# Table of Contents

1. Introduction
   1.1 Goals and objectives
      1.1.1 For the society
      1.1.2 For a single Person
      1.1.3 For the sponsors
   1.2 Statement of scope
   1.3 Software context
   1.4 Major constraints

2. Data design
   2.1 Internal software data structure
   2.2 Global data structure
   2.3 Temporary data structure
   2.4 Database description
      2.4.1 Table users
      2.4.2 Table events
      2.4.3 Table players
      2.4.4 Table teams
      2.4.5 Tables for each teams (member of the team)
      2.4.6 Simple representation of database and tables
      2.4.7 tables attributes

3. Architectural design
   3.1 Program Structure
      3.1.1 Architecture diagram
      3.1.2 Alternatives
   3.2 Description for Component n
      3.2.1 Generating new event
         3.2.2.1 delete a team
         3.2.2.2 remove a member
         3.2.2.3 upgrade member
         3.2.2.4 downgrade member
         3.2.2.5 edit team profile
         3.2.2.6 create an event
         3.2.2.7 cancel an event
         3.2.2.8 list events: (indirect function)
3.2.2 Generating a new team

3.2.2.1 delete a team

3.2.2.2 remove a member

3.2.2.3 upgrade member

3.2.2.4 downgrade member

3.2.2.5 edit team profile

3.2.2.6 create an event

3.2.2.7 cancel an event

3.2.2.8 list events: (indirect function)

3.3.1 Processing narrative (PSPEC) for component n

3.3.1.1 Generating new event

3.3.1.2 cancel an event

3.3.1.3 join an event

3.3.1.4 leave an event

3.3.1.5 Generating a new team

3.3.1.7 Delete a team

3.3.1.8 Remove a member

3.3.1.9 upgrade member

3.3.1.10 downgrade member

3.3.1.11 edit team profile

3.3.1.12 Generating new event (Team owner or captains)

3.3.1.13 cancel an event (Team owner or captains)

3.3.1.14 list events (Team owner or captains)

3.3.2 Component n interface description

3.3.3 Component n processing detail

3.3.3.1 Generating new event

a) cancel an event

b) join an event

c) leave an event

d) list events

3.3.3.2 Generating a new team

a) delete a team

b) remove a member

c) upgrade member

d) downgrade member

e) edit team profile

4. Schedule

4.1 Scheduling diagram
5. User interface design
   5.1 Description of the user interface
      5.1.1 Screen images
         a) user
         b) team
      5.1.2 Objects and actions
   5.2 Interface design rules
      5.2.1 simplicity
      5.2.2 interactive
   5.3 Components available
   5.4 UIDS description
6. Restrictions, limitations, and constraints
   6.1. Geographic
   6.2. Veracity
   6.3. Resources
1.0 Introduction

1.1 Goals and objectives

The goal for the uSports system is to create an environment in which anyone is able to organize sporting events among friends and other players in their area. The only requirement a player needs it is just apply for a match or create a new one if there is not a current one to join. In short we want to cover these following goals with our software:

1.1.1 For the society:
- Make easier for everyone to play any sport.
- Increase the sport activity of the population.

1.1.2 For a single Person:
- Play any sport, any date at any time, alone or with friends.
- Choose against who he wants to play.
- Meet new people who share his sport interests.
- Create and modify his own sport team easily adding/removing people who live around.

1.1.3 For the sponsors:
- Redirect their advertisements to people who are really interested in their products.

1.2 Statement of scope

In order to make easier the understanding of the software we are going to use a basic schema.
1.3 Software context

Thinking about the business context, there is no competence in this area. Because, as we know, there is not a similar website/application/service which offers the same than our application wants to. Nowadays everyone is using standard social networks to organize sport events with friends, but they are not specially designed for that purpose. The ability to create teams or join an event with people who the player doesn’t know it is simply impossible. Thus, we believe in our software capacities in order to be used by a lot of people, because there is no similar way to do what we offer using a different method.

1.4 Major constraints

Our software will be designed mainly for healthy and sporty people, that’s why we should guide our interface on this way, including pictures of people practicing sports or healthy activities, also focusing the profile of each player in sport preferences and information about sport activities they practice.

Also the software is designed for a wide average of ages. From 14 years old to 60-70, so They are not supposed to have high knowledge in computers that’s why the simpler is our interface the better results we will obtain.

It can be good to test our software with this kind of people, in order to know their opinion and make changes according to their reports.

2.0 Data design

2.1 Internal software data structure

The data of this website is stored in a MySQL database. The database will
have five separate tables. The first table will hold the static information about the website. The second table will keep the user’s information. The third table will handle the sport events generating, cancellation and editing. The data is mostly modified by the user of the website, but the user will have access for specific functionality based on his location. Forth table is the players, which tracks the join activates. The fifth table is the Team table.

**2.2 Global data structure**

The user will have access to all tables, but these accesses depend on where the user is located within the website. If the user is located at the profile page, a specific function will be enabled. Thus, the user will be altering the stored data from different views. The first view is as an individual via his profile, and second view as an owner of a team.

**2.3 Temporary data structure**

The website will have a folder named configuration. The folder configuration, includes specific codes and scripts that are used only when installing the website on the public server. These files create the database, tables of database, and inserting the static information about the website and also creating the admin control panel. After the installation, the folder is removed. A copy must be saved somewhere else for future use.

**2.4 Database description**

The database will have five tables at a starting point. The tables are users, events, players and teams. However, whenever the user is inserting a new team into the table event another unique table for that team will be created. Thus, the number of teams created will have the same number of member of team tables. In another word, each team will have a unique member table.

**2.4.1 Table users**

This table will store all the user information.

**2.4.2 Table events**

This table will store all the events that are created by users.

**2.4.3 Table players.**

This table will store all the users who joined an event. It will store the user ID, event ID; the date the user joined the event.

**2.4.4 Table teams.**

This table will store all the teams.

**2.4.5 Tables for each teams (member of the team).**

Each team will have a member table that is created with the team. This table will have the name "member_" plus the team ID as illustrated below.
2.4.6 Simple representation of database and tables

2.4.7 tables attributes.

### [events table]

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Title</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Sport</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Date</td>
<td>date</td>
</tr>
<tr>
<td>Starttime</td>
<td>date</td>
</tr>
<tr>
<td>Endtime</td>
<td>date</td>
</tr>
<tr>
<td>Agemin</td>
<td>int(11)</td>
</tr>
<tr>
<td>Agemax</td>
<td>int(11)</td>
</tr>
<tr>
<td>Gender</td>
<td>varchar(46)</td>
</tr>
<tr>
<td>Address</td>
<td>varchar(145)</td>
</tr>
<tr>
<td>Zipcode</td>
<td>varchar(15)</td>
</tr>
<tr>
<td>Gametype</td>
<td>varchar(46)</td>
</tr>
<tr>
<td>Playersmin</td>
<td>varchar(46)</td>
</tr>
<tr>
<td>Playersmax</td>
<td>varchar(46)</td>
</tr>
<tr>
<td>Locationrange</td>
<td>varchar(46)</td>
</tr>
<tr>
<td>Registratetyp</td>
<td>varchar(46)</td>
</tr>
<tr>
<td>Ownerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Dateofcreation</td>
<td>date</td>
</tr>
</tbody>
</table>

### [teams table]

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Teamname</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Sportlist</td>
<td>varchar(225)</td>
</tr>
<tr>
<td>Ownerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Creationdate</td>
<td>date</td>
</tr>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Teamname</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Sportlist</td>
<td>varchar(225)</td>
</tr>
<tr>
<td>Ownerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Creationdate</td>
<td>date</td>
</tr>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Teamname</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Sportlist</td>
<td>varchar(225)</td>
</tr>
<tr>
<td>Ownerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Creationdate</td>
<td>date</td>
</tr>
</tbody>
</table>

### [users table]

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Teamname</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Sportlist</td>
<td>varchar(225)</td>
</tr>
<tr>
<td>Ownerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Creationdate</td>
<td>date</td>
</tr>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Teamname</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Sportlist</td>
<td>varchar(225)</td>
</tr>
<tr>
<td>Ownerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Creationdate</td>
<td>date</td>
</tr>
</tbody>
</table>

### [players table]

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int(11)</td>
</tr>
<tr>
<td>UID</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Playerid</td>
<td>varchar(125)</td>
</tr>
<tr>
<td>Playertype</td>
<td>varchar(45)</td>
</tr>
<tr>
<td>Date</td>
<td>date</td>
</tr>
</tbody>
</table>
3.0 Architectural design

3.1 Program Structure

Since, this is a website, the structure is based on the user logging to his account from the home page. The user then can interact with the system in different ways.
3.1.2 Alternatives

We have used the WAE modeling, since it is particularly suited to modeling the client-server interactions that are typical of webApps. The WAE represents different icons that when they are combined it becomes possible to represent a
complex interaction between data objects, functional object and presentation objects as well to indicate where the interaction between these objects occurs. Examples of icons used by WAE modeling are client page, HTML form, Link target, Server page, JavaScript Object, dynamic page, web page and finally physical root. The WAE model categorizes these icons into three different views. The first view is analysis view, second view is the logical view and the third view is component view. The WAE modeling allows us to mix UML within its icons and representation. Thus, we could easily describe the interaction between the database and the user’s functions.

3.2 Description for Components

3.2.1 Generating new event:

This function allows the user to insert a new event into database. The user can do different things with the generated event. The user can cancel the event, join other event, leave an event or list all events.

3.2.1.1 cancel an event:

This function allow the user to cancel an event he created before.

3.2.1.2 join an event:

This function let the user join an open event.

3.2.1.3 leave an event:

This function let the user leave an event.

3.2.1.3 list events:

This function let the user list all events that matches his information.

3.2.2 Generating a new team:

This function allows the user to create a new team and set himself as the owner. The user can do different things with the generated team. The user can join a team, delete a team (if he is the owner).

3.2.2.1 delete a team:

This function allows the user to remove a team that he owns.

3.2.2.2 remove a member:

This function allows the user to remove a member from the team he owns.

3.2.2.3 upgrade member:

This function allows the user to upgrade a member of his team to be a captain.

3.2.2.4 downgrade member.

This function let the owner of a team downgrade a member from captain to regular member.

3.2.2.5 edit team profile.

This function let the user change/edit the profile of the team.
3.2.2.6 create an event:
This function let the owner create an event under the team name. This function works similar as “create an event” for an individual user. The main difference is rather than creating an event under the user ID it creates the event under the ID of the team.

3.2.2.7 cancel an event:
This function let the owner cancel the event he created for the team. This function works similar as the “cancel function” for user. The main difference is rather than checking the logged user ID, it checks if the user ID is the owner of the visited team’s page.

3.2.2.8 list events: (indirect function)
This function let the owner to list all events that matches the team information.

3.3.1 Processing narrative (PSPEC) for component n
3.3.1.1 Generating new event:
The user must be logged in. The user will fill in some details about the event such event name, sport name, age range, address, targeted location range, gender and other restrictions. These inputs are inserted to a table in database named “events”. However, it is required that for each event there is an owner, the owner is inserted using a session that is called whenever the user login to the system. The session will store the user ID and it will be called whenever the user submits the request to generate the new event. The database will receive the input from the HTML form; the user ID from SESSION and also the Php script will automatically generate a unique code for the event. In addition the Php script will store date of today into the database. Also, the Php script will convert the date and time of the event into a useful format that can be used later for filtering purpose.

3.3.1.2 cancel an event:
The user must be logged in. The user will only be able to cancel the event where he is the creator of that event. The user will be able to list the events that he created. The system will check the user ID and search from database where the owner’s ID of all events matches the logged user ID. However, the system will only list the events that are in the future. The system will get the date and time of each event the user is the owner and check if it in future, if yes the system will provide the user the event. The list will include some information about the event and a button for canceling the
event. The button will have the value of the unique ID for that particular event. The system then removes the event from the database. Furthermore, the system will notify each user who joined the event about the cancellation. That is, the system will have to check the players’ table and find where the event’s ID occurs and takes the entire user IDs and then search for their email address from the user tables.

3.3.1.3 join an event:

The user must be logged in. The user will only be able to list the events that match his information. Since, each event have some restrictions such as; age range, gender and other restrictions. The user will view all the events and each event that occurs on the list will have some information and a button for joining the event. When the user submits the request, the system will first check if there is a room for the user to join or not. If so, it will ask the user to select which team to join. After the user select which team he want to join, the system will take the user’s ID date/time and the event’s ID and then insert these information into table named “players”.

3.3.1.3 leave an event:

The user must be logged in. The user lists all the upcoming events he is signed in. This information is brought from the table players. The systems will takes the user’s ID and search from table players where the future events are taking place. Since, each entry have a unique ID the user will be able to click a button to remove his ID from table players. However, the system will need both the user’s ID and the event’s ID to be able to remove the user from the table “players”.

3.3.1.4 list events:

The user must be logged in. The user will have two different ways to list all events. The default one is; the system will only display events that match the user information (each event have restrictions). The other way is the user can filter the list for specific restrictions.

3.3.1.5 Generating a new team:

The user must be logged in. The user will fill in some details about the team such name of the team and sport. This information is inserted into database to a table named “teams”. Since, each
new team entry will have a unique ID the unique ID that is auto
generated when the form is submitted, this unique ID is used to
generate a pair table called "members_" + the team unique ID.

3.3.1.7 Delete a team:

The user must be logged in. The user will only be able to
delete a team where he is the creator of that team. The user will be
able to list the all teams that he created. The system will check the
user ID and search from database where the owner’s ID of all
teams matches the logged user ID. The list will include some
information about the team and a button for deleting the team,
editing profile and editing members. The button will have the value
of the unique ID for that particular team. The system then removes
the team from the database. Furthermore, the system will notify
each user who is a member of that team. That is, the system will
have to check the paired table “members_[team unique Id]” and get
all members’ IDs and from the table users the system then gets
their emails and sends messages to each one. After sending the
message the system will drop the paired table of that deleted team.

3.3.1.8 Remove a member:

The user must be logged in. The user must be in the team
page to display functions that only presented for the owner and the
captains of the team. The system will get the user’s ID and check if
he is the owner from the table “teams”, if so functions such delete a
member will be shown. The owner will click on the delete sign
beside member name. The button “delete” will have a value of the
user ID and the team ID. Thus, the system will search team
members’ table and delete the member from that table.

3.3.1.9 upgrade member:

The user must be logged in. The user must be in the team
page to display functions that only presented for the owner and the
captains of the team. The system will get the user’s ID and check if
he is the owner from the table “teams”, if so functions such upgrade
a member will be shown. The owner will click on the upgrade sign
beside the member name. The button “upgrade” will have a value
of the user ID and the team ID. Thus, the system will search team
members’ table and change the value of member type from 0 → 1.
The system will notify the user by taking the user ID and search
from the table users and then get their emails and sends a
message to each one.

3.3.1.10 *downgrade member:*

The user must be logged in. The user must be in the team page to display functions that only presented for the owner and the captains of the team. The system will get the user’s ID and check if he is the owner from the table “teams”, if so functions such downgrade a member will be shown. The owner will click on the downgrade sign beside the member name. The button “downgrade” will have a value of the user ID and the team ID. Thus, the system will search team members’ table and change the value of member type from 1 → 0. The system will notify the user by taking the user ID and search from the table users and then get their emails and sends a message to each one.

3.3.1.11 *edit team profile:*

The user must be logged in. The user must be in the team page to display functions that only presented for the owner and the captains of the team. The system will get the user’s ID and check if he is the owner from the table “teams”, if so functions such edit team profile will be shown. The owner will click on the edit sign from the team’s owner menu. A form that request the new desired team name if wanted, the new sport and a file to upload in case the owner want to change the background image of the team. The form will pass these information into a php script where the php script update the new values for the team and upload the image into a folder in the directory “teams/Team_ID/img”.

3.3.1.12 *Generating new event (Team owner or captains):*

The user must be logged in. The user must be in the team page to display functions that only presented for the owner and the captains of the team. The system will get the user’s ID and check if he is the owner from the table “teams”, if so functions such create a new event will be shown. The owner will click on the new event sign. The button “new event” will have a value of for the team ID. Thus, the system will store the team ID and pass it to the next page. The next page is a HTML form. The owner will fill the form for the new event. Based on the choices the owner selects the form may display more functions or may disable some functions. If the owner wants to select which member of his team to be called for this new event, a list of the team members will be shown. The owner then selects which member he wants to join the event. When the form is
submitted the system will check what choice the owner have selected. If he selected to choose the players by himself then the system will take the list of invited member and automatically insert their IDs into the table “players” including the event ID, the team ID.

3.3.1.13 cancel an event (Team owner or captains):

The user must be logged in. The user must be in the team page to display functions that only presented for the owner and the captains of the team. The system will get the user's ID and check if he is the owner from the table “teams”, if so functions such list team events will be displayed. The owner will click on the list team events sign. The button “list team event” will have a value of the team's ID. Thus, the system will search for all upcoming events that are under the team ID. The system will list all the upcoming events created by the team. Each row will have some information about the event. The owner then can click on a button “cancel event” and the button will pass the value of event's ID and the team's ID then the Php script will delete the event from the table “events”. The system will also check the table “players” and remove all the members who joined the event and notify them via email.

3.3.1.14 list events (Team owner or captains):

The user must be logged in. The user must be in the team page to display functions that only presented for the owner and the captains of the team. The system will get the user's ID and check if he is the owner from the table “teams”; if so functions such list events will be displayed. The owner of the team will have two different way to list all events. The default one is; the system will only represent events that match the team information (each event have restrictions). The other way is the owner can filter the list for specific restrictions.

3.3.3 Component n processing detail

3.3.3.1 Generating new event:

a) cancel an event:

1. from table events search for upcoming events
2. filter where user ID appears as an owner of these events
3. if user clicks on cancel event button go to 4
4. store the user ID and the event ID
5. from table players store all user ID where the owner Id=stored user ID and where the event Id= the stored ID.
6. from table users get the email address for each ID and store it.
7. send a message to each email.
8. delete from table players where owner ID= the stored ID and the event ID = the stored event.
9. delete from table events where owner ID = stored user ID and where the event ID = the stored ID.

b) join an event:
1. from table events search for upcoming events
2. filter where restrictions matches the user information
3. if user clicks on join event button go to 4
4. ask the user to select which team to join.
5. store the user ID and the event ID
6. insert the user ID and ID into table players.

c) leave an event:
1. from table players search for upcoming events.
2. filter where the user ID matches the current logged user.
3. if the user click leave button go to 4.
4. store the user ID and the event ID.
5. from table players delete where user ID = the stored ID and the event ID = the stored event ID.

d) list events:
1. from table events search for upcoming events.
2. filter where restrictions matches the user information.

3.3.3.2 Generating a new team:

a) delete a team:
1. from table teams search for teams that the owner ID = the current logged user.
2. filter the teams.
3. if the user click delete team go to 4.
4. store the team ID and owner ID.
5. from table teams delete where owner ID = stored ID and team ID= the stored team ID.
6. append the string “member_” with the team ID.
7. drop from database a table “member_teamID”.

b) remove a member:
precondition: The user is the owner of a team and his current location is the team page.
1. store the team ID.
2. append the string “member_” with the team ID
3. find the table member_teamID and list all members
4. if the user clicks on remove a member.
5. store the member ID
6. delete from table member_teamID where the member ID=the
stored member ID.

c) upgrade member:
   precondition: The user is the owner of a team and his current location is the team page.
   1. find the table members of the team
   2. list all members
   3. if the user click on upgrade member
   4. change the value of member type from 0 → 1

d) downgrade member.
   precondition: The user is the owner of a team and his current location is the team page.
   1. find the table members of the team.
   2. list all members
   3. if the user click on the downgrade
   4. change the value of the member type from 1 → 0

e) edit team profile.
   precondition: The user is the owner of a team and his current location is the team page.
   1. show edit button
   2. take the user to edit page
   3. show fields
   4. the user clicks on update
   5. store the team ID.
   6. update from table teams where team ID= the stored team ID.

4.0 Schedule

This program development will undergo four major cycles. The first cycle is the Documentation Phase where the Requirements and Software Design documents are built. These documents will provide the group with the layout for each module and diagrams on how everything works together. The second cycle is to divide the components among each member of the team and each member will do the actual coding. The Third cycle is to bring these components together and integrate them. The fourth cycle is to testing the website.
4.1 Scheduling diagram

5.0 User interface design

5.1 Description of the user interface
The user interface will be designed like most modern websites, with a fixed navbar at the top and a scrollable content area below. There will be a user drop-down menu at the top with important buttons such as logging out, viewing their profile, creating events, viewing activity, etc. If a user is not signed in there will also be a sign up button in case a new user is visiting the website. Then, there are navigational buttons for the main web pages such as home and search for now.

5.2 Interface design rules
The interface is being designed minimalistically, while maximizing the functionality of the application. The time constraint of the class has demanded we try to maximize functionality while minimizing work on the user interface.
Another important aspect of the application design is that it be optimized across multiple devices. Today, web apps can be viewed on any screen from high-definition tvs to mobile phones, so it’s important to consider this in design. To combat this issue we will be using Twitter’s Bootstrap CSS Framework, which has the ability to automatically resize some views depending on the screen size.

5.3 Components available
Twitter Bootstrap comes with some built-in design components:
- Navbar
- Pagination
- Breadcrumbs
- Buttons
- Thumbnails
- Labels
- And more...

5.4 UIDS description
Because we are only designing a website, the code can be edited in nothing more than a basic text editor. To see the changes, however, we’re going to be running a local web server such as MAMP or Apache. These tools allow the developer to host websites from their local machine and viewed by visiting localhost:port#

6.0 Restrictions, limitations, and constraints
Our main restrictions are:

6.1 Geographic:
- Our software is based in the real world, we are just an interface between fields and players, for instance a squash event can not be organized in a small town which can not provide a squash field.
- For the same reason a match cannot be organized between 2 guys who lives far away.

6.2 Veracity:
- If people apply for events and after they do not go all our power is broken, that’s why we will implement a truthfulness rate system based in the previous events (similar to ebay) in order to avoid this situation.

6.3 Resources:
- If a sport center just have a soccer field, only one soccer game will be played there at the same time.
- The same will happen for working days and open hours