Web Applications A.Y. 2023-2024

Homework 1 – Server-side Design and Development

Master Degree in Computer Engineering

Deadline: 29 April, 2024

Group Acronym	**DAM**	
Last Name	**First Name**	**Badge Number**
Basaglia	Alberto	2119289
Bruttomesso	Andrea	2120933
Corrò	Alessandro	2125034
Popovic	Milica	2119069
Seghetto	Davide	2122548
Stocco	Andrea	2108885

1 Objectives

The primary objective of our web application is to provide a comprehensive platform for users to access and explore football (and its variants like futsal, 8-a-side football ecc.) tournaments effortlessly. Tournaments consist of a single group stage with a team ranking. Our platform aims to simplify the process of tournament management, allowing organizers to effortlessly set up tournaments, schedule matches, and track results, whether they're organizing a friendly neighborhood tournament or a competitive league.
For users, our platform provides easy access to tournament information, including match schedules, team standings, and top scorers' lists.
Key features of our platform include:

- **Tournament Creation and Management:** Organizers can easily create tournaments, define match schedules, manage team registrations, input match results, including goals scored, and update match statuses in real-time.
- **Comprehensive Tournament Information**: Users have access to detailed tournament information, such as match schedules, team standings, and individual player statistics.

By addressing these objectives, our goal is to deliver a robust and feature-rich web application that enhances the overall experience of football enthusiasts, providing them with a centralized platform to engage with their favorite tournaments, teams, and players.

2 Main Functionalities

Every user, even if not registered on the site, can view past and present tournaments with their related match schedules, team standings, and top scorers' list. Once the user register on the site and login, he gets the ability to create tournaments and manage them (changing the name, the deadline for registering a team, the logo, the starting date etc.) until their conclusion. The admin of a tournament can also delete it if needed.
More in-depth, the main functionalities are the following:

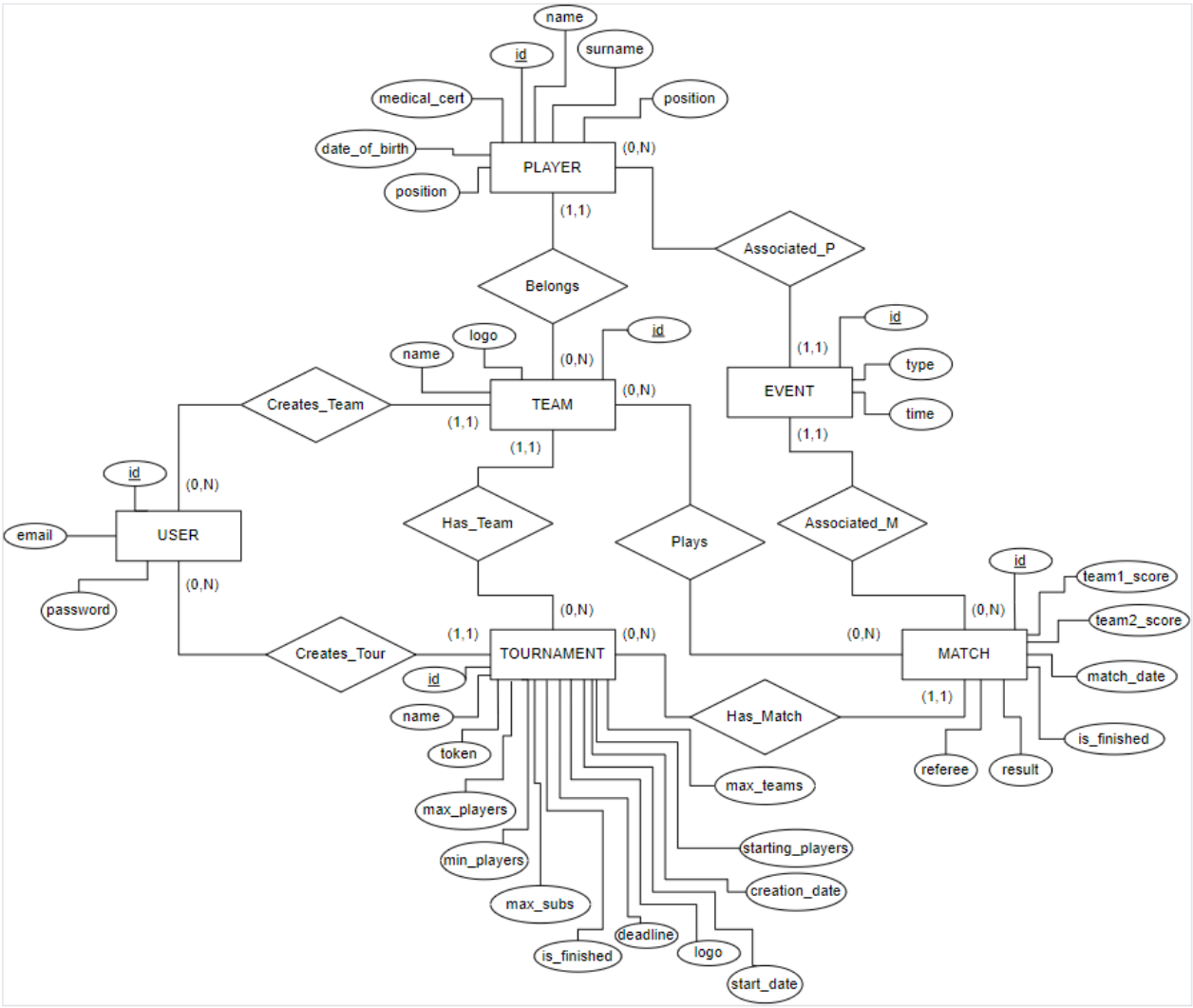
- **Creation of a new tournament:** This functionality offers an opportunity to everyone to create a football tournament. On the main page of the web app, it is possible to create an account which is required for the tournament creation. After successfully completing the sign up proccess, there are other steps which should be taken such as providing basic data about the tournament (name, logo, deadline for the team creation, the starting players, max number of players, max number of teams etc.).
- **Edit/deletion of a tournament:** Only the tournament creator is allowed to edit (name, logo, starting date, etc.) or delete the tournament.
- **Creation/edit/deletion of a team for the tournament:** Once the user has logged in, he can select the tournament of his interest and enroll his team by providing all the specifications needed, like the team name, the logo, etc. Only the creator of the team and the admin of the tournament are allowed to edit the team info or delete the team if needed.
- **Addition/edit/deletion of players:** Only the team creator and the tournament admin are allowed to add, edit, or delete players to/from the team.
- **Creation of the draw** Each tournament creator has the ability to make a draw for the tournament. There are two options for making a draw:
    1. After the deadline for the teams creation is met, a job is automatically started for creating a draw for the tournament, which is going to be visible on the page.
    2. If the admin of the tournament deems the tournament is ready to start, he can manually create the games.

- **Updating the info of a tournament's matches:** Only the admin of the tournament is able to insert the result of a match that has been played, with also the main events like the scorers and the sanctions (yellow/red cards).
- **Follow of the tournament's matches:** Another feature of the app is displaying results of the tournament matches and current statistics like team standings and top scorers' ranking. That feature is available for every active and past tournament, after choosing the particular tournament from the app's main page (the user don't have to be logged in).

User registration and authentication functionalities allow individuals to create accounts or login to the application, enabling them to register teams for participation in tournaments. Administrators have the authority to create, update, and delete tournaments as needed, ensuring flexibility in organizing and managing various soccer events.
The application dynamically updates content such as tournament tables, results, and top scorers in real-time, ensuring that users have access to the latest information and statistics. The user interface is designed to be intuitive and responsive.
By incorporating these main functionalities, our web application aims to deliver a comprehensive and user-friendly platform for soccer enthusiasts to engage with tournaments, teams and players, while also providing administrators the tools for efficient management and organization.

3 Data Logic Layer

3.1 Entity-Relationship Schema



The ER schema contains 6 entities which are going to be described in details in this section.

1. **User:** Entity Type User contains ID (INT), Email (CHAR) and Password (CHAR) for every user. The primary key is ID, while Email is a unique column. Every user can create one or more tournaments as well as one or more teams. Every user can create zero or more teams and tournaments.

2. **Tournament:** Entity Type Tournament contains ID (INT), Name (CHAR), Token (CHAR), Creator_User_ID (INT), Max_Teams (INT), Max_Players (INT), Min_Players (INT), Starting_Players (INT), Max_Substitutions (INT), Deadline (DATETIME), Start_Date (DATETIME), Creation_Date (DATETIME), Logo (CHAR) and Is_Finished (BOOL). The primary key is ID, Name is a unique column while Creator_User_ID is a foreign key constraint to the User table. Every tournament must be created by exactly one user. Logo contains path to the file which represents logo for a particular tournament. Max_Players and Min_Players are maximum and minimum numbers of players for each team in the tournament, while Starting_Players is a number of players in each match of the tournament and defines the type of tournament that is being organized (football, futsal ecc.). Deadline represents the date until which teams can be enrolled. Starting_Date is the date of the

first match, while Creation_Date indicates the date of the creation of the tournament. Is_Finished indicates whether a tournaments is over or not. Each tournament can be created by only one user and has zero or more teams and matches.

3. **Team:** Entity Type Team contains ID (INT), Name (CHAR), Logo (CHAR), Creator_User_ID (INT) and Tournament_ID (INT). The primary key is ID, Name is a unique column while Creator_User_ID is a foreign key constraint to the User table and Tournament_ID is a foreign key constraint to the Tournament table. Every Team must belong to at most one tournament and must be created by a user. This Entity Type can have multiple players and can play multiple matches.

4. **Player:** Entity Type Player contains ID (INT), Name (CHAR), Surname (CHAR), Team_ID (INT), Position (ENUM), Medical_Certificate (CHAR), Date_Of_Birth (DATETIME). The primary key is ID while Team_ID is a foreign key constraint to the Team table. Medical_Certificate contains path to the file which represents Medical Certificate for a particular player. Every player must belong to a team and can be participant in more events.

5. **Match:** Entity Type Match contains ID (INT), Team1_ID (INT), Team2_ID (INT), Tournament_ID (INT), Team1_Score (INT), Team2_Score (INT), Result (CHAR), Referee (CHAR), Match_Date (DATETIME) and Is_Finished (BOOL). The primary key is ID while Team1_ID and Team2_ID are foreign keys constraint to the respective teams. Tournament_ID is a foreign key constraint to the respective tournament. Every match must belong to only one tournament, have always two teams and can have multiple event associated to it.

6. **Event:** Entity Type Event contains ID (INT), Match_ID (INT), Player_ID (INT), Type (Enum) and Time (DATETIME). The primary key is ID while Match_ID is a foreign key constraint to the Match table and Player_ID is a foreign key constraint to the Player table. Every event belongs to one match and has one Player who made the event. Type of event can be goal or sanction.

3.2 Other Information

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

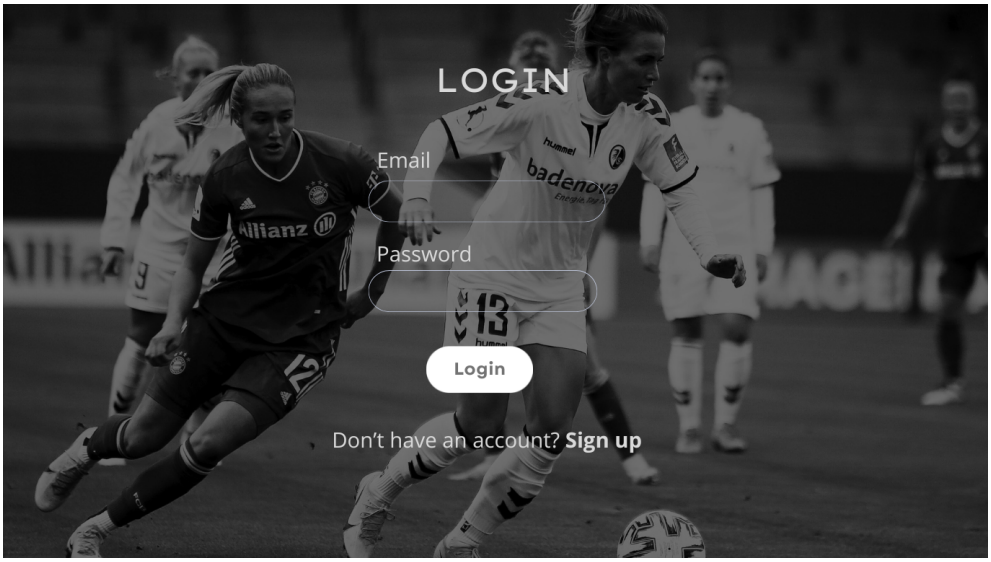
4 Presentation Logic Layer

There will be present approximately 10 pages. The default one will be a page showing all the past and active tournaments, that every user can check even if not logged in. A page for the login and the sign up. Once logged in, the user will be able to see also the past and active tournaments that he created. Finally, there will be the pages dedicated to displaying the information and the ones dedicated for the creation of the teams and the tournaments and for the updating of the matches.

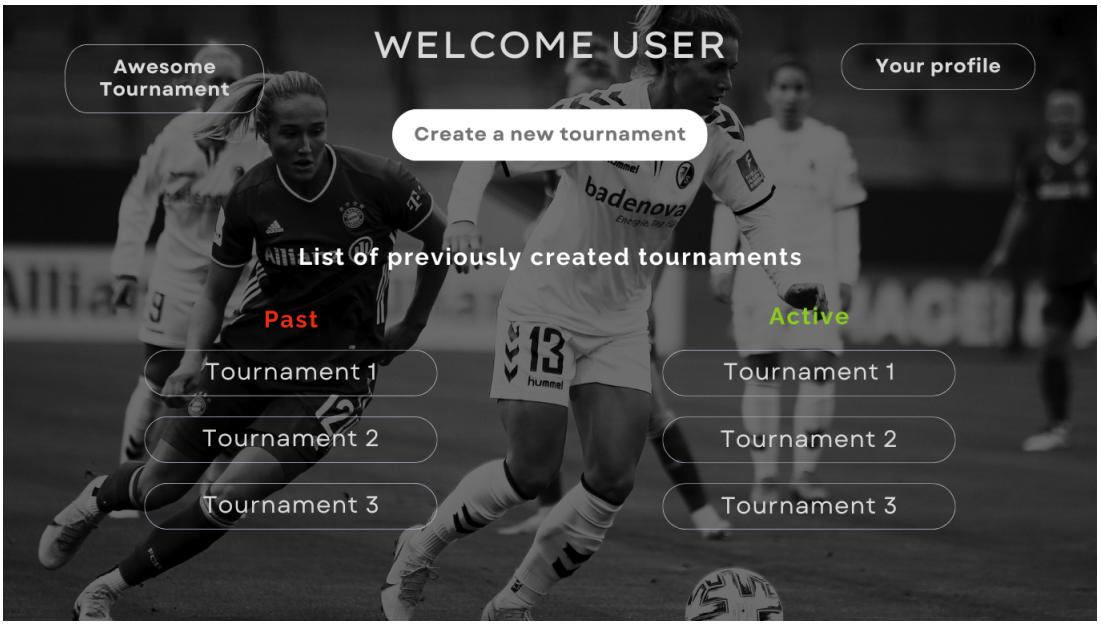
Below are some examples of pages implementing the functionalities described before.

- **Homepage:** it displays both past and active tournaments even if the user is not logged in. By clicking on a past tournament all the details are shown (results, both table ecc.); by clicking on an active one it is displayed the same thing but in realt time (to be updated). In the top right corner there is a button that allows the login for the user.

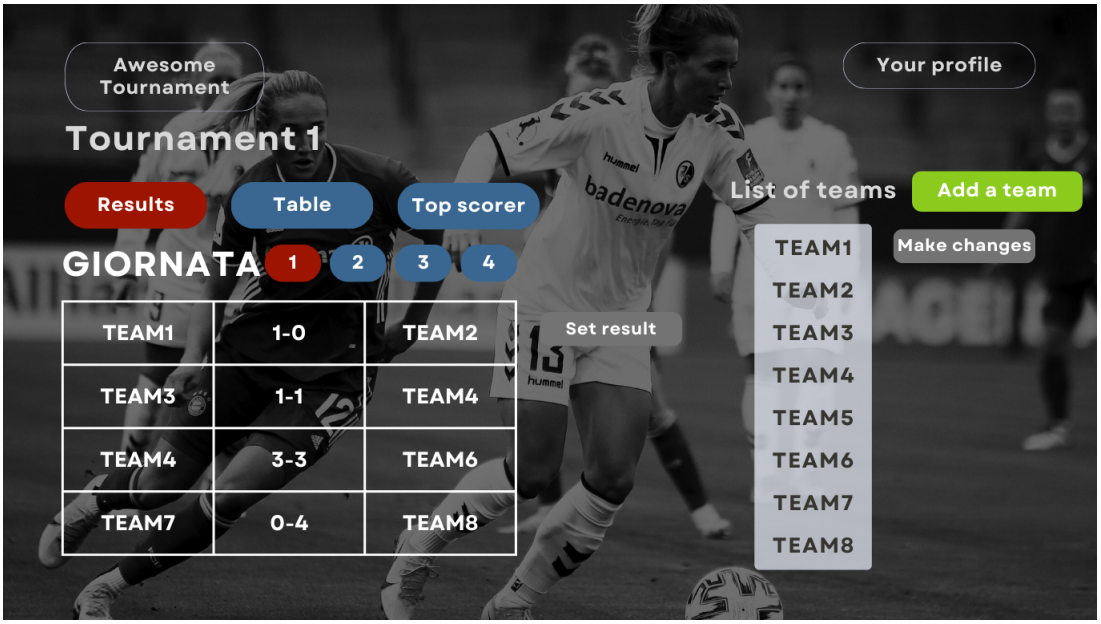
- **Login in page:** it is requested to provide a valid email and a secure password. By clicking on "Sign up" the user can register on the site.



- **Homepage after login:** after the login all the past and active tournaments created by the user are displayed. From here the user can start the process to create a new tournament.



- **Page of a tournament:** the page displays the infos about a tournament. The user can decide what to check from the three button (Results, Table and Top scorer). It is also possible to choose the matchday to check. For the admin of the tournament and the creator of that team it is also possible to manage the team.



- **Page of a specific match:** the page displaying the infos about a specific match of a tournament. Only the admin of the tournament is allowed to fill in the events for a match by clicking on "update result and scorers".

- **Process of creating a tournament:** the image below display the process of creating a tournament. Only the last required info are displayed, like the starting date, the deadline and the logo.

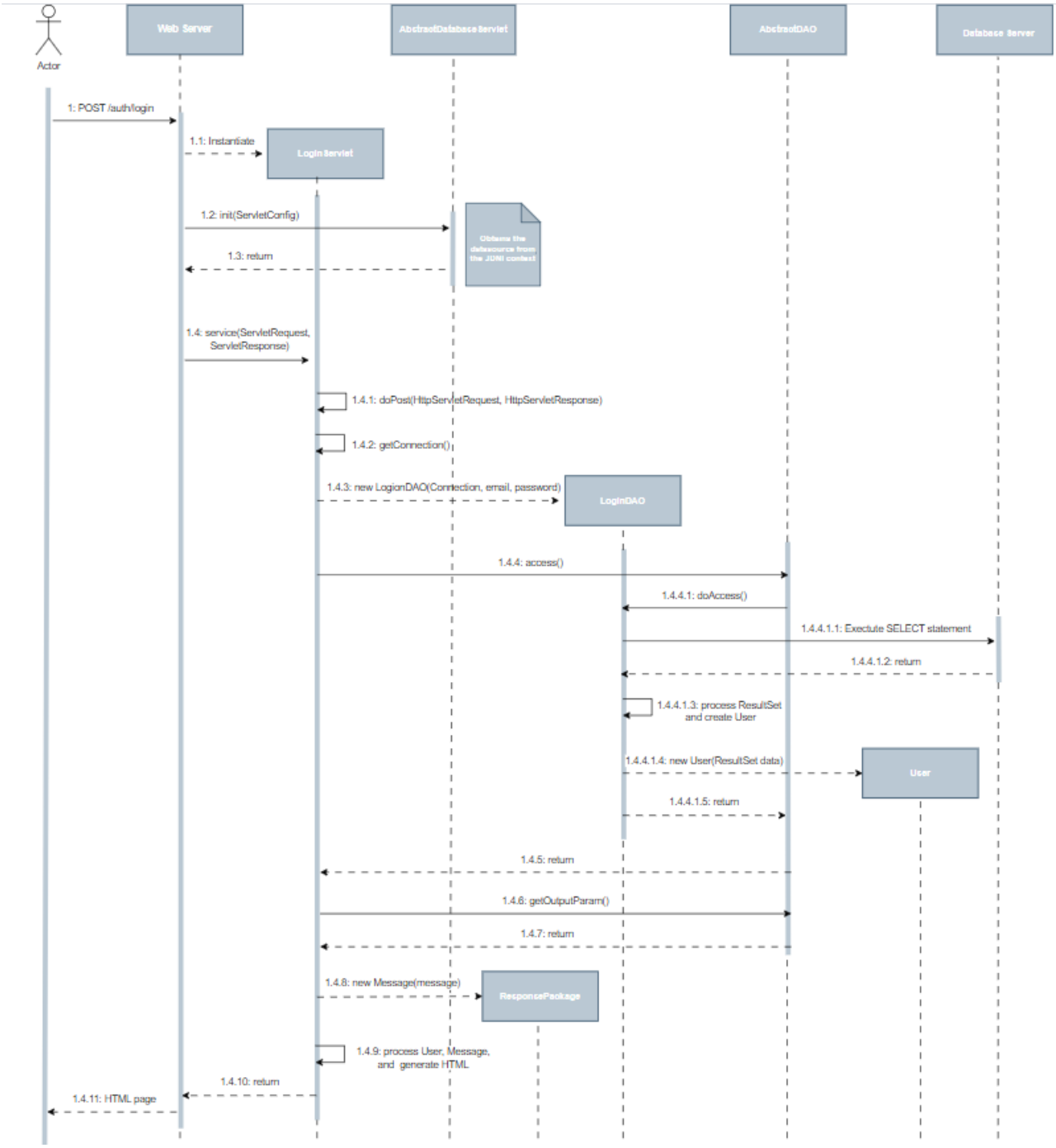
5 Business Logic Layer

5.1 Class Diagram

IMMAGINE

DA MODIFICARE: The class diagram contains (some of) the classes used to handle six types of resources: users, tournaments, matches, teams, players and events. It is possible to observe that there is only one resource handled by only one Servlet, that is Member. All servlets implement the doGet and doPost methods, as sublcasses of the HttpServlet class (here omitted). Regarding the Organizer, there are two Servlets used to handle the resource. In particular, OrganizerServlet handles the registration of new organizers and allows to see a list of all the organizers, together with their information. The AuthenticationServlet handles the logins in the web application, in particular its doGet method handles the request for the logout, meanwhile the doPost method handles the login (or in other terms, the authentication) of the users in the application. Once the servlet has processed the request, it forwards it to the proper DAO class (Organizer or User). Concerning the Member, the only servlet that handle this resource is UserServlet. Such Servlet implements the doGet and doPost methods, that handle different operations. The doGet method, through a URI request, allows to retrieve a list of all the users with their information or the information of a single user given the phone number. The doPost method, on the other hand, manages the registration of a new user and the upgrade of a pre-existing user's role. Regarding the Conferences and the conference bookings, these resources are handled through a servlet and a REST class. The Servlet, called ConferenceReservationServlet, has a doGet method that allows to retrieve a list of reservations given a parameter (a date or a user phone number), and a doPost method to create a new reservation for a conference. The REST class, called ConferenceRest, manage the REST call about conference. This class contains four methods, used to delete and insert conferences and to retrieve a list of the attendances for all conferences or a specific one. ConferenceRest is a subclass of RestResource and is invoked from RestDispatcherServlet. To handle the interaction with the database, we have five Data Access Objects (DAO), one for each resource plus one to handle conference bookings: OrganizerDAO, MemberDAO, UserDAO, ConferenceDAO and ConferenceBookDAO. Each DAO implements the methods to retrieve, insert and update the different resources. ConferenceDAO is the only one that allows the delete operation. Each resource is mapped into a specific data class, that has been omitted due to space reasons. Instances of such classes are instantiated by and passed between servlets and DAOs.

5.2 Sequence Diagram



DA MODIFICARE: Here is reported the sequence diagram for the operation about the insert of a simple user. The user executes a POST request to the web server. The web server instantiates the UserServlet, get the information about the datasource from the init method of the AbstractDatabaseServlet. After that it calls the doPost method of the UserServlet, passing the HttpServletRequest and the HTTP Servlet response. The UserServlet analyzes the request and recognizes that it is an insert operation, thus it creates an object User with the request parameters. At last, with the using of the insertNewUser method of the UserDAO class, the servlet

inserts the User in the database. After that, the UserServlet replies to the web server with the methods setStatus and getWriter that notify if everything goes well or not.

5.3 REST API Summary

Here is the list of REST API implemented.

URI	Method	Description
/matches	What HTTP method uses?	Describe briefly the result of calling the URI
/players	What HTTP method uses?	Describe briefly the result of calling the URI
/tournaments	What HTTP method uses?	Describe briefly the result of calling the URI
/teams	What HTTP method uses?	Describe briefly the result of calling the URI
/events	GET	Retrieves an event by its ID
/events	PUT	Updates an existing event
/events	DELETE	Deletes an event by its ID
/matches/{matchId}/events	GET	Retrieves events associated with a specific match
/matches/{matchId}/events	POST	Creates a new event for a specific match

Table 2: REST API Summary

5.4 REST Error Codes

Here is the list of errors defined in the application.

Error Code	HTTP Status Code	Description
200	OK	The request has succeeded
201	CREATED	A new resource has been created
204	NO_CONTENT	The server does not need to return any content in the response body
302	FOUND	The requested resource has been temporarily moved to a different URL
400	BAD_REQUEST	The server cannot process the request due to invalid request structure
401	UNAUTHORIZED	The request lacks proper authentication credentials or the credentials provided are invalid

403	FORBIDDEN	The user is not authorized to perform that particular action
404	NOT_FOUND	The server cannot find the requested resource
405	METHOD_NOT_ALLOWED	The server does not support the HTTP method used in the request for the specified resource
500	INTERNAL_SERVER_ERROR	The server encountered an unexpected condition that prevented it from fulfilling the request
501	NOT_IMPLEMENTED	The server does not support the functionality required to fulfill the request
503	SERVICE_UNAVAILABLE	The server is currently unable to handle the request

Table 3: REST Error Codes

5.5 REST API Details

We report here (?) **DA DECIDERE** different type of resources that our web applications handle
Insert new player

- URL: `the URL to retrieve it`
- Method: `Method to retrieve it`
- URL Parameters:
- Data Parameters:
- Success Response:
- Error Response:

Delete an event

- URL: `the URL to retrieve it`
- Method: `Method to retrieve it`
- URL Parameters:
- Data Parameters:
- Success Response:
- Error Response:

Get a list of all the matches within a tournament

- URL: `the URL to retrieve it`
- Method: `Method to retrieve it`
- URL Parameters:
- Data Parameters:
- Success Response:
- Error Response:

...

- URL: `the URL to retrieve it`
- Method: `Method to retrieve it`
- URL Parameters:
- Data Parameters:
- Success Response:
- Error Response:

6 Group Members Contribution

Alberto Basaglia Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Andrea Bruttomesso Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Alessandro Corrò Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Milica Popovic Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Davide Seghetto Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Andrea Stocco Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.