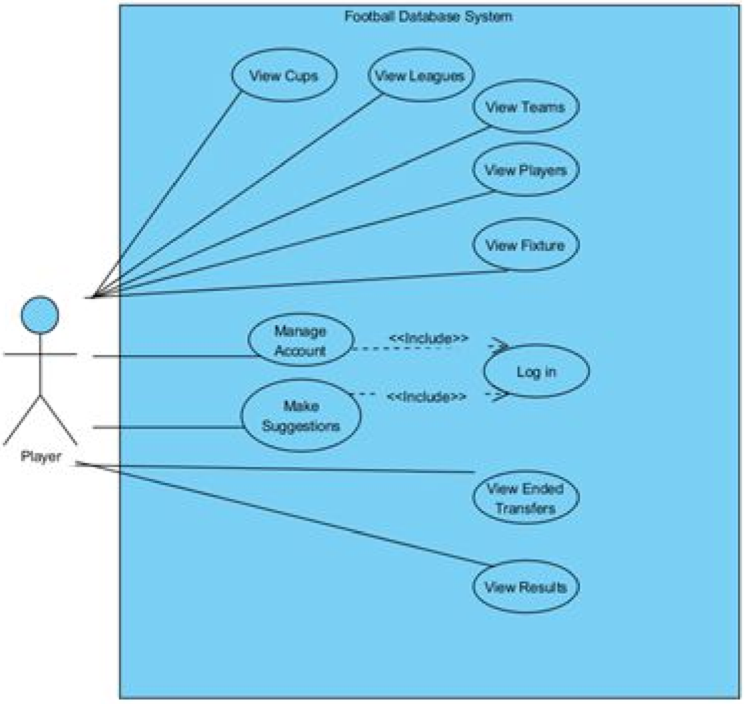
4.1.2 Fan/Regular User



- **Login:** Fans or regular users login to the system by entering the password and username or e-mail address. Fans must log in the system in order to manage their account, make suggestions, and also set their favorite teams and players so that fans can see the relevant information about their favorite teams and players.
- **Create Account:** Fans can create an account in the system by entering, appropriate and unique username, password and an e-mail address.
- **Manage Account:** Fans can update their account information such as changing password, username and e-mail, can update favorite teams and players by adding or removing them. Changed username and password should be appropriate and unique.
- **Make Suggestions:** Fans can make suggestions to the system, for example fans might offer some user-interface changes or new functionality. Also, fans might detect an error in the information and they can report it to the system. All suggestions will be stored, after consideration, changes will be applied to the system.
- **View Teams:** Fans can search for teams and can select the specific team to see team’s detailed information. The teams are also selectable from the league and cups.
- **View Players:** Fans can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.
- **View Fixture:** Fans can see the fixture by selecting the league.

- **View Ended Transfers:** Fans can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.
- **View Results:** Fans can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.
- **View Cups:** Fans can view general and detailed information about the cups, such as previous winners, match results and the statistics
- **View Leagues:** Fans can view general and detailed information about leagues such as the previous winners, score table of the league including previous years’, fixtures and results.

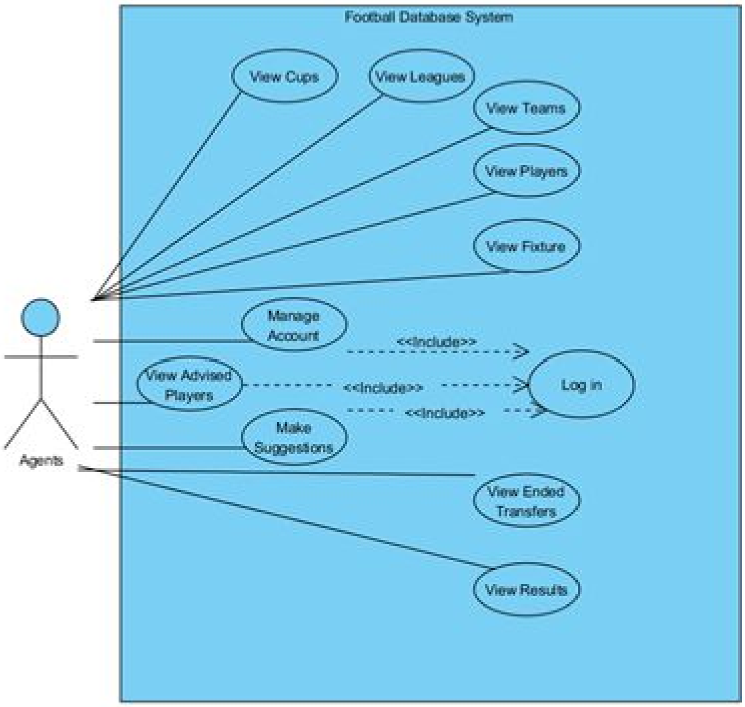
4.1.3 Player



- **Login:** Players can login the system with the username and password that given from the admin. They can manage account and make suggestions when they are logged in.
- **Manage Account:** Players can change their username and password.
- **Make Suggestions:** Players can make suggestions to the system. All suggestions will be stored, after consideration, changes will be applied to the system.
- **View Teams:** Players can search for teams and can select the specific team to see team’s detailed information. The teams are also selectable from the league and cups.
- **View Players:** Players can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.
- **View Fixture:** Players can see the fixture by selecting the league.
- **View Ended Transfers:** Players can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.
- **View Results:** Player can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.
- **View Cups:** Player can view general and detailed information about the cups, such as previous winners, match results and the statistics

·   **View Leagues:** Player can view general and detailed information about leagues such as the previous winners, score table of the league including previous years', fixtures and results.

4.1.4 Agent



·   **Login:** Agents can login the system with the username and password that given from the admin. They can manage account and make suggestions when they are logged in.

·   **Manage Account:** Agents can change their username and password.

·   **View Advised Players:** Agents can view a table that contains the information of the players that they advised.

·   **Make Suggestions:** Agents can make suggestions to the system. All suggestions will be stored, after consideration, changes will be applied to the system.

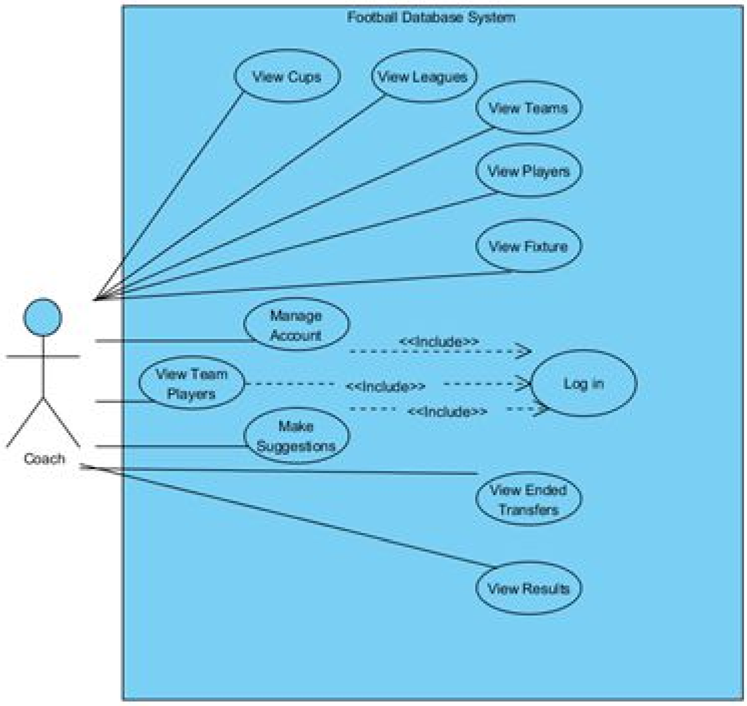
·   **View Teams:** Agents can search for teams and can select the specific team to see team's detailed information. The teams are also selectable from the league and cups.

·   **View Players:** Agents can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.

·   **View Fixture:** Agents can see the fixture by selecting the league.

- **View Ended Transfers:** Agents can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.
- **View Results:** Agents can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.
- **View Cups:** Agents can view general and detailed information about the cups, such as previous winners, match results and the statistics
- **View Leagues:** Agents can view general and detailed information about leagues such as the previous winners, score table of the league including previous years', fixtures and results.

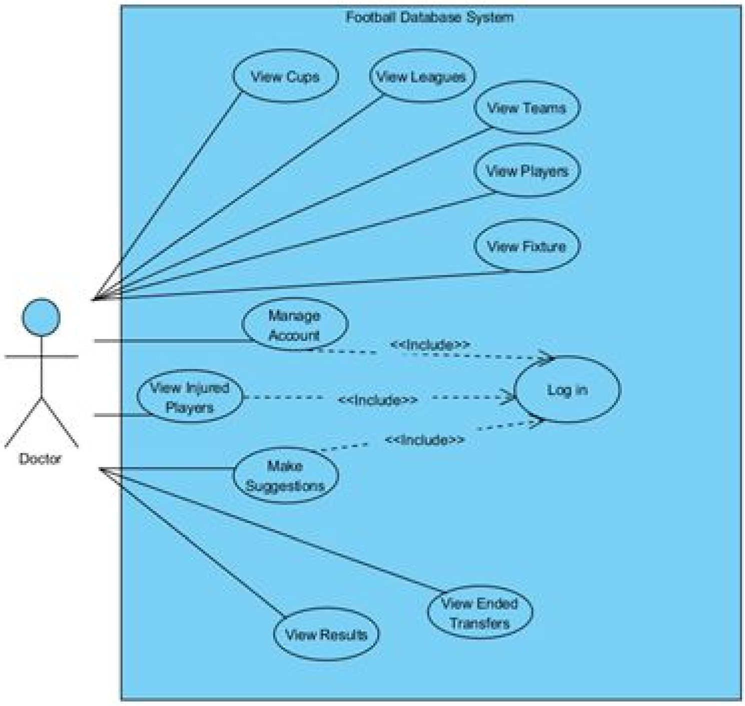
4.1.5 Coach



- **Login:** Coaches can login the system with the username and password that given from the admin. They can manage account and make suggestions when they are logged in.
- **Manage Account:** Coaches can change their username and password.
- **View Team Players:** Coaches can view a table that contains the information of the team's players.
- **Make Suggestions:** Coaches can make suggestions to the system. All suggestions will be stored, after consideration, changes will be applied to the system.
- **View Teams:** Coaches can search for teams and can select the specific team to see team's detailed information. The teams are also selectable from the league and cups.

- **View Players:** Coaches can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.
- **View Fixture:** Coaches can see the fixture by selecting the league.
- **View Ended Transfers:** Coaches can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.
- **View Results:** Coaches can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.
- **View Cups:** Coaches can view general and detailed information about the cups, such as previous winners, match results and the statistics
- **View Leagues:** Coaches can view general and detailed information about leagues such as the previous winners, score table of the league including previous years', fixtures and results.

4.1.6 Doctor



· **Login:** Doctors can login the system with the username and password that given from the admin. They can manage account and make suggestions when they are logged in.

· **Manage Account:** Doctors can change their username and password.

· **View Injured Players:** Doctors can view a table that contains the information of the team players' health condition.

· **Make Suggestions:** Doctors can make suggestions to the system. All suggestions will be stored, after consideration, changes will be applied to the system.

· **View Teams:** Doctors can search for teams and can select the specific team to see team's detailed information. The teams are also selectable from the league and cups.

· **View Players:** Doctors can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.

· **View Fixture:** Doctors can see the fixture by selecting the league.

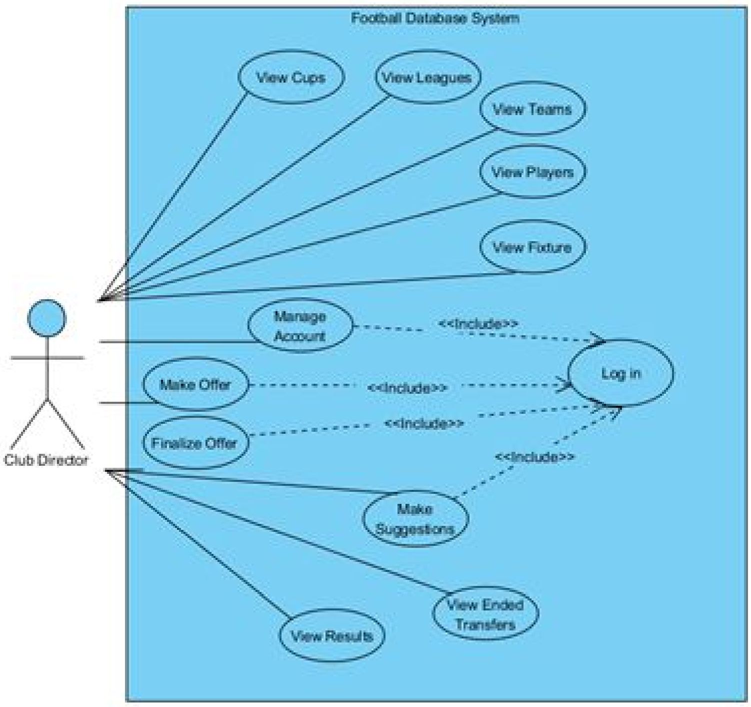
· **View Ended Transfers:** Doctors can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.

· **View Results:** Doctors can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.

· **View Cups:** Doctors can view general and detailed information about the cups, such as previous winners, match results and the statistics

· **View Leagues:** Doctors can view general and detailed information about leagues such as the previous winners, score table of the league including previous years', fixtures and results.

4.1.7 Club Director



· **Login:** Club Directors can login the system with the username and password that given from the admin. They can manage account and make suggestions when they are logged in.

· **Manage Account:** Club Directors can change their username and password.

· **Make Offer:** Club Directors can make an offer to other teams' players by setting a transfer fee or club directors can contract renewal to their own players.

· **Finalize Offer:** Club Directors can receive offer from another team's club director. They can accept or refuse the offers.

· **Make Suggestions:** Club Directors can make suggestions to the system. All suggestions will be stored, after consideration, changes will be applied to the system.

· **View Teams:** Club Directors can search for teams and can select the specific team to see team's detailed information. The teams are also selectable from the league and cups.

- **View Players:** Club Directors can search for players and can get detailed information about that player such as wage, statistics, which clubs he played for, nationality etc.
- **View Fixture:** Club Directors can see the fixture by selecting the league.
- **View Ended Transfers:** Club Directors can view the ended transfers, and the background information about that transfer such as, the team that buy and sell, transfer fee, wage, contract length, the name of the player, etc.
- **View Results:** Club Directors can view the results of the matches or competitions by looking through the specifying teams or looking through the leagues or cups.
- **View Cups:** Club Directors can view general and detailed information about the cups, such as previous winners, match results and the statistics
- **View Leagues:** Club Directors can view general and detailed information about leagues such as the previous winners, score table of the league including previous years', fixtures and results.

4.2 Algorithm

4.2.1 Transfer Related Algorithms

In order to make a transfer, club director should log in the system and make an offer. There are two kinds of transfer in this system. First one is, the club director wants to buy a player from another team. Second one is, club director wants to renew one of the player's contract. Both functions can be done by entering the transfer section. Transferring a player, affects the players and attributes directly because the club director will make an offer and the other team's club director can accept, reject the offer and, they can alternatively respond the offer by changing the value of the player to counter the offer. If the other team's director rejects the offer, the offer will be terminated so, club director must make a new offer however, if the value of the player has changed by the club director, then the transfer status should still be active.

4.2.2 Champion Selecting Algorithm

In this system we determine the champion of the league by checking the points that the teams had earned. If the points are equal, goal average will be checked. If they are equal too goals scored will be checked for the teams. This champion system might differ to league to league.

4.2.3 Logical Requirements

Logical errors are expected to occur in attributes of domain type data date, time and datetime. In the offer table (which is a relation between player and club), we have a date attribute to store the date of the offer. In contract table, there are start_date and end_date which refer to start and end of the contract. If there is a transfer offer, start_date and date of the offer attribute should be checked because start_date can't occur before the date attribute in the offer table. If the offer attribute is contract renewal, it can be in the range of start_date and end_date.

4.3 Data Structures

Our relational schema uses Numeric Type, String Type, Date and Time types.

Numeric types are used for storing numeric data, for example, ID numbers, transfer fee, etc. We used **INT** and **BIT** for numeric types. We used **BIT** to decrease space usage. **BIT** is used for determining whether the player is a captain or not.

String types are required to store attributes which consists of characters such as username, password, etc. We used **VARCHAR** in terms of string type.

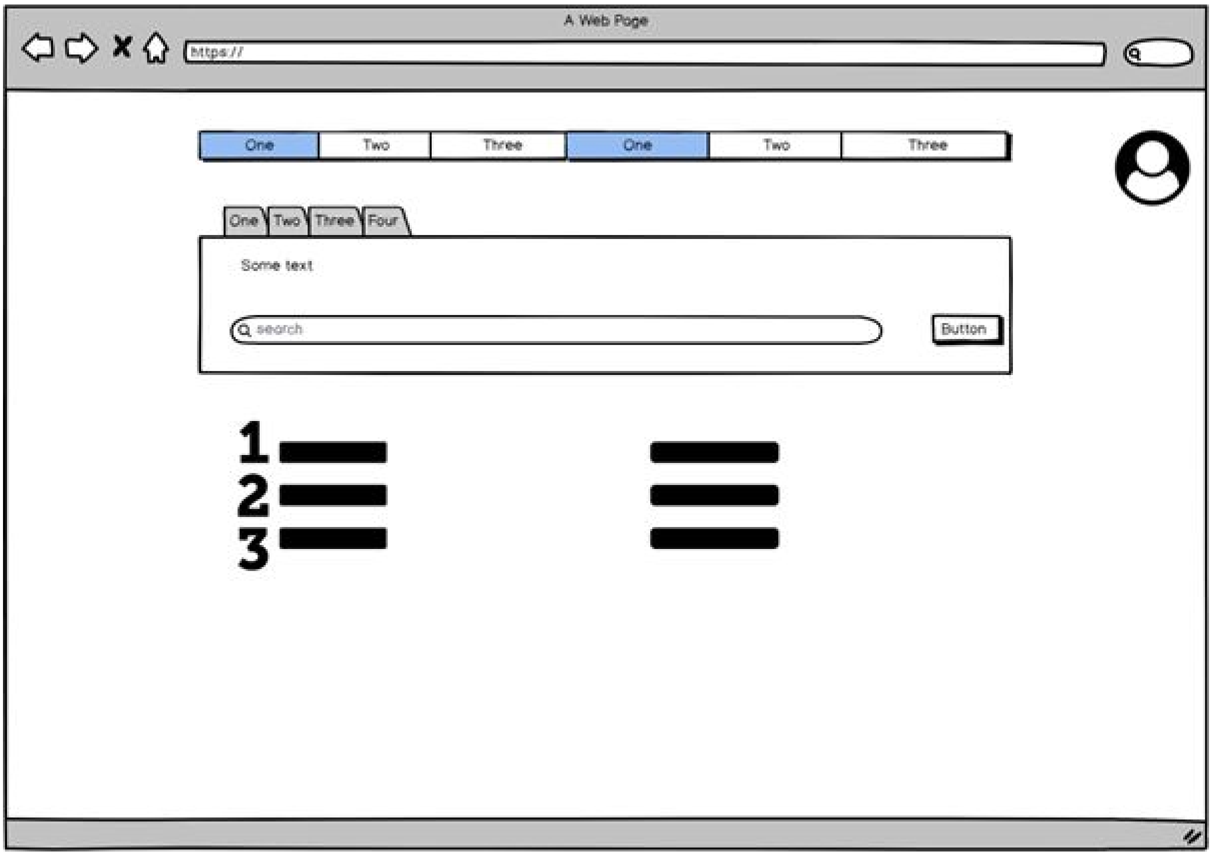
Date and Time types are used for keeping the time values such as date of the match. Those dates and times are storing with the domains of **DATE**, **DATETIME** and **TIME**.

Enum type is used for distinguishing variables. We use Enum type for distinguishing the play position of the football players by using the domain **ENUM**.

5. User Interface Design And Corresponding SQL Statements

5.1 Fan User Interface Design

5.1.1 Main Page

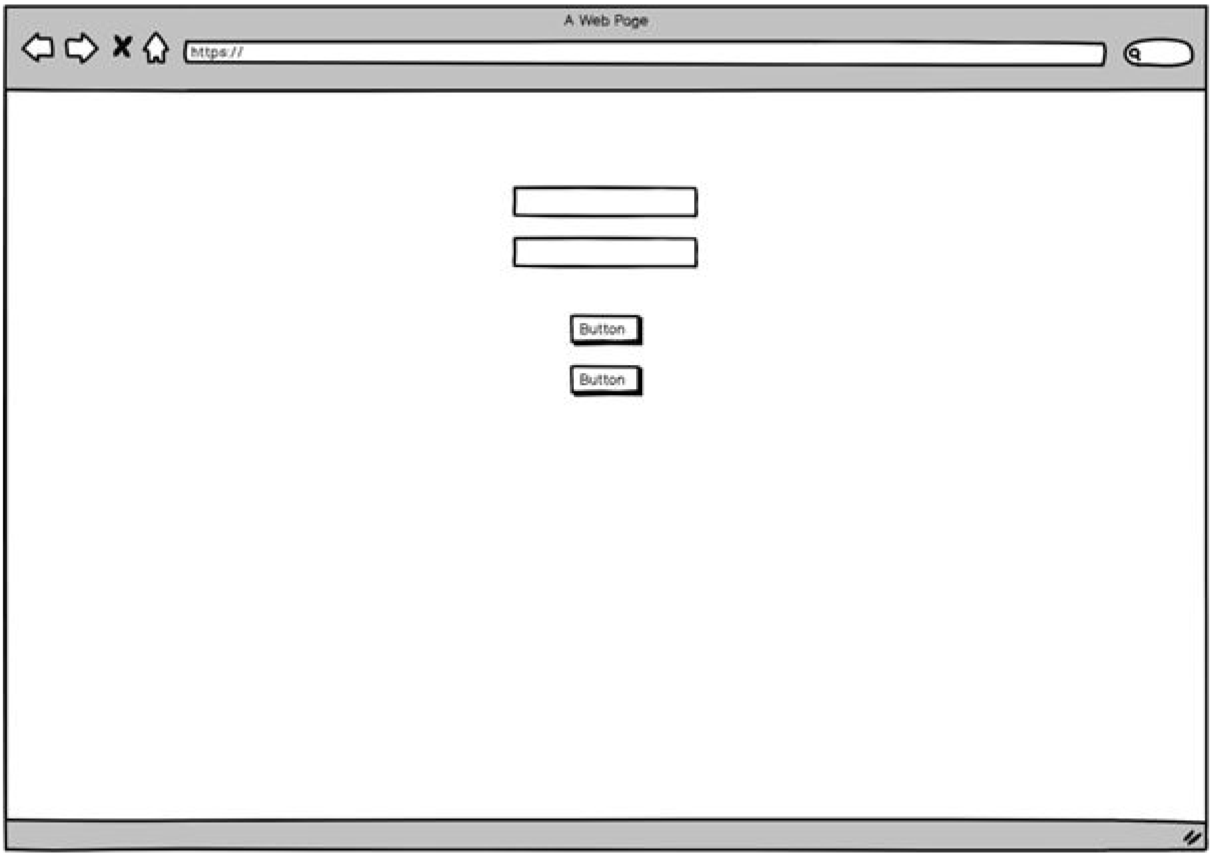


Inputs: @searchkey
Process: The main page of Football Database Management is above. On the top, there are buttons that will take user to the main page, leagues page, cups page and teams page. On the right to corner, there is a user button that user can login or go to sign up page. Below the buttons, there is a search box for user to search for people related to football (players, directors, doctors, agents), teams, leagues or cups. There is a list of top 10 richest teams on the left bottom and 5 leagues from Turkey with their page links.

SQL Statements:
Displaying 10 richest teams:

```
CREATE VIEW clubs_top10(club_name, total_amount, wage_budget, transfer_budget)
AS SELECT(name, total_amount, wage_budget, transfer_budget)
FROM club
SELECT club_id,SUM(wage_budget) + SUM(transfer_budget) AS total amount
FROM club
GROUP BY(club_id)
ORDER BY total_amount DESC
LIMIT 10
```

5.1.2 Login Page



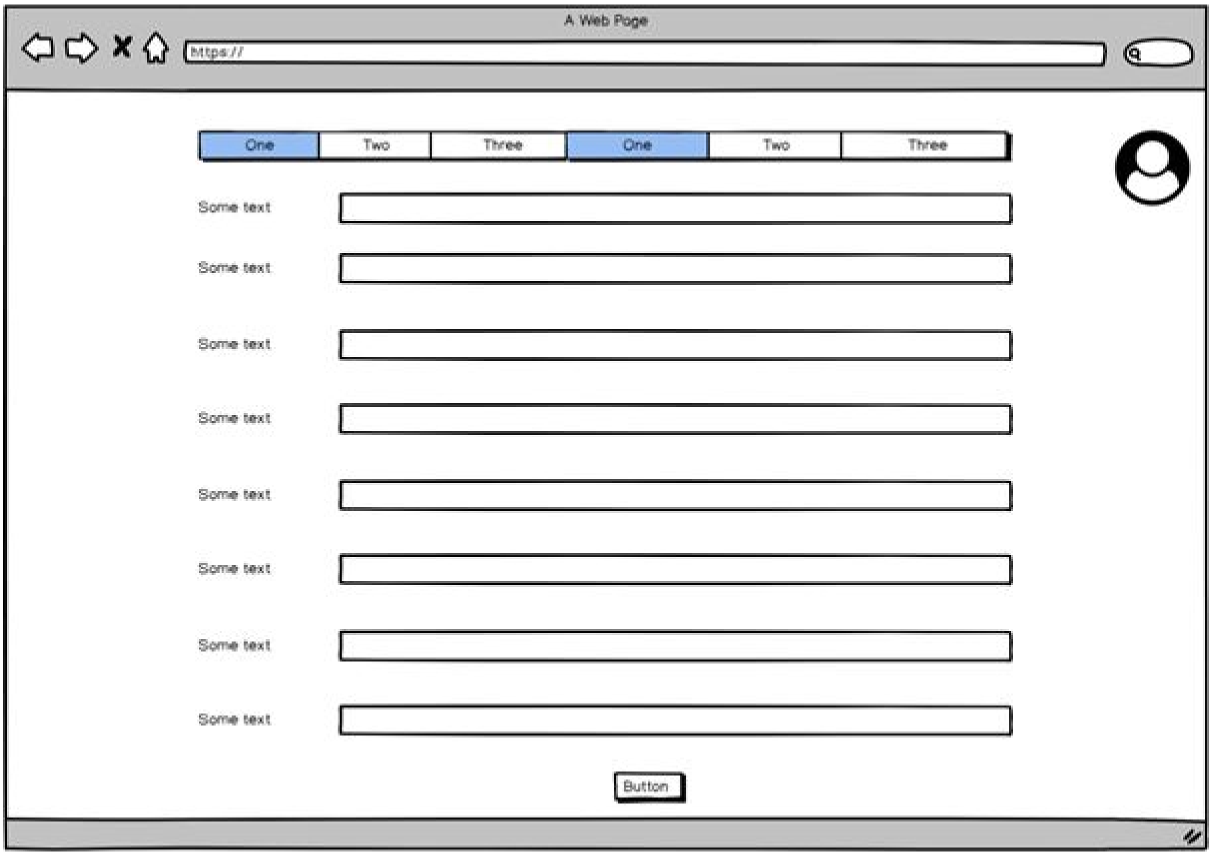
Inputs: @person_id, @password

Process: Users can enter their usernames and passwords to login or go to sign up page from here.
SQL Statements:

Logging in

```
SELECT *
FROM Person
WHERE(person_id = @person_id OR
          email = @email) AND password = @password
```

5.1.3 Sign Up Page / Manage User Page



Inputs: @person_id, @password, @firstname, @lastname, @email, @favorite_team, @date_of_birth, @phone_number, @newperson_id, @newpassword, @newmail, @confirmpassword, @newfavorite_team

Process: Logged in users can edit their information on this page, their user name and already known information will come filled. For a few users, all page will be blank and after filling up, system will record these information as a new user.

SQL Statements:

Registration

```
INSERT INTO fan
VALUES(@firstname, @surname, @email, @password, @confirmpassword)
WHERE @password = @confirmpassword
```

View Fan Details

```
SELECT person_id, email, password, fullname, favorite_team
FROM fan
WHERE person_id = @person_id
```

Change User Name

```
UPDATE person
SET person_id = @newperson_id
WHERE person_id = @person_id
```

Change Password

```
UPDATE person
SET password = @newpassword
WHERE person_id = @person_id AND password = @newpassword
```

Change Email

```
UPDATE person
SET email = @newemail
WHERE person_id = @person_id
```

Add Date of Birth

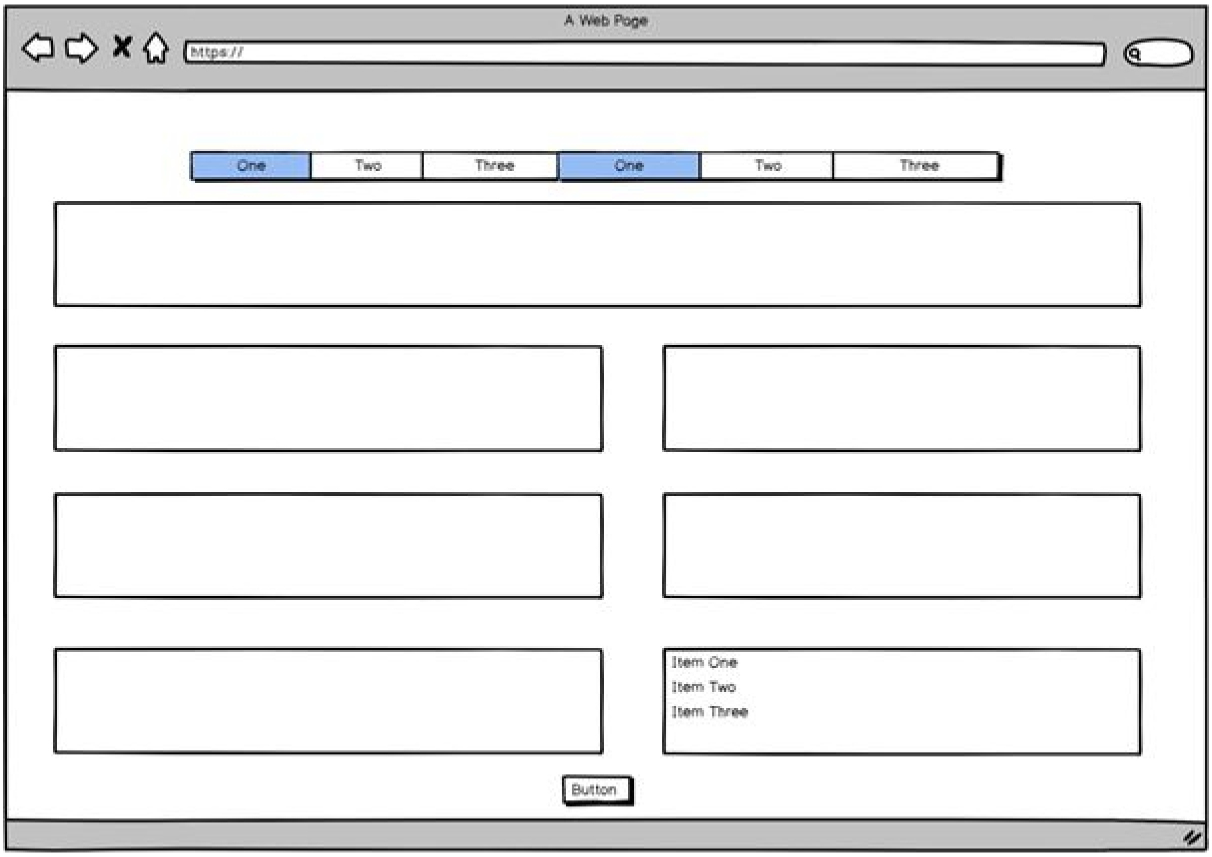
```
INSERT INTO person.date_of_birth
VALUES (date_of_birth, @date_of_birth)
```

Add Favorite Team

INSERT INTO fan.favorite_team

VALUES (favorite_team, @newfavorite_team)

5.1.4 Team/Player Page



inputs: @searchquery, @club_search

Process: By clicking on the Clubs or Players on the main page or searching for one will bring the user to this page. Information of the team or the player will be shown here. inputs: @searchquery, @club_search

SQL Statements:

Searching Clubs:

SELECT name, colors

FROM club

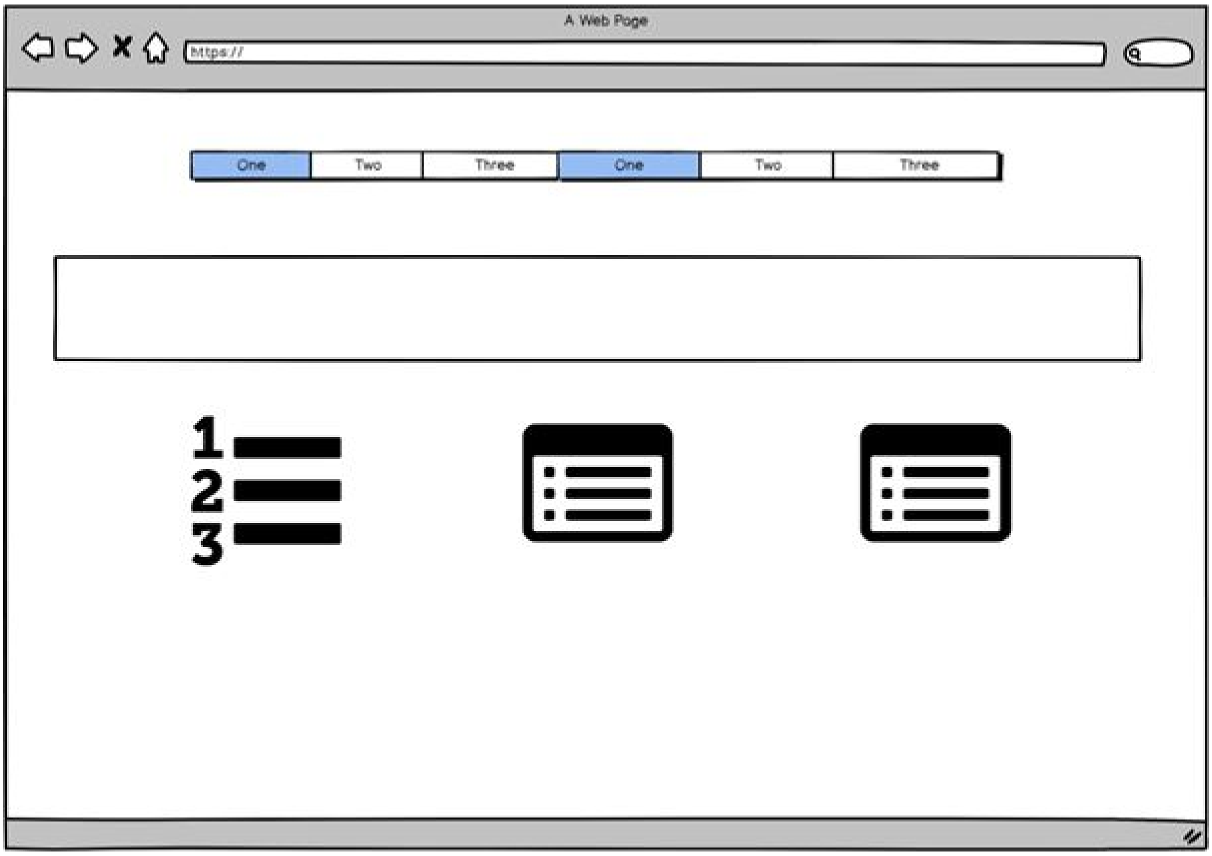
WHERE @club_search = @club_name

Searching Player:

inputs: @searchquery, @player_search

```
SELECT first_name,last_name
FROM Person
WHERE @player_search = first_name&&last_name
```

5.1.5 League/Cup Page

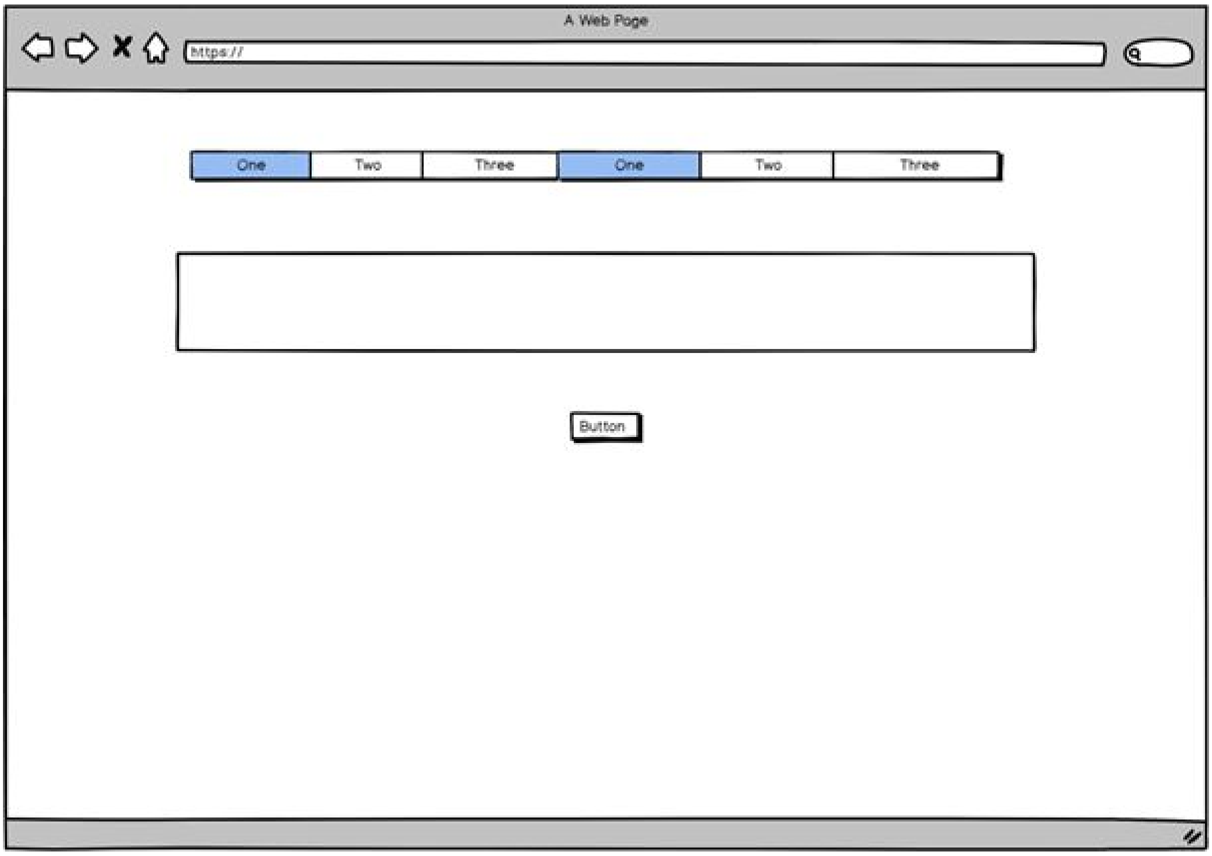


Process: By clicking on the Leagues or Cups on the main page or searching for one will bring the user to this page. Top 5 teams, fixture and the results of the latest games of the league or the cup will be shown here.

SQL Statements:

```
 SELECT league id, date, number_of_ cups
 WHERE league_id =  @clubd_id AND date = L.date AND cups = C. number_of_cups
 END
 FROM League L and Club C
 ORDER BY(date);
```

5.1.6 Suggestions Page



Inputs: @string

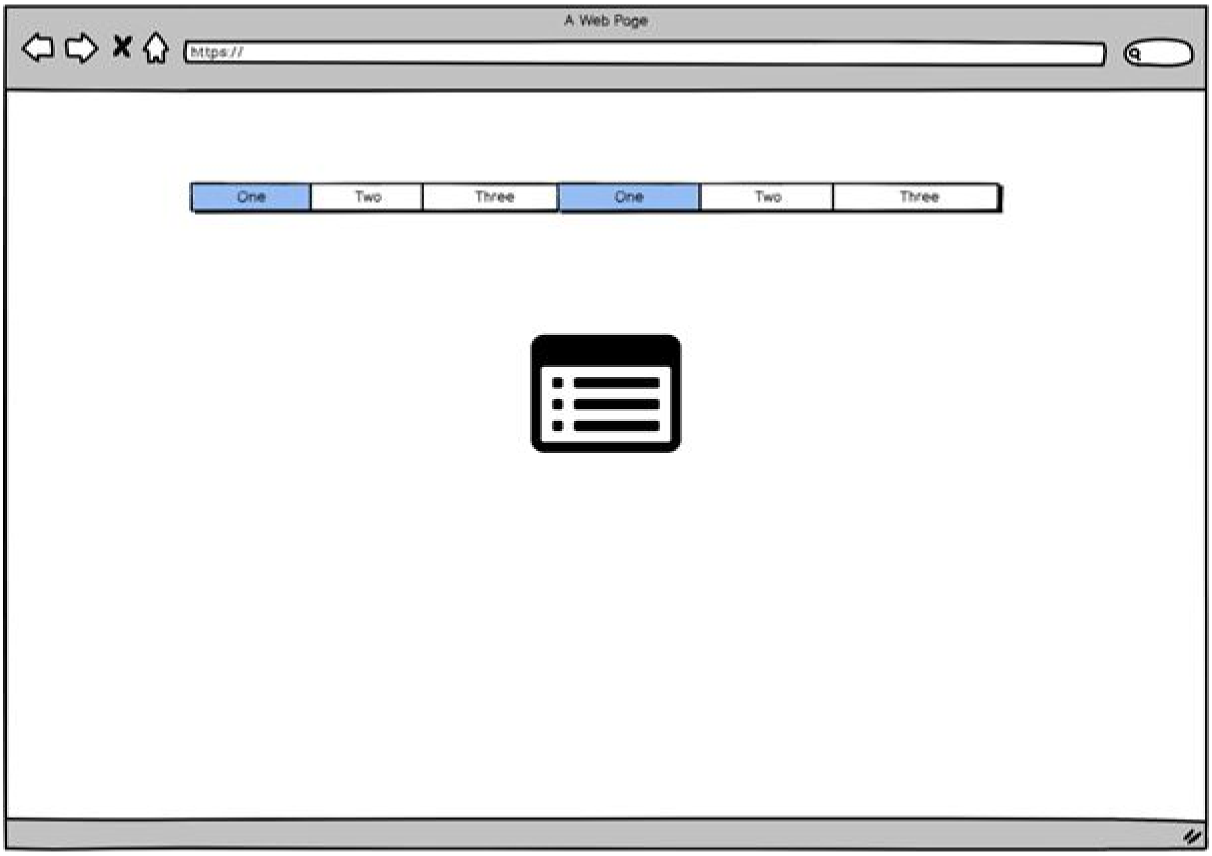
Process: User can write a suggestion and send it to the developers of the page.

5.2 Player User Interface Design

Players doesn't have any difference with the user interface from users. Only difference is that, their account is given by the administrators.

5.3 Agent User Interface Design

5.3.1 Advised Players Page



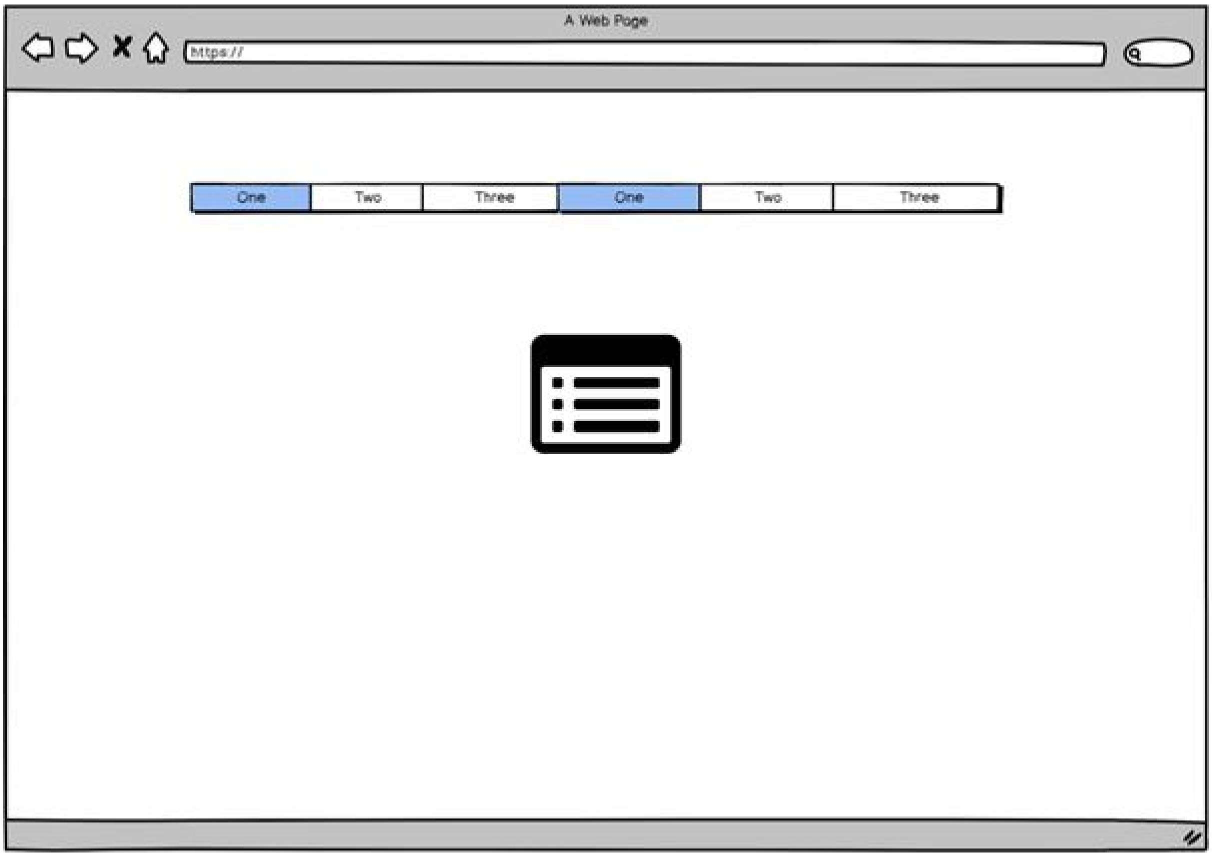
Process: Other than any default pages, agents can view their advised players from the button on the main page.

SQL Statements:

```
SELECT player_name
FROM Player P, Agent A
WHERE P.agent = A.id
```

5.4 Doctor User Interface Design

5.4.1 Patients Page



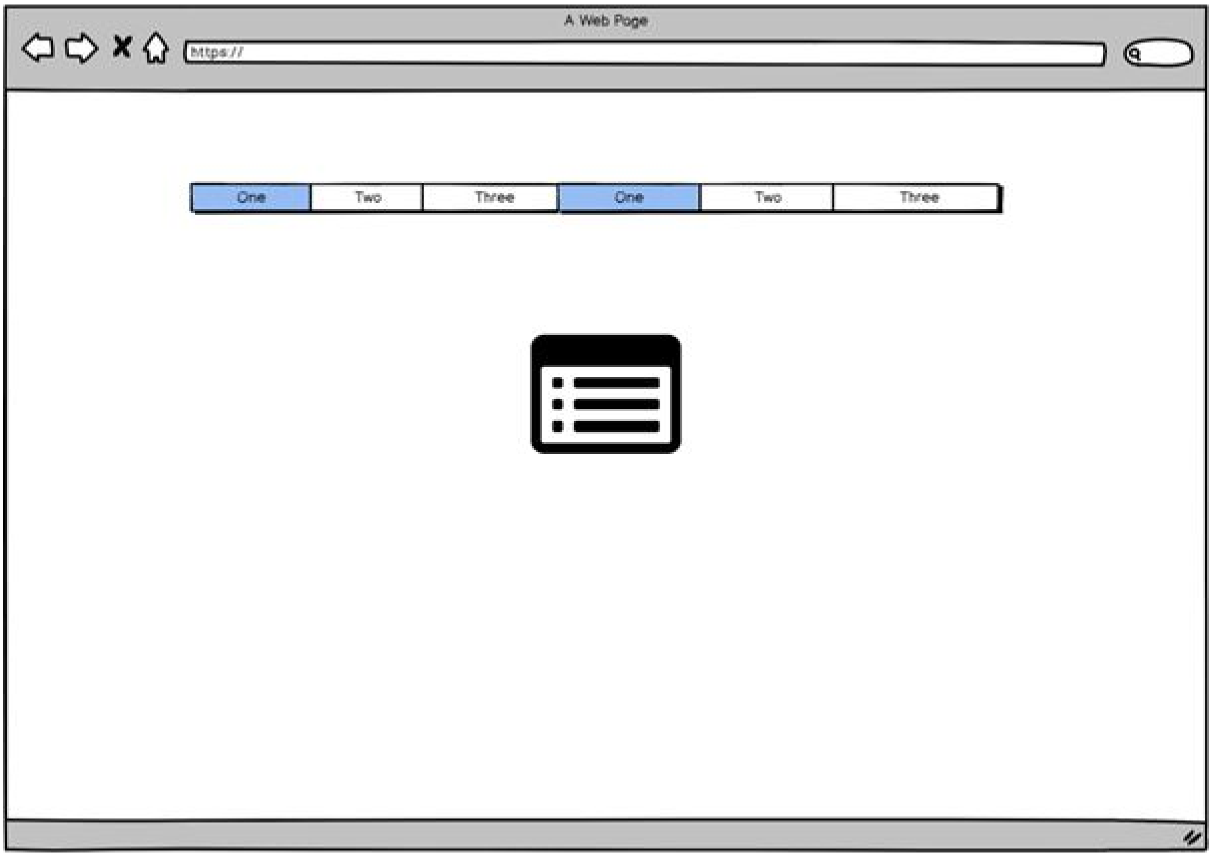
Process: Other than any default pages, doctors can view their patients from the button on the main page.

SQL Statements:

```
SELECT player_name
FROM Player P, Doctor D
WHERE P.doctor = D.id
```

5.5 Coach User Interface Design

5.5.1 Trained Players Page

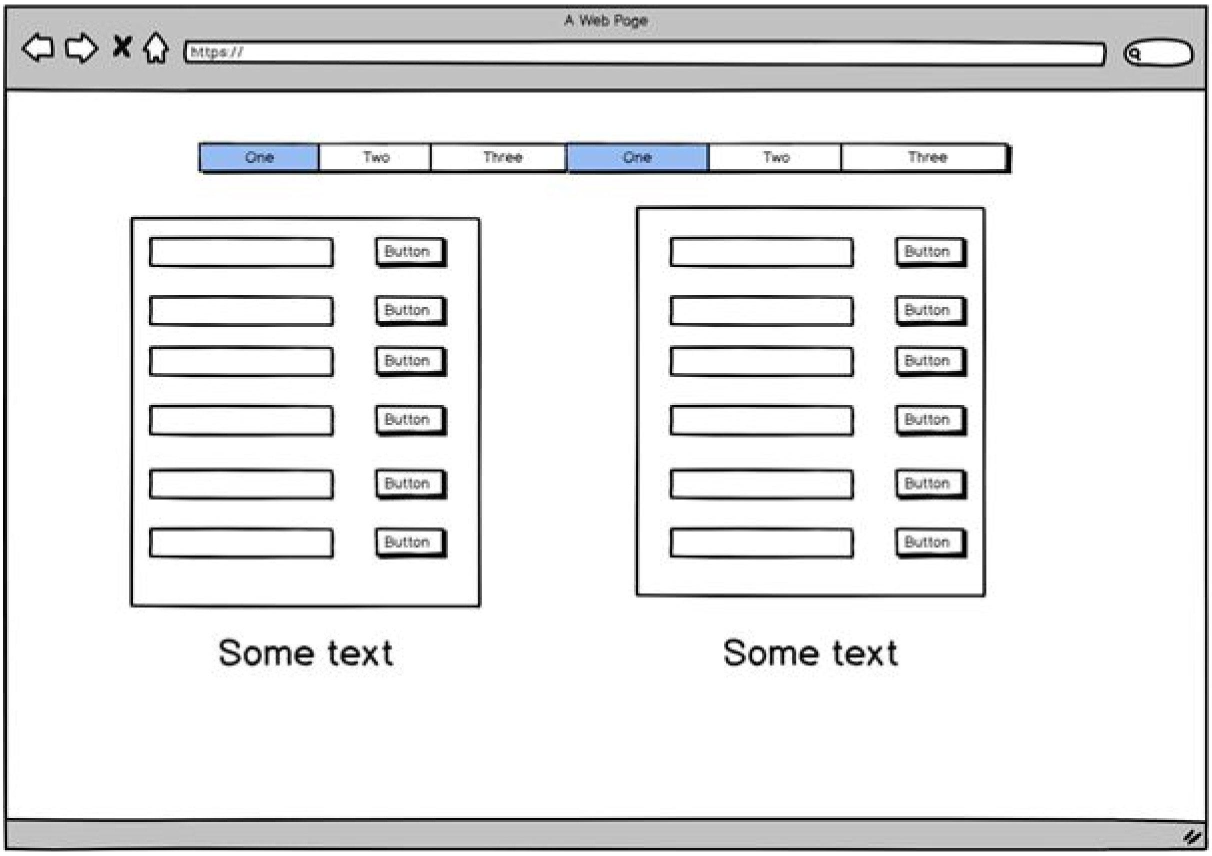


Process: Other than any default pages, coaches can view their team's players from the button on the main page.

SQL Statements:

```
SELECT player_name
FROM Player P, Agent A
WHERE P.agent = A.id
```

5.6 Transfer Page



Inputs: @searchkey, @fee
Process: Directors can search for players and make offer for them on their special page which can be accessed from the main page. Also, directors can see transfer fee of players on the right side of this page

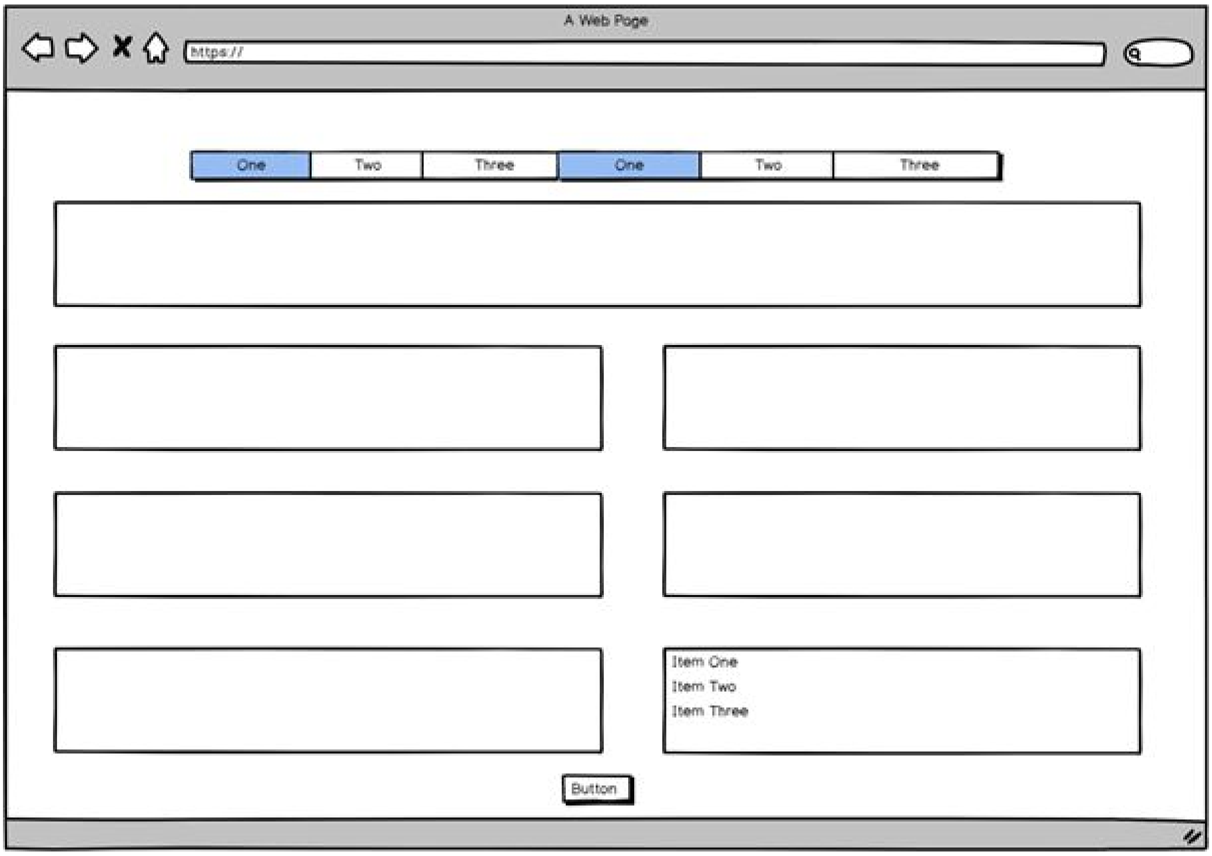
SQL Statements:

```
SELECT Player_name, transfer_fee
FROM Player NATURAL JOIN Offer
ORDER BY transfer_fee
```

5.7 Administrator User Interface Design

Other than any other pages, administrators can access to the set team, player, league and cup pages. Only they can access to other users' pages and manage their accounts.

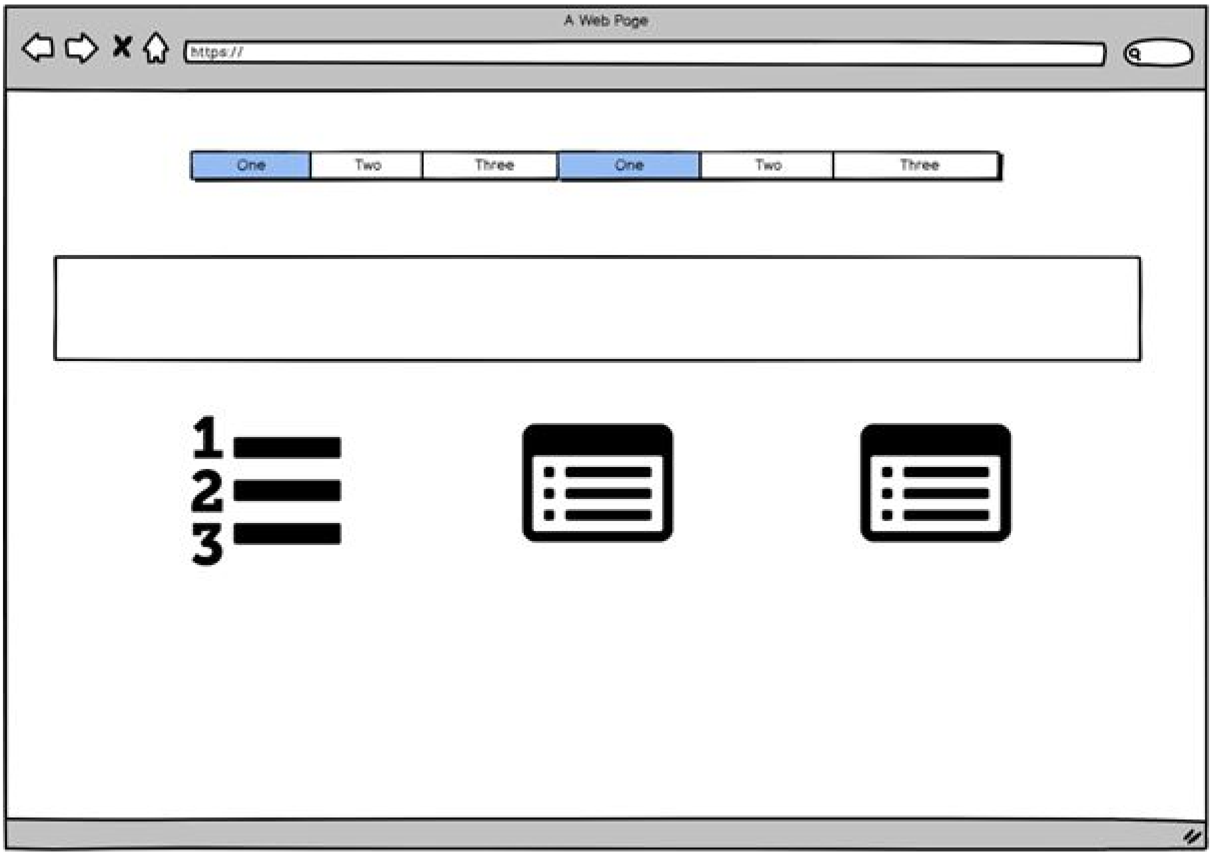
5.7.1 Set Team/Player Page



Inputs: @string

Process: This page looks like the normal page but the attributes are in an input box which administrator can change.

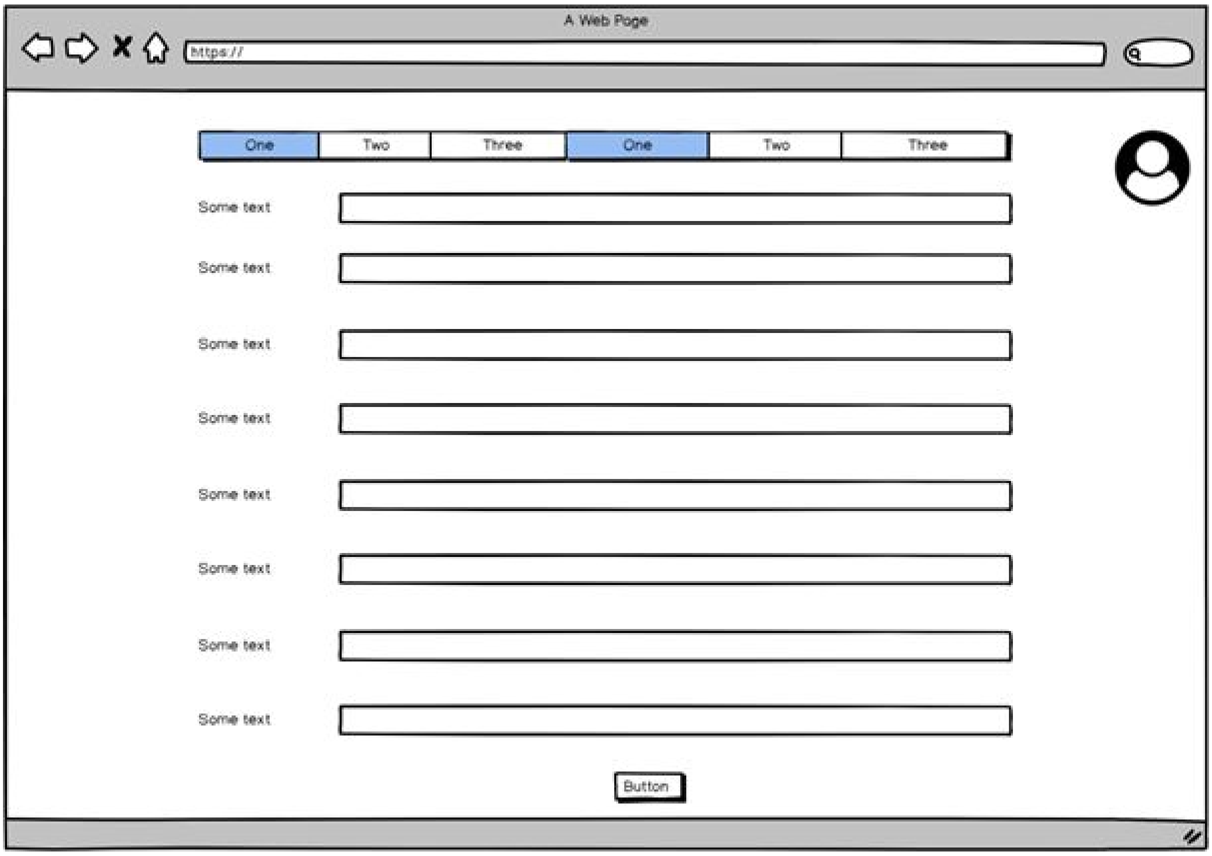
5.7.2 Set League/Cup Page



Inputs: @string

Process: This page looks like the normal page but the attributes are in an input box which administrator can change.

5.7.3 Manage Page



Inputs: @string

Process: This page looks like the normal page but when admin writes a value to username and it matches with a user, rest of the attributes are filled and admin can manage them.

6. Advanced Database Components

6.1 View

6.1.1 Top 10 Most richest football clubs

```
CREATE VIEW clubs_top10(club_name, total_amount, wage_budget, transfer_budget)
AS SELECT(name, total_amount, wage_budget, transfer_budget)
FROM club
SELECT club_id,SUM(wage_budget) + SUM(transfer_budget) AS total amount
FROM club
GROUP BY(club_id)
ORDER BY total_amount DESC
LIMIT 10
```

6.2 Reports

6.2.1 Injury Reports

Doctors, coaches and directors can view the injured players. Doctor can decide which patient to treat first and coach can put aside the player for a time or change the captain.

6.2.2 Advised Players Report

Agents can view the player who they advise and see which team, league and cup they are in. They can decide to advise them to make carrier choices.

6.3 Triggers

- When a team buys or sells a player, its budget will decrease or increase accordingly.
- When a team’s budget change, feasibility of the ongoing transfers will be checked.
- When players change their teams, their information will change accordingly.

6.4 Constraints

- Budget of team cannot be less than the sum of it’s all employee’s salaries.
- Offers cannot overflow the team’s budget.

7. Implementation Plan

For the user interface of Football Database System, we plan to use CSS, HTML and CSS. In order to manage data easily, we are using MySQL.

8. Website

https://sekin72.github.io/