CS353 Spring 2020 Semester Project

Final Report

Group 14 - Online Scout

Özge Yaşayan   21704154

Rabia Nur Önal   21702641

Fatih Çakır   21601370

Celal Bayraktar 21501801

Table of Contents

1. Description of the Project

This project is a web-based application for mainly five different user types; football clubs, scout agencies, football players' agents, and journalists. Each user can use different functionalities of the system. Clubs can communicate with scout agencies to find a player with specifications, receive reports about their requests, as well as making transfer offers to other clubs regarding specific players. Namely, after scouts find appropriate players, clubs can make an offer using the interface.

Scout agencies can assign available scouts in their agencies to clubs. Scouts will join their scout agency page with activation key and after that they can see their assigned tasks through the website. Scouts can send their reports to clubs using the system.

Agents can recommend players they represent to a club. Clubs can make requests to agencies in order to find a player that fits their requirements.

Journalists will be notified if they subscribed to the teams they want to be notified about. Notifications will be about subscribed teams' interests. For example, if a team sends a request for a certain position to scout agency, journalists will be notified about that team is looking for a player for that position.

By using this website, the clubs and agents will save time, agencies will be managed easily and journalists can obtain information directly from clubs without putting in a lot of effort.
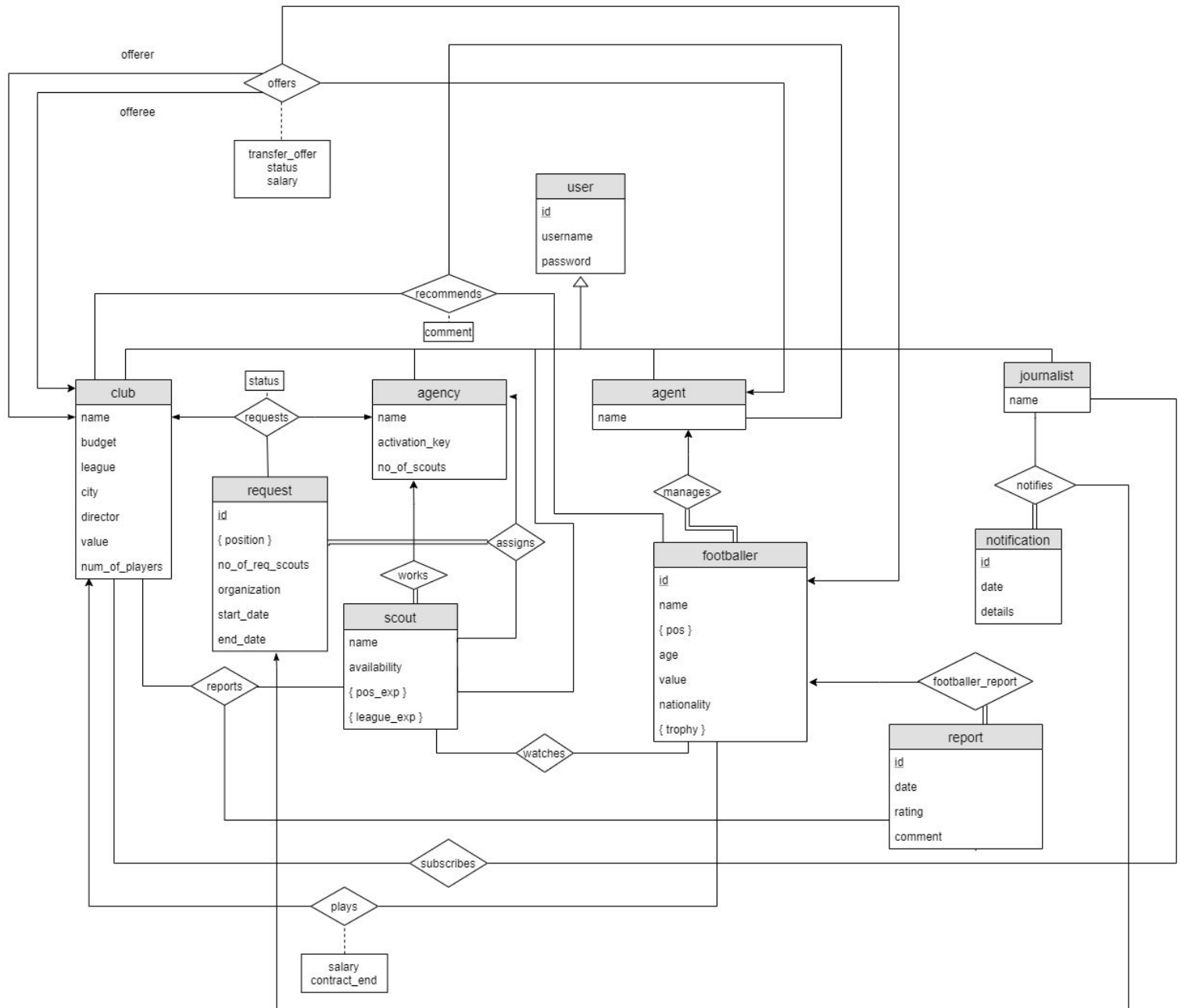
2. Final ER Diagram

Figure 1: Final version of the ER Diagram

3. Final List of Tables

3.1. User

Relational Model: user (<u>id</u>, username, password)

Primary Key: {(id)}

3.2. Journalist

Relational Model: journalist (<u>id</u>, name)

Primary Key: {(id)}

Foreign Keys: (id) as id in user table

3.3. Agent

Relational Model: agent (<u>id</u>, name)

Primary Key: {(id)}

Foreign Key: (id) as id in user table

3.4. Agency

Relational Model: agency (<u>id</u>, name, activation_key, num_of_scouts)

Primary Key: {(id)}

Foreign Keys: (id) as id in user table

3.5. Club

Relational Model: club (<u>id</u>, name, budget, league, city, director, value, num_of_players)

Primary Key: {(id)}

Foreign Keys: (id) as id in user table

3.6. Request

Relational Model: request (<u>id</u>, no_of_req_scouts, organization, start_date, end_date)

Primary Key: {(id)}

3.6.1. Request Positions

Relational Model: request_positions (<u>id</u>, <u>position</u>)

Primary Key: {(id, position)}

Foreign Keys: (id) as id in request table

3.7. Scout

Relational Model: scout (<u>id</u>, name, availability, agency_id)

Primary Key: {(id)}

Foreign Keys: (id) as id in user table, (agency_id) as id in agency table

3.7.1. Scout Position Experience

Relational Model: scout_position_exp (<u>id</u>, <u>position</u>)

Primary Key: {(id, position)}

Foreign Keys: (id) as id in scout table

3.7.2. Scout League Experience

Relational Model: scout_league_exp (<u>id</u>, <u>league</u>)

Primary Key: {(id, league)}

Foreign Keys: (id) as id in scout table

3.8. Footballer

Relational Model: footballer (<u>id</u>, name, age, value, nationality, club_id, agent_id)

Primary Key: {(id)}

Foreign Keys: (club_id) as id in club table, (agent_id) as id in agent table

3.8.1. Footballer Positions

Relational Model: footballer_positions (<u>id</u>, <u>position</u>)

Primary Key: {(id, position)}

Foreign Keys: (id) as id in footballer table

3.8.2. Footballer Trophies

Relational Model: footballer_trophy (<u>id</u>, <u>trophy</u>)

Primary Key: {(id, trophy)}

Foreign Keys: (id) as id in footballer table

3.9. Notification

Relational Model: notification (<u>id</u>, date, details)

Primary Key: {(id)}

3.10. Report

Relational Model: report (<u>id</u>, date, rating, comment, footballer_id)

Primary Key: {(id)}

Foreign Keys: (footballer_id) as id in footballer table

3.11. Subscribes

Relational Model: subscribes (<u>journalist_id</u>, <u>club_id</u>)

Primary Key: {(journalist_id, club_id)}

Foreign Keys: (journalist_id) as id in journalist table, (club_id) as id in club table

3.12. Watches

Relational Model: watches (<u>scout_id</u>, <u>footballer_id</u>)

Primary Key: {(scout_id, footballer_id)}

Foreign Keys: (scout_id) as id in scout table, (footballer_id) as id in footballer table

3.13. Assigns

Relational Model: assigns (<u>agency_id</u>, <u>scout_id</u>)

Primary Key: {(agency_id, scout_id)}

Foreign Keys: (agency_id) as id in agency, (scout_id) as id in scout

3.14. Requests

Relational Model: requests (<u>request_id</u>, club_id, agency_id, status)

Primary Key: {(request_id)}

Foreign Keys: (club_id) as id in club, (agency_id) as id in agency, (request_id) as id in request

3.15. Recommends

Relational Model: recommends (<u>footballer_id</u>, <u>agent_id</u>, <u>club_id</u>, comment)

Primary Key: {(footballer_id, agent_id, club_id)}

Foreign Keys: (footballer_id) as id in footballer, (agent_id) as id in agent, (club_id) as id in club

3.16. Reports

Relational Model: reports (<u>club_id</u>, <u>scout_id</u>, <u>report_id</u>)

Primary Key: {(club_id, scout_id, report_id)}

Foreign Keys: (club_id) as id in club, (scout_id) as id in scout, (report_id) as id in report

3.17. Notifies

Relational Model: notifies (<u>journalist_id</u>, <u>notification_id</u>, <u>request_id</u>)

Primary Key: {(journalist_id, notification_id, request_id)}

Foreign Keys: (journalist_id) as id in journalist, (notification_id) as id in notification, (request_id) as id in request

3.18. Offers

Relational Model: offers (<u>offerer_id</u>, <u>offeree_id</u>, <u>footballer_id</u>, <u>transfer_offer</u>, agent_id, status, salary)

Primary Key: {(offerer_id, offeree_id, footballer_id, transfer_offer)}

Foreign Keys: (offerer_id) as id in club, (offeree_id) as id in club, (footballer_id) as id in footballer, (agent_id) as id in agent

4. Implementation Details

We used MySQL as the relational database management system.In order to connect to the database and create the tables, we used Java's sql package [1]. To be able to use this, we needed the MySQL Connector for Java. This worked fine for localhost however we needed to change localhost to a private IP address so that we could host it on one of our computers. For this, one of us (the host) added an incoming rule to her firewall for port 3306 which we were going to use. However, even though this worked on her computer, it did not work for others, they were getting an error saying that the packages were being delivered but there was no response so a timeout was occuring. They could not reach the database on her computer via her private IP address.

Therefore, we looked for other platforms that would let us host a database remotely. We found Heroku [2]. Heroku is a cloud based application platform. We included the ClearDB add-on in order to be able to host a MySQL database from there [3]. Through this add-on, we were given access to a username and a password in order to be able to connect to the database remotely. After we had access to the database, we populated the tables and manipulated them through MySQL Workbench 8.0. Another problem we faced was that the remote database did not allow us to add any triggers or stored procedures and we could not solve this problem because we did not have the correct type of privilege to make that change to the database. Instead, after the button was clicked by a club to make a request to an agency, the same operations were performed on the frontend side.

For the frontend, we decided to use HTML, CSS, PHP and occasional JavaScript. Before we had access to a remote database, we used XAMPP [4] which came with phpmyadmin that let us experiment with PHP on our local computers. However, after we had the remote database

access, we didn't need it anymore. We coded the frontend in PHP, HTML and CSS on our preferred IDE's and used GitHub as a version control tool. JavaScript was needed for Bootstrap components, to have a more visually appealing website. In order to host the website, we used InfinityFree [5] which helped us host it. This service gave us the username, password and the hostname to be able to upload our files (PHP, HTML files etc.) to the website through FileZilla.

5. Work Allocation for Functionalities

Rabia Nur Önal:
- View requests as agency.
- Accept/reject a request.
- Assign scouts to a request.
- Subscribe to clubs as journalist.
- Profile pages.

Celal Bayraktar:
- Login/logout.
- Create report as scout.
- Send request to agency as club.
- Scout's watchlist.
- See notifications as journalist.

Fatih Çakır:
- Sign up for users (scout, journalist, agent and agency).
- Signup for pre registered users (footballer, club).
- See offers as agent, acceptance, rejection.
- See footballers as agent, releasing.
- See submitted offers, editing and canceling as club.
- Send offer to agent as club.

Özge Yaşayan:
- See tasks as scout.
- See reports as club.
- See recommendations as club.
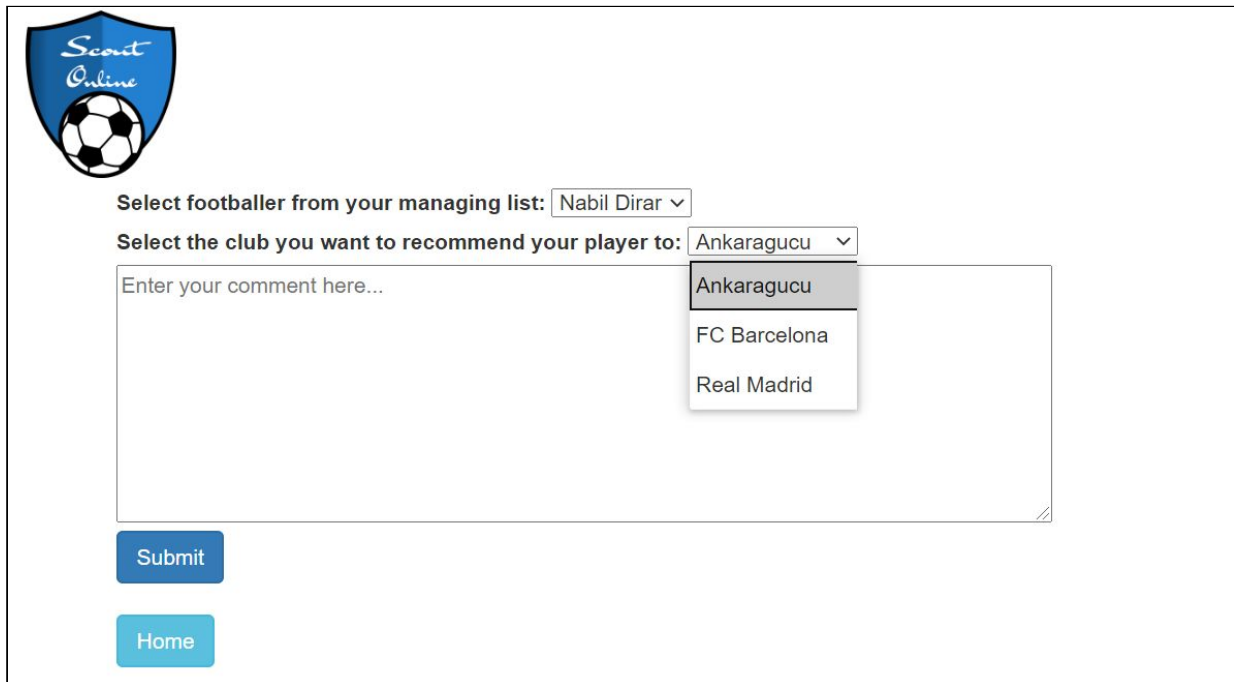
- Make a transfer offer as club.
- Recommend a player as agent.
- See requests as agency.

6. Advanced Database Components

- As a complex SQL query we used

```
SELECT distinct club.name FROM footballer, plays, club
where plays.club_id = club.id and plays.footballer_id = footballer.id
and club.id not in (select plays.club_id from footballer, manages, plays
        where manages.footballer_id = footballer.id
        and  plays.footballer_id = footballer.id
        and manages.agent_id = '$aid')
        order by club.name;
```

Where $aid refers to the agent's id that is signed in. This query finds the names of the clubs except the club that the agent's footballer plays for.

Figure 2: Recommendation of a player

In Figure 2, you can see that the agent of Nabil Dirar, who is a player of Fenerbahçe SK, does not see Fenerbahçe SK as an option to recommend Nabil Dirar to. The query mentioned above is what made that possible.

● As a trigger for journalist table, we planned to use:

```
CREATE TRIGGER notify_journalist
AFTER INSERT
ON requests FOR EACH ROW
BEGIN
    insert into notification(date)
    values(curdate());

    insert into notifies(notification_id, journalist_id, request_id)
    select (select max(id) from notification), journalist.id, NEW.request_id
    from journalist, subscribes
    where subscribes.journalist_id = journalist.id
    and subscribes.club_id = NEW.club_id;
END
```

This basically sends a notification to a journalist if a new request is made by a club that the journalist is subscribed to. However, we used a free remote database that did not allow us to add triggers so we instead did the same operations when a certain button was clicked. You can see the notifications page in Figure 3.

Figure 3: Notifications of a journalist

● We have used foreign key constraints for many of our tables which can be seen in Section 3.

7. User's Manual

Online Scout is a website that brings different people of the football community together for them to be able to communicate. It also contains information about clubs and footballers for public use. We have five types of users that have accounts on the webpage and footballers. All users can login to their accounts via their usernames and passwords they have chosen while signing up.
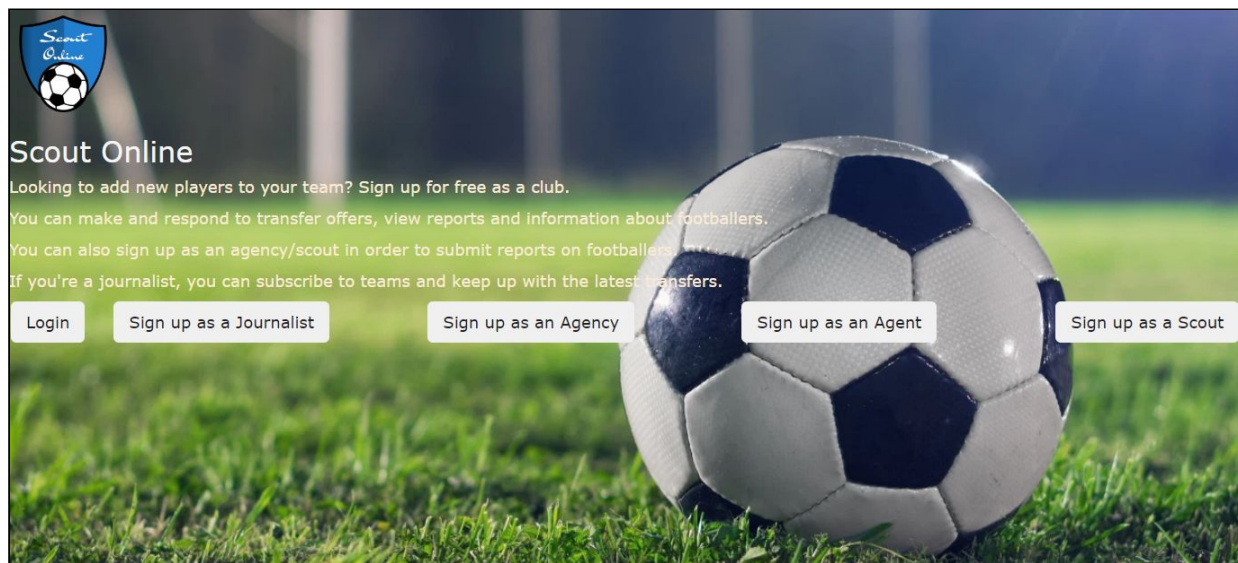
Figure 4: Welcoming page
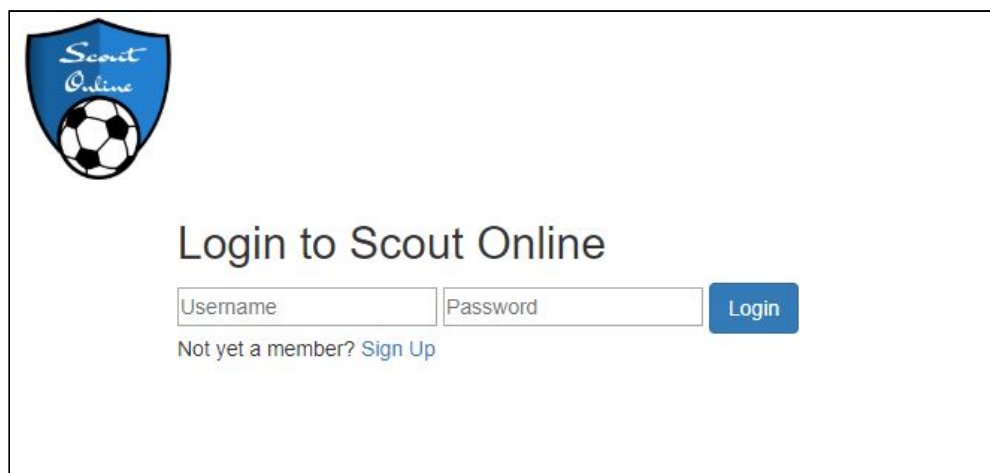
Figure 5: Login

7.1. Club

Clubs do no sign up to the webpage. Since their information already exists on the system and database, appropriate accounts are created by the administration. If they wish to use the system, they can contact with Scout Online and receive their usernames and passwords.

After login clubs can see their options. They can log out or see their profiles.
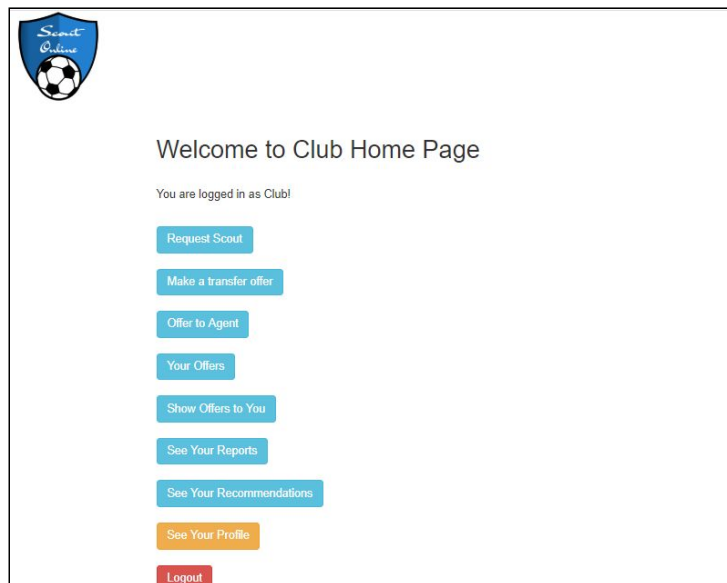
Figure 6: Club homepage
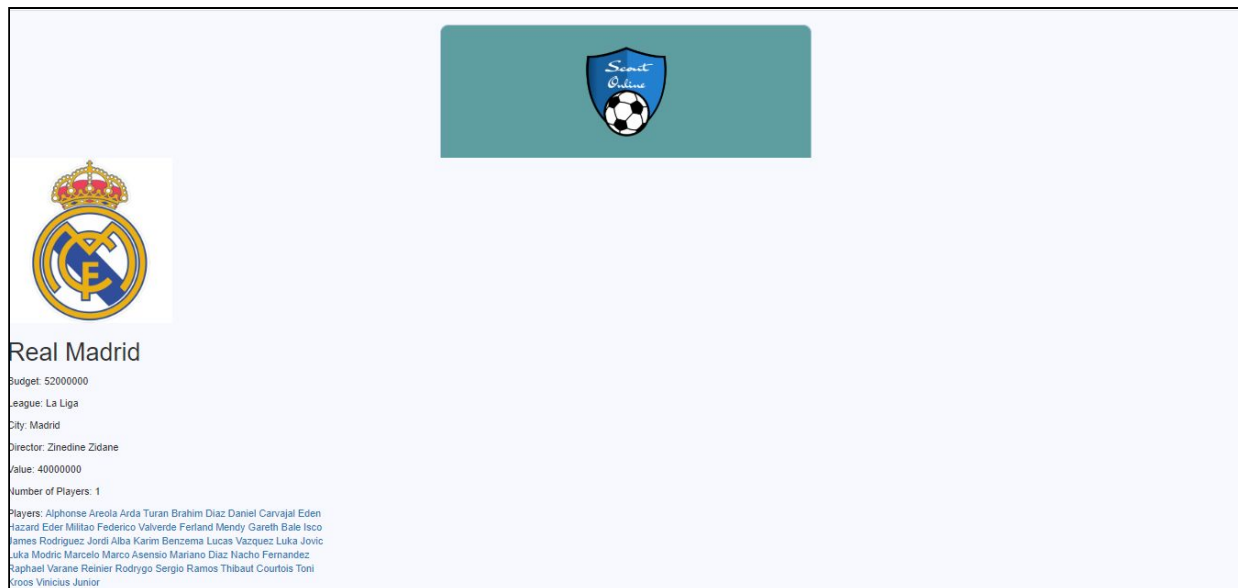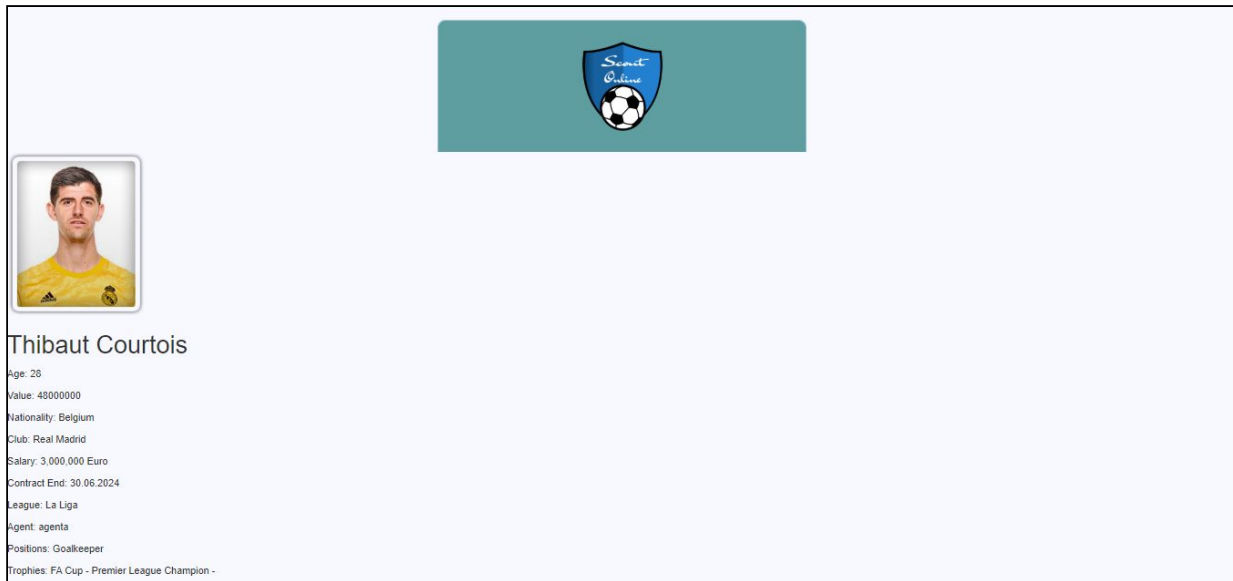
Figure 7: Club profile
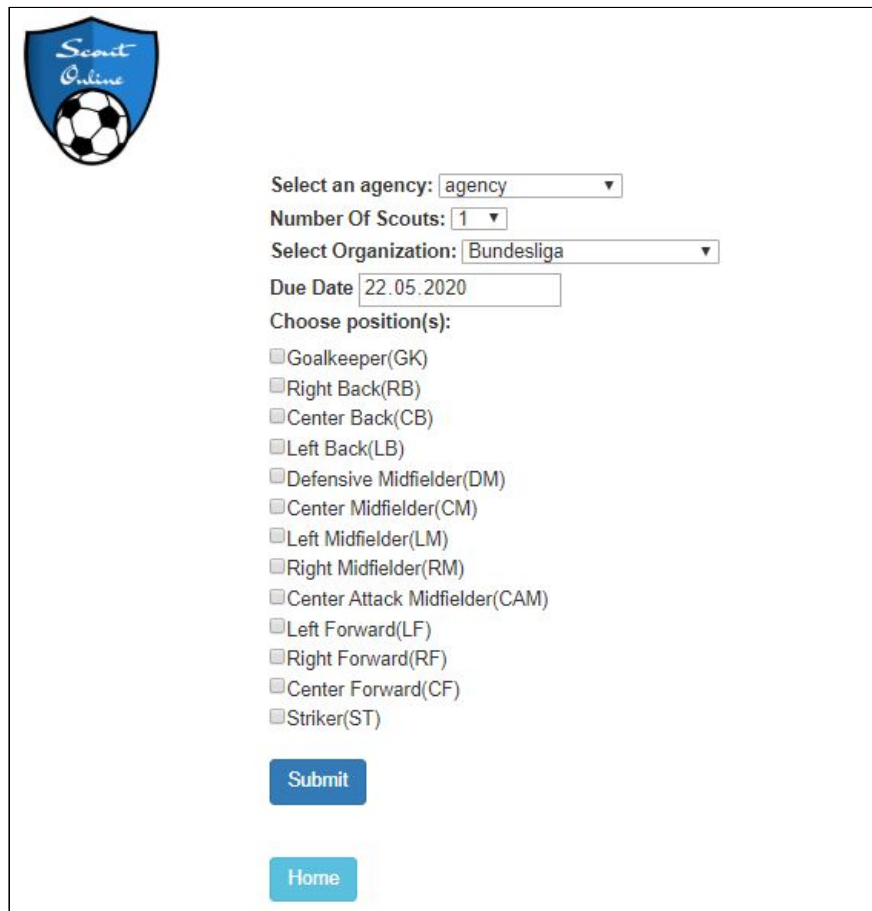
By clicking on the name of their footballers, they can also see the players' profiles.

Figure 8: Footballer profile

From the home page, clubs can choose to request scouts from an agency. From the opened page, they can select an agency, required number of scouts, organization and positions by specifying a deadline for getting the reported information back.

Figure 9: Making a request

They can make a transfer offer to another club by choosing the footballer and writing the offer price.

Figure 10: Making a transfer offer

They can see the offers they previously made and edit the offered price by writing a new price or cancel it if they wish.

Figure 11: View previously made offers

Figure 12: Edit a previous offer

They can also see the offers made for their footballers and and accept or reject them.

Figure 13: View transfer offers made for itself

Figure 14: Respond to an offer

They can see the offers that are accepted by the other clubs and, they can make an offer to the footballer's agent for the salary. If the agent accepts the salary offer, transfer is made and the club can see the footballer in the list on profile.

Figure 15: View transfer offers accepted by clubs

Figure 16: Make a salary offer for transfer offers accepted by club

Clubs can see the list of reports about footballers, written by the scouts according to their request, view them and go make a transfer offer to the clubs for the footballers on the reports.

Footballer	Scout	Rating	Comment	Date	
Sergio Ramos	ali	7.3	asdfgh jkqwe rtyuzx ...	25.05.2020	○
Luka Modric	Fatih Çakır	9.1	good player...	2020-05-22	○
Arda Turan	Fatih Çakır	6.7	eh iste...	2020-05-22	○
Jordi Alba	test sout	7.9	very good player...	2020-05-22	○
Altay Bayindir	scout5	7.1	ortalama bir kaleci,...	2020-05-21	○
Raphael Varane	ali	8.4	asdfgh jkqwe rtyuzx ...	20.05.2020	○

View

Transfer Offer Page

Home

Figure 17: View reports list
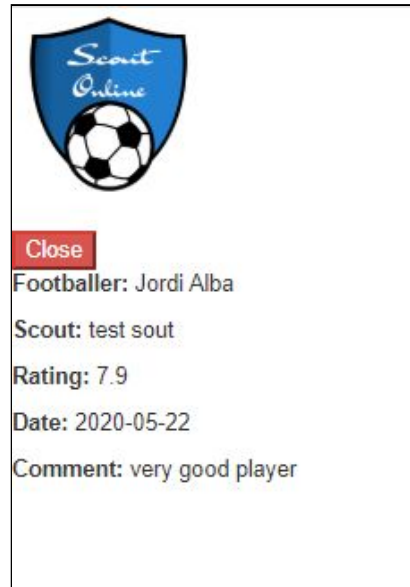
Figure 18: View a reports up close
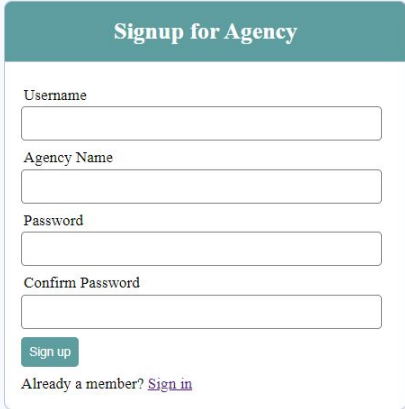
Clubs can view the recommendations made to them by agents.

Footballer	Comment	Agent
Arda Turan	good player	Rabia Önal
Atila Turan	öneriyorum	Rabia Önal
Jordi Alba	recommendation comment	Test agent

Home

Figure 19: View recommendations

7.2. Agency

To signup, agencies must enter their names and choose a username with a password.

Figure 20: Agency signup

After login, agencies see their homepage that contains buttons for logout, see their scouts and see their requests.
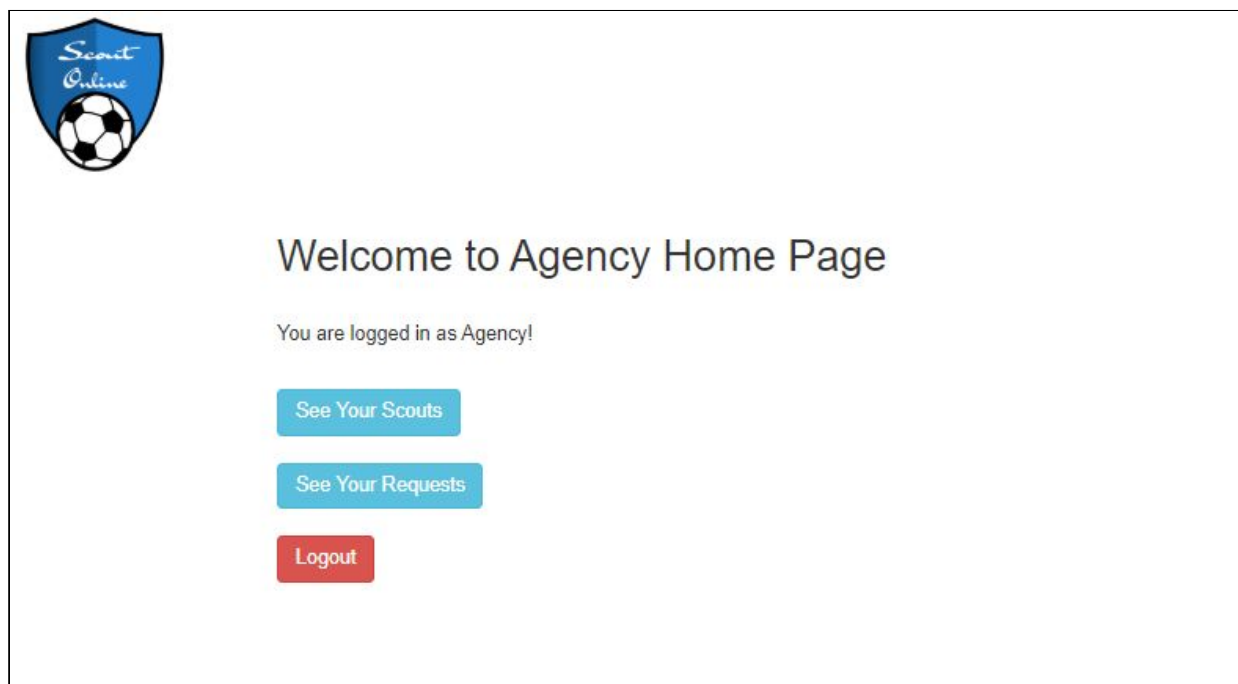
Figure 21: Agency homepage

They can view the scout that work for them along with their experiences and availability status.

Figure 22: View scouts

They can also view the requests from clubs and choose to accept or reject them.

Figure 23: View requests

If they choose to accept the offer they should assign some scouts. They have to assign as much as the number that is specified by the club, if they try to assign a different number of a scout or if there is not enough number of scouts available currently, system will alert the agency and not let the assignment to be made.

Figure 24: Assign scouts
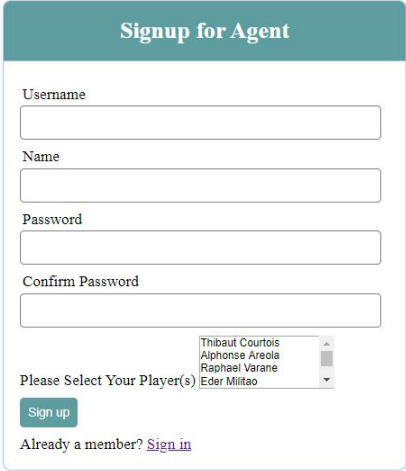
Figure 25: Assign scout error message

7.3. Agent
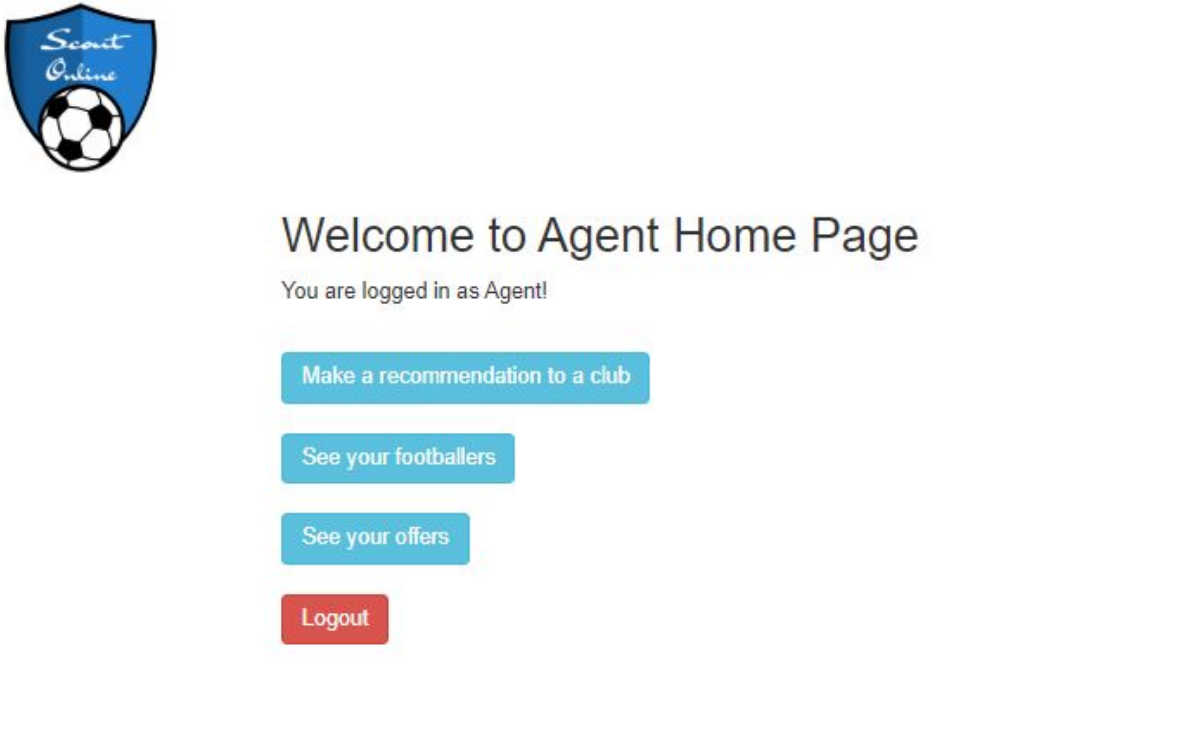
To sign up, agents should enter their name, select a username and a password, also they should choose the footballers they manage.

Figure 26: Agent signup

After login agents see their homepage with the options to logout, make a recommendation to a club, see the footballers they manage and see the offers made for their footballers.

Figure 27: Agent homepage

To make a recommendation, they should choose a footballer from their management list, the club to recommend and enter some comments about the player.

Figure 28: Make a recommendation

They can see the list of footballers they manage and they can disengage any of them at any time.

Figure 29: View footballers being managed

Agents can see the salary offers made for their footballers and respond to them. If they accept the offer, the footballer will be transferred to offerer club.

Figure 30: View salary offers
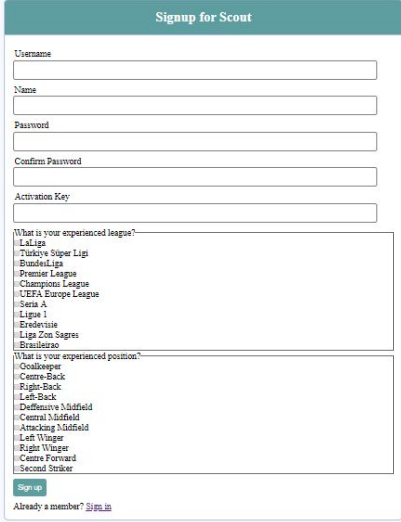
Figure 31: Respond to salary offer

7.4. Scout

To sign up, scout s should enter their names, username, password. The should also enter the activation key of the agency they will work for and specify which leagues and positions they are experienced in.

Figure 32: Scout sign up

After login scouts see the options to logout, create report, add a player to his watchlist and see tasks on the homepage.
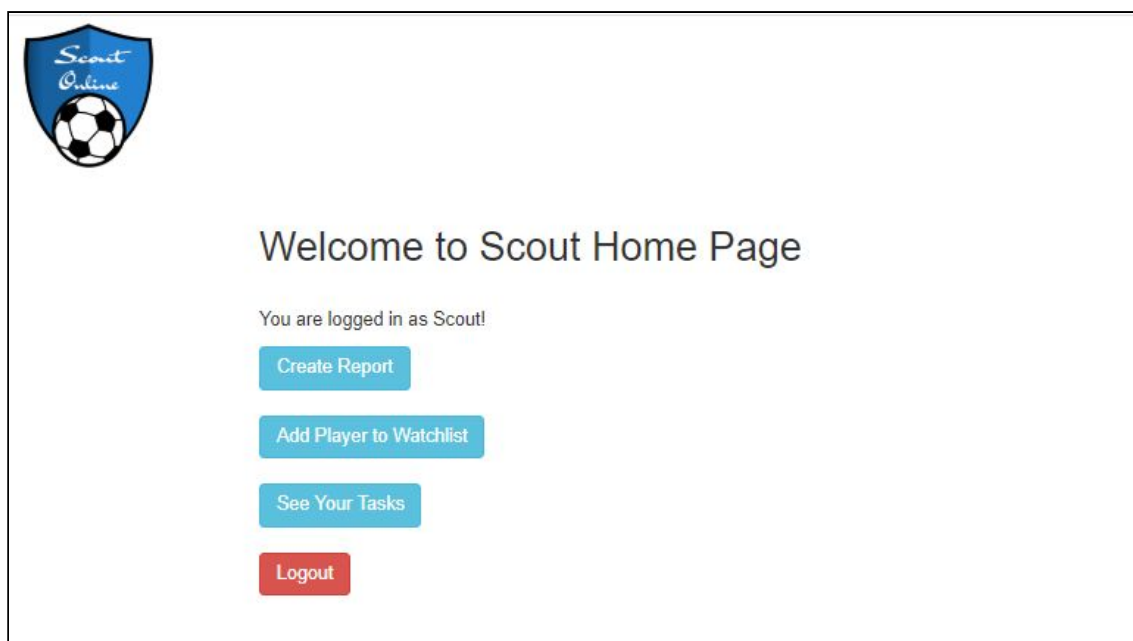

Figure 33: Scout homepage

When agencies assign scouts to club requests, scouts can see their task along with the deadline.

Figure 34: View tasks

Scouts can add footballers to their watchlists by selecting and submitting them one by one. When they are done they can click on the button "I am done" and return to their homepage.

Figure 35: Add footballer to watchlist

When scouts are done watching a footballer, they can submit a report about him by rating him out of 10 and adding some comments to the club that made the request.
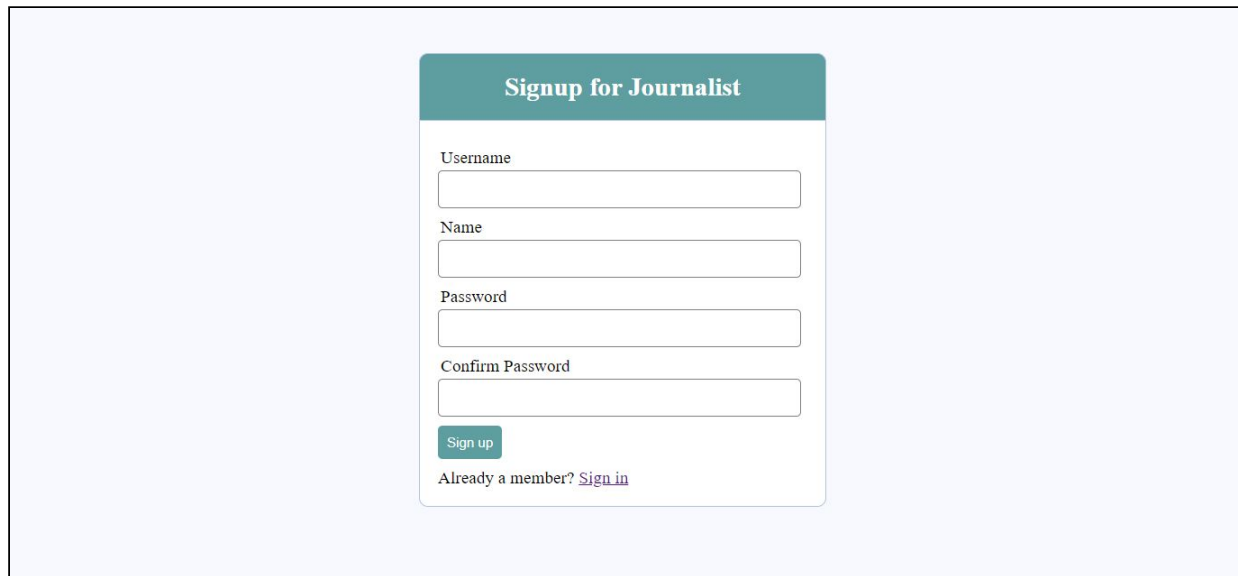

Figure 36: Submit a report

7.5. Journalist

To sign up, journalists enter their name and select a username and a password.

Figure 37: Journalist sign up

After login, journalists see the options to logout, subscribe and see notifications options on their homepage.
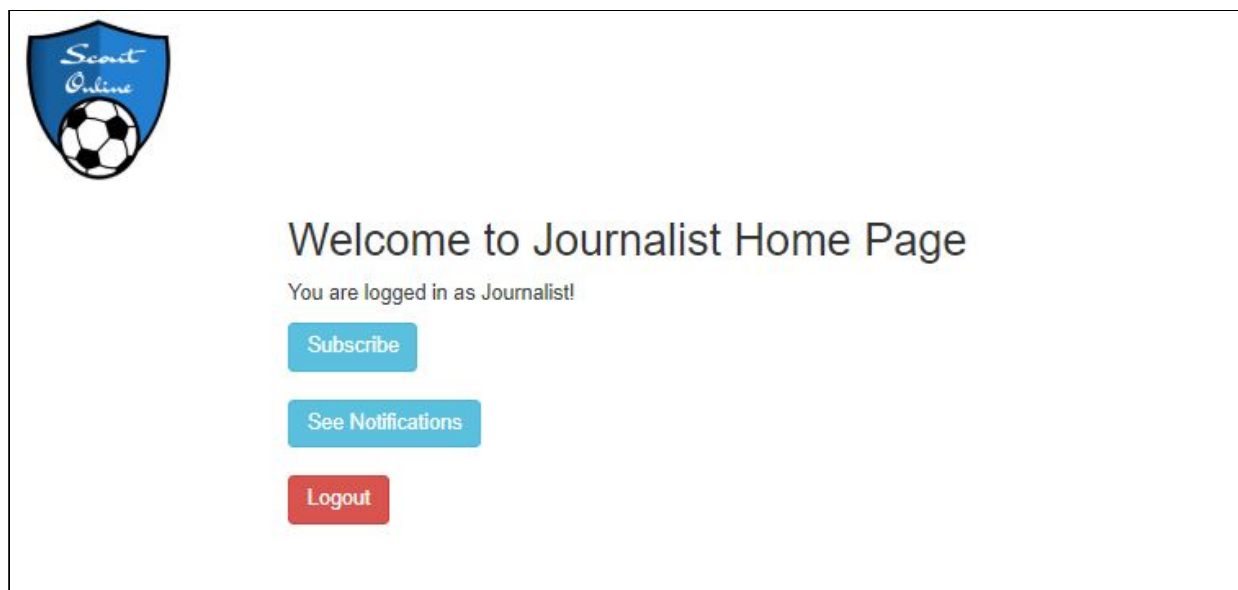

Figure 38: Journalist homepage

The can subscribe to a club by choosing one from the list and agreeing the terms and conditions. Leaving either space empty will alert the user and not perform subscription.
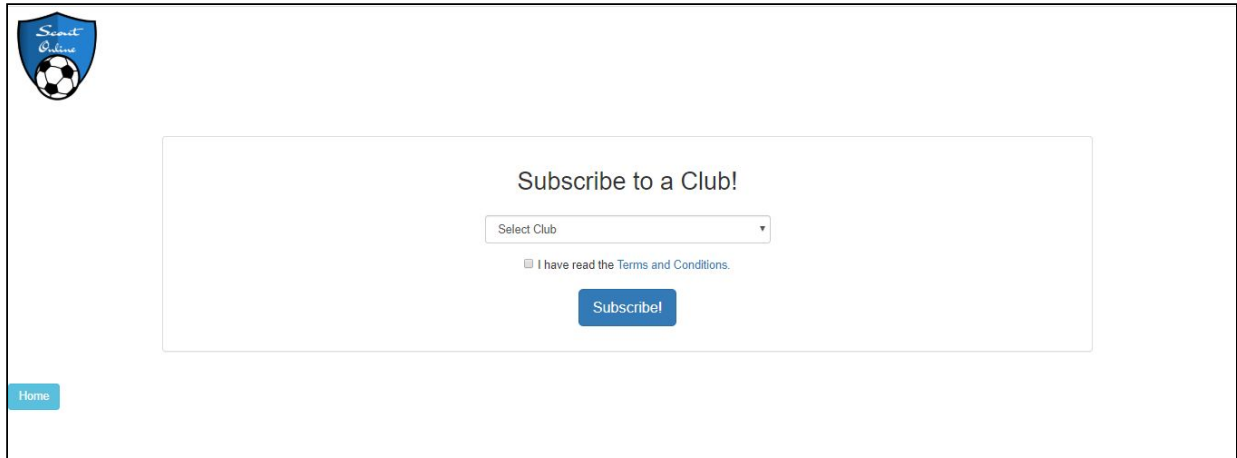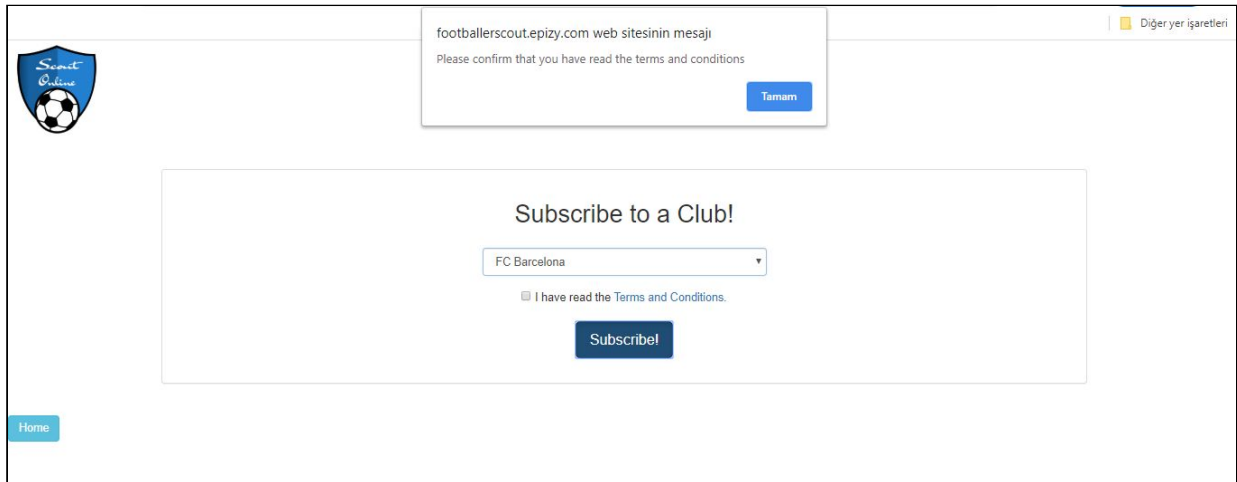

Figure 39: Subscribe to a club

Figure 40: subscribe to a club error message

Journalists can see some notifications when the clubs they subscribed made a request.

Notification No.	Club	Positions	Organization	Date
1	Real Madrid	cb, lb	La Liga	2020-05-21
2	Real Madrid	dm, cm, lm	Serie A	2020-05-22
3	Real Madrid	lb	La Liga	2020-05-22
4	Real Madrid	dm, st	La Liga	2020-05-22

Figure 41: View notifications

8. Website and GitHub Repository

You can find our website on http://footballerscout.epizy.com/ and our GitHub repository on https://github.com/ozgey99/CS353 .

9. References

[1] https://docs.oracle.com/javase/8/docs/api/java/sql/package-summary.html

[2] https://www.heroku.com

[3] https://elements.heroku.com/addons/cleardb

[4] https://www.apachefriends.org/index.html

[5] https://app.infinityfree.net/