Spinshot

Technical Design Document

Gonçalo de Jesus

ARBORESIS
STUDIO

Table of Contents

Executive Specifications 2
  Game Information 2
Design Specifications 4
  Control System 4
  Game Modes 7
  Customization System 9
  Filing/File Exposition System 9
  Online Functionalities 9
Game Database Specifications 10
    Structs, Enums and Custom Classes 11
    General 20
    Game Pieces 23
Technical Specifications 40
  Control System 40
    Control System Flowchart 40
  Game Modes 42
  Customization System 42
  File System Handling 42
  Networking Structure 43
Design References 44
Sport Overview 44
  Sport and Rule Summary 44

Executive Specifications

Game Information

Name
Spinshot

Proposal:
IHF Spinshot (if the IHF license is obtained)

Genre
Sports, Simulation

Platforms
Windows, Linux, PlayStation, Xbox, Switch

Product Overview
Spinshot is the handball simulator that brings you to the heart of the game, lets you submerse yourself in the atmosphere of playing handball at the highest level. Experience the intensity and physicality of being out on the court, and feel your heart skip a beat when you're face-to-face with your opponent. Customize your teams, your players and even your arenas, and sense it all through our immersive and intuitive control system, so you never miss a queue.

Unique Selling Points

- **Jump into fast-paced handball action**
  Every decision is key, and juggling physical prowess with tactical know-how will be vital on your journey to glory.
- **Manage, compete, dominate**
  Various game modes bring you multiple ways to play; from career and manager modes to friendly competitions with friends, feel handball from different perspectives, on your own or with company.
- **Personalize every aspect of your game**
  With our in-depth tools, create and edit players, teams, and whatever else you need to make each match feel yours.
- **Find your style, level up your skill**

> Our immersive control system allows you to keep your favorite shots on the tip of your fingers. Blur the line between you and your players: read the play, call the shots, and bring home the W.

Licensing

The main idea would be to obtain the license sponsorship from IHF (International Handball Federation).
If this is not possible, the second-best license to obtain would be the EHF (European Handball Federation), as Europe is the main continent for the sport.

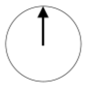
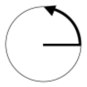
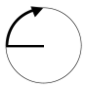
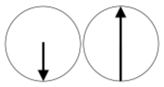
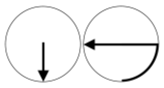
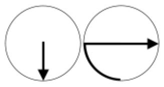
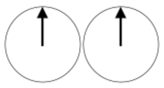
Design Specifications

Control System

Attacking

Base controls

Left analog stick	- Move player, define passing direction, and aim shot
Right analog stick	- Shooting (through gestures)

Shot type	Analog stick movement
Quick shot	
Wrist shot	
Overhead shot	
Normal "whip" shot	
Spin shot (outer)	
Spin shot (inner)	
Lob shot	 *(in quick succession)*

R2	- Steps without dribbling (on release, jump)
R1	- Pass
L2	- ‘Bounce’ modifier
L1	- ‘Fake’ modifier (used to fake shots and passes, …)

↑	- Trigger created play 1
→	- Trigger created play 2
↓	- Trigger created play 3
←	- Trigger created play 4

✕	- Cycle backcourt lines (CB, LB, RB)
○	- Cycle frontline lines (P, LW, RW)
△	- Toggle Winger as second Pivot
□	- Toggle the main Pivot's side of the field

Explanation

R2 allows the user to take steps without dribbling, and have the player jump when it's released.
By default, players dribble the ball when they start moving. With **R2** held down, the player holding the ball starts taking steps without dribbling, preparing a jump. When the user lets go of **R2**, the player holding the ball should jump within the context of the position and direction that player was heading.
For example, if a winger receives the ball and the user is pressing **R2** as they run up to the 6-meter line, the winger should perform a classic winger run-up and, when the user releases the button, jump facing the opposing team's goal, with the intention of shooting.

L2 acts as a modifier that allows the user to perform bouncing passes or shots. With **L2** held down, if the user performs a pass, the player will throw the ball at the ground with the intention of passing to the indicated teammate.

L1 acts as a modifier that allows the user to fake a pass or shot. With **L1** held down, any passing or shooting commands should be taken as fake and the player holding the ball should reflect that action, with the intention to trick the opponent.

✕ and ○ control substitutions.
For substitutions, there is the option to do them through the pause menu during the game. However, for convenience, the user can define four frontline and backcourt 'lines'.
'Lines' are combinations of players, which may contain repeated players.
Example:

Frontline Line 1	Winger 1, Pivot 1, Winger 2
Frontline Line 2	Winger 1, Pivot 2, Winger 2

Frontline Line 3	Winger 3, Pivot 1, Winger 4
Frontline Line 4	Winger 3, Pivot 2, Winger 4

These lines would allow the user to rotate between all the wingers and all the pivots, giving the team depth while allowing quick access to the different lines during the game without needing to interrupt play.
By default, the lines aren't named, but there should be an option for the user to define the name of each line, so that they can better understand the changes they're making without too much effort during a match.

△ acts as a toggle and controls whether one of the wingers is acting as a second pivot (or main pivot if the team's shorthanded and the player missing was the main pivot).
If the button is pressed when the winger is on the wing, they become a pivot; if the button is pressed when the winger is acting as a pivot, they go back to the wing they belong to.
The winger that assumes these commands should be defined in the team's management menu (in a pre-game menu or, alternatively, in the pause menu).

□ acts as a toggle and controls which side of the field the main pivot should play in.
If the button is pressed when the main pivot is on the left side, they should go to the right; if the button is pressed when the main pivot is on the right side, they should go to the left.
If there is a second pivot at play, the second pivot should assume the side opposite of the main pivot's. As such, when the main pivot, for example, changes from the left side to the right side, the second pivot should change from the right side to the left.

Defending

Base controls

Left analog stick	- Move player and define which player to switch to *(see **R1**)*
Right analog stick	- Define the defender's motion when blocking a shot

R2	- Jump (on release)
R1	- Switch player
L2	- Lower stance
L1	- Intercept

↑	- Switch to a 5-1 defensive formation
→	- Switch to a 3-2-1 defensive formation
↓	- Switch to a 6-0 defensive formation
←	- Switch to a 4-2 defensive formation
✕	- Cycle backcourt lines (CB, LB, RB)
○	- Cycle frontline lines (P, LW, RW)
△+□	- Pull goalie for the next attacking play (if pressed again, the pull is canceled)

Explanation

L2 acts as a modifier and should lower the stance of the player being controlled in defense.
This means that, while **L2** is held down, the player should go from a regular defending stance to a lower, almost squat-like stance so as to prevent low shots or bounce passes. When **L2** is released, the player should assume the regular defending stance.

L1 makes the player being controlled lunge at the ball with the intention of intercepting a pass. It is most effective when a pass flies near the player, but just beyond reach, as the player will go all-out to try and catch the ball when this action is used, leaving the defense open if the catch isn't successful.

Game Modes

Exhibition Match

Exhibition Matches are 'loose' matches, in which users may quickly start a match without context and with very little commitment. These are designed to be a good way to jump into the actual gameplay, needing almost no setup besides selecting the participating teams.
By default, these should be friendly matches between the two teams.
However, there should also be an option to select variations, such as cup finals or deciding league matches, to allow players to play a quick match with a bit more immersion if they want to.

This mode should offer an option for 'best-of-3' and 'best-of-5' sequences, to improve the experience for users who, for example, opt for a small head-to-head with a friend.

Career Mode

Career mode is the game mode where users can jump into the shoes of a (real or fictional) player or manager, surrounded by everyday activities, situations and decisions that come with that.

In be-a-player mode, the user will step into the shoes of a professional handball player. Here, the user can choose to play with an existing player, or create a player from scratch.

If the user chooses to create a player from scratch, they'll embark on a slightly different journey, starting from the bottom and working their way up. This will include a short interactive cinematic introduction to the career mode, where multiple stages of the player's junior development will be shown. More information on this mode in the Career Mode tab of the Technical Specifications.

In be-a-manager mode, the user will have access to all the decisions of the team, from choosing the lineup for each match all the way to player transfers and contract renewals.
Each of these can be automated, with the decisions being delegated to the assistant manager.

In this mode, users may opt for an existing manager, or to create their own.

The two forks of career mode have some common elements.
There will be options to perform multiple actions during the player's career, such as:
- Post on social media about the match *(pre- and post-match)*
- Speak with the team about the match *(pre- and post-match)*
- Comment about the match for the club blog *(occasionally, pre-match only)*

In all of the options above, the player's attitude regarding the match cannot be one of rebellion, since this is considered unsportsmanlike conduct, is not allowed by regulatory bodies, and is **–** generally **–** frowned upon by the clubs themselves.

Customization System

DOCUMENT SAMPLE: SECTION NOT INCLUDED

Filing/File Exposition System

DOCUMENT SAMPLE: SECTION NOT INCLUDED

Online Functionalities

DOCUMENT SAMPLE: SECTION NOT INCLUDED

Game Database Specifications

The persistent elements referring to game pieces (such as players, competitions, ...) should be made available to the user publicly, but relatively hidden, such as, for example, a specific folder in the game's directory.
The main idea behind this philosophy is to facilitate modding.
None of these elements should be game-breaking, but it does imply value checks when game elements are created to make sure they're within the expected values.
By default, when a variable is mentioned in the sub-categories below, it's an exposed variable, except when mentioned otherwise such as "**(private) Variable**".

For the effects of internal operation, the game uses the Metric and European standard systems (meter and centimeter, kilograms, 24-hour format, Euros, monthly salary, ...). When information is obtained with the purpose of being presented to the player, it is then translated according to the defined settings.

int game stats (and most attributes) range from **0 to 20**, with 0 being the lowest possible level and 20 being the maximum level regarding that stat or attribute.

For reference, game parameters are mentioned in this document in the following structure:

Parameter name · *type*
Description (if needed).

Example:

First name · *string*
The player's first name.

Games for reference:

- Handball Action Total
- Handball 21
- NHL 21 (for its analog stick-based control system)

Structs, Enums and Custom Classes

General

(enum) Weekdays

- Sunday
- Monday
- Tuesday
- Wednesday
- Thursday
- Friday
- Saturday

Competitions

(enum) Competition types

- League
- Cup
- Friendly Tournament
- *x*-Hours Tournament

If a match doesn't belong to any of these, it's considered to be an isolated friendly match.

(enum) Competition scope

- Domestic
- National
- Continental (Clubs)
- Continental (Nations)
- Intercontinental (Clubs)
- Intercontinental (Nations)
- Worldwide (Nations)
- Olympic (Nations)

(struct) Competition stamp

Stamp texture · *texture*
Stamp location · *Vector3*
Stamp rotation · *Vector3*

(struct) Match

Home team · *Unique ID*
Away team · *Unique ID*
Day of the month · *int*
Month · *int*
Year offset · *int*

The offset of the year, so that the month number may be absolute, and the year will change based on the year offset. For example, if the competition starts in October, a match in October will have a 0 year offset; a match in February will have a 1 year offset, because it's in the year after the competition started.

(struct) Round

Games · *Match[]*

(struct) Schedule

Rounds · *Round[]*

Clubs / Teams

(enum) Emblem position

- Chest left side
- Chest right side
- Chest middle
- Not present

(struct) Kit

Emblem position · *Emblem position*
Shirt type · *int*
Shirt colors · *color[]*
Shorts type · *int*
Shorts colors · *color[]*
Lettering font type · *int*
Lettering font colors · *color[]*
GK Shirt type · *int*
GK Shirt colors · *color[]*
GK Sweatpants type · *int*
GK Sweatpants colors · *color[]*
GK Lettering font type · *int*

GK Lettering font colors · *color[]*

(struct) Captain armband

Armband type · *int*
Armband colors · *color[]*

Rivalries

(enum) Rivalry type

- Historical
- Local
- Competitive

Venues

(enum) Floor material

- Wooden panels
- PVC panels
- Cement
- Outdoor rubber flooring

(struct) Flooring colors

Main flooring · *color*
6-meter area flooring · *color*
Goalpost color · *color*
Line markings · *color*
Line markings out-of-bounds padding · *color*
Center circle flooring · *color*
Out-of-bounds flooring · *color*
Alternate out-of-bounds flooring behind goals · *bool*

(struct) Venue branding elements

Center circle advertising · *texture*
6-meter area advertising · *texture[]*
Half-court advertising · *texture[]*

Fixed advertising boards · *texture[]*
Corner advertising boards · *texture[]*
Long LED (or rotating) board · *texture[]*

People

(enum) Person type

- Player
  An active player (with or without club).
- Non-Player
  An inactive player; p.e. a retired player still connected to a club.
- Referee
  A match official referee.
  Usually coupled with another referee, though not mandatory.
- Technical table operator
  An operator that stays at the technical table.
  Usually coupled with another technical operator, though not mandatory.
- General manager
  A team's active general manager.
- Assistant manager
  A team's general manager's assistant.
- Coach
  A team's coach (physical, mental, tactical, technical, ...).

(enum) Sex

- Male
- Female

(struct) Preference

Target · *Unique ID*
Preference level · *int (10 by default, 20 is favorite, 1 is disliked)*

(struct) Hairstyle

Preset (overrides other options) · *int (0 for no override)*
Back · *int*
Left side · *int*
Right side · *int*
Top · *int*
Fringe · *int*

(struct) Facial hair type

Preset (overrides other options) • *int (0 for no override)*
Sideburn • *int*
Cheek line • *int*
Mustache • *int*
Side (jaw) • *int*
Goatee • *int*
Neck line • *int*

(enum) Shirt tuck type

- Untucked
- Fully tucked
- Right side tucked
- Left side tucked

(enum) Shirt sleeve type

- Short sleeve
- Long sleeve
- Short sleeve w/ undershirt

(enum) Sock type

- Medium socks
- High socks
- Above-knee socks

(struct) General characteristics

Skin tone • *color*
Hairstyle • *Hairstyle*
Hair colors • *color[]*
Hair accessory • *int*
Hair accessory color • *color*
Facial hair type • *Facial hair type*
Facial hair color • *color*
Shirt tuck type • *Shirt tuck type*
Shirt sleeve type (including undershirt) • *Shirt sleeve type*
(Optional, only if undershirt is on) Undershirt color • *color*
Right arm accessory • *int*
Right arm accessory colors • *color[]*
Left arm accessory • *int*
Left arm accessory colors • *color[]*
Right hand taping pattern • *int*

Right hand taping color · *color*
Left hand taping pattern · *int*
Left hand taping color · *color*
Right knee accessory · *int*
Right knee accessory colors · *color[]*
Left knee accessory · *int*
Left knee accessory colors · *color[]*
Sock type · *Sock type*
Sock colors · *color[]*
Shoe type · *int*
Shoe colors · *color[]*

(struct) Facial characteristics

Preset (overrides other options) · *int (0 for no override)*
Head shape · *Vector3*
A 2D axis controls how wide/narrow and tall/short the head is.
A slider controls how deep the head is.
Skin preset · *int*
A preset that allows adding aging signs (p.e. nasolabial folds) or freckles.
Eye preset · *int*
The shape and openness of the eyes.
Eye positioning · *Vector3*
A 2D axis controls how far/close apart and high/low the eyes are.
A slider controls how deep into the face the eyes are.
Eye color · *color*
Eyebrow preset · *int*
The cut and thickness of the eyebrows.
Eyebrow positioning · *Vector3*
A 2D axis controls how far/close apart and high/low the eyebrows are.
A slider controls how deep into the face the eyebrows are.
Eyebrow color · *color*
Nose preset · *int*
The general shape of the nose and nostrils.
Nose positioning · *Vector3*
A 2D axis controls how wide/narrow and high/low the nose bridge is.
A slider controls how deep into the face the nose is.
Mouth preset · *int*
The general shape of the lips.
Mouth positioning · *Vector3*
A 2D axis controls how wide/narrow and high/low the mouth is.
A slider controls how deep into the face the mouth is.
Cheeks preset · *int*
The general shape and format of the cheeks.

Cheekbones positioning • *Vector3*

A 2D axis controls how far/close apart and high/low the cheekbones are.
A slider controls the cheekbones' protrusion.

Ears preset • *int*

The general shape and format of the ears.

Ear positioning • *Vector2*

A 2D axis that controls how wide/narrow and tall/short the ears are.

Ear orientation • *Vector2*

A 2D axis that controls how high/low and inner-/outer-faced the ears are.

Jaw positioning • *Vector3*

A 2D axis controls how tight/loose and lifted/dropped the jawline is.
A slider controls the jaw's squareness/roundness.

Chin preset • *int*

The general shape of the chin.

Cheekbone positioning • *Vector3*

A 2D axis controls how wide/narrow and high/low the chin is.
A slider controls the mandibular protrusion (overbite, underbite, ...).

Neckline type • *int*

The looseness of the submental triangle (under chin).

(struct) Positions

Goalkeeper • *int*
Left Wing • *int*
Left Back • *int*
Center Back • *int*
Right Back • *int*
Right Wing • *int*
Pivot • *int*

(enum) Role type

- No defined archetype
- Calm Goalkeeper

  A goalkeeper that rarely reacts to fakes, but is a bit slower to reach unexpected shots.
- Athletic Goalkeeper

  A goalkeeper that is light on their feet, usually susceptible to mispositioning.
- Traditional Winger

  A winger with a focus on getting the job done without much fuss, adding more efficiency to normal shots, but lessening disposition for trick shots (spin shots, lob shots).
- Flashy Winger

  A winger with a tendency for flamboyant shots, making them more unpredictable but weakening their normal shots.

- Playmaker Back
  A side back with a focus on playmaking, attempting to open spaces and pass to teammates.
- Confident Back
  A side back with a tendency for shots from outside of the 9-meter line.
- Cautious Center
  A center that specializes in shotcalling, coordinating the team perfectly.
- Hyperactive Center
  A center with a knack for taking initiative on opening the defense, often attempting to penetrate the defensive line alone, looking for a shot or pass opportunity.
- Involved Pivot
  A pivot that likes to participate in plays, attempting to be more involved in play setups and frequently coming out of the 6-meter line to provide an additional passing lane.
- Grinder Pivot
  A pivot specialized in using their physical prowess to open passing or shooting lanes deep in the defensive line, playing at the limit of the 6-meter line.

(struct) Preferred hand

Right hand • *int*
Left hand • *int*

(enum) Base body type

- Ectomorph
- Mesomorph
- Endomorph

(struct) Body type

Base body type • *Base body type*
Neck height • *int*
Neck thickness • *int*
Shoulder height • *int*
Shoulder breadth • *int*
Chest measurement • *int*
Upper arm size • *int*
Lower arm size • *int*
Arm length • *int*
Finger length • *int*
Thigh size • *int*
Calf size • *int*
Leg length • *int*
Ankle size • *int*

Foot size • *int*

(struct) Player competition participation

Competition participation • *Unique ID (-1 for no participation)*
Competition performance result • *int*

(struct) Player yearly statistics

Year • *int*
Team • *Unique ID*
Competition • *Unique ID (-1 for none or not applicable)*
Games played • *int*
Goals • *int*
Assists • *int*
Saves • *int*
Save percentage • *float*
Blocks • *int*
Interceptions • *int*
Penalty minutes • *int*

(struct) Statistics

Year • *int*
Team • *Unique ID*
League participation • *Player competition participation*
Domestic cup participation • *Player competition participation*
Continental participation • *Player competition participation*
International participation • *Player competition participation*
Olympic participation • *Player competition participation*
General player yearly statistics • *Player yearly statistics*
League player yearly statistics • *Player yearly statistics*
Continental player yearly statistics • *Player yearly statistics*
International player yearly statistics • *Player yearly statistics*

General

In-Game Metadata

Start date · *year, month, and day*
The definition of the default start date of the game (used for career modes, for example).

Helpers, Colors

Color name · *string*

Color value · *RGB(A) value*

Helpers, Date Formats

European · *dd/mm/yyyy*

North American · *mm/dd/yyyy*

International · *yyyy/mm/dd*

Helpers, Hour Formats

24-hour clock · *24-hour format*

12-hour clock · *12-hour format (AM and PM)*

Military · *hhmm format*

Helpers, Currency *(Exchange rates defined at the time of development)*

Euro

British Pound

United States Dollar

Canadian Dollar

Australian Dollar

Brazilian Real

Japanese Yen

Helpers, Wager Formats

Monthly · *Salary/month*

Weekly · *Salary/week, ~23,0947% monthly salary*

Yearly · *Salary/year, ~1200% monthly salary*

Helpers, Distance

Metric · *meters (and centimeters)*

Imperial (yard) · *yards, ~0.9144m*

Imperial (foot) · *feet (and inches), ~0.3048m*

Helpers, Height

Metric · *meters (and centimeters)*

Imperial (foot) · *feet (and inches), ~0.3048m*

Helpers, Weight

Metric · *kilograms*

Imperial (foot) · *pounds, ~0.4536kg*

Game Pieces

Common Base Information

Attributes

(private) Unique ID · *int*
A unique ID that will represent the game piece anywhere in the game, *regardless of anything*.

ID · *int*
An ID that will represent the game piece anywhere in the game, *starting from its specific type*. For example, a player may have the *Unique ID* of 1000, but the *ID* of 1 because they are first on the list of *Players*.
e.g. This ID may be used to swap teams between leagues, player swaps, copying information from one player to another, among other uses.

Competitions

Attributes

Name · *string*
The competition's actual name.

Local name · *string*
A variation of the name that may be given locally.
e.g. The Portuguese league is sometimes referred to as "Liga Portuguesa", although it's not its official name.

Continent · *string*

Country · *string*

Ruling body · *string*
The ruling body that governs the competition (p.e. IHF, EHF, FAP, ...).

Competition type · *Competition types*

Competition scope · *Competition scope*

Competition importance index · *int*
The index of importance of the competition, such as in the following examples:
- 1st index - Premier league, Champions League
- 2nd index - Second league, Euro League
- 3rd index - Third league

Competition ball · *int*
The *Unique ID* of the competition's ball.
Used to spawn the ball at the beginning of a match.

Trophy model prefab · *prefab path*
The path to the prefab of the competition's trophy, if any.
Used to spawn the trophy in a cup final or league-deciding match, for example.

Game weekdays · *Weekdays[]*
The weekdays in which this competition has matches.

Competition stamps · *Competition stamp[]*
The marks on the players' shirts that indicate the competition.
e.g. In EHF Champions League matches, the teams usually have the logo on their left sleeve.

Type-specific Information

Number of teams · *int*

Has group stage · *bool*
Does the competition have a group stage?
Ideally, the group stage is what separates a league and a cup. A league will only have a group stage.

Has knockout stage · *bool*
Does the competition have a knockout stage?
A cup should only have a knockout stage.
The Champions League format, for example, implies both a *group stage* and a *knockout stage*.

Teams per group · *int*

The amount of teams per group.
In a league, the group has all the teams in the competition.

Schedule · *Schedule*
The schedule for the competition, according to its rules regarding scheduling.
This should be regenerated every year in career modes to ensure variety.

Points obtained · *int[]*
The amount of points obtained by winning, drawing, and losing.

Sorting rules · *condition[]*
The competition's sorting rules for draws in points.
e.g. For a league, the first sorting rule would probably be
points / possible points
where
*possible points = games played * points for win*.

Winner eligible for continental cup · *bool*
Should the winner of the competition be eligible to play in the continental cup?
In some domestic or continental cups, the winner is automatically eligible to participate in the following year's continental cup.

Amount of teams going to major continental cup · *int*
Amount of teams that finish at the top of the table that become eligible for the highest level continental cup.

Amount of teams going to minor continental cup · *int*
Amount of teams that finish just after the major continental spots, thus becoming eligible for the second-highest level continental cup.

Amount of teams relegated · *int*
The amount of teams that finish at the bottom of the table that are sent to the competition directly below in the importance index.

Clubs / Teams

Attributes

Team name (full) · *string*
The team's full name (with abbreviations only for secondary parts of the name).

e.g. FC Porto.

Team name (abbreviated) · *string*
The team's abbreviated name.
e.g. Porto.

Team name (three-letter abbreviation) · *string*
The team's three-letter abbreviation, according to its name (not necessarily the letters that each word in the name starts with).
e.g. FCP.

Team/fans nicknames · *string[]*
The nicknames given to the team and fans.
e.g. Portistas, Dragões.

Primary team color · *color*
The team's primary color (may be used for venue seats, supporter flags and banners, shirts of supporters not using the team's official jersey, goalposts, ...).

Team captains · *Unique ID[]*
The *Unique ID*s of the team's captain and alternate captains. Only the first player on the list (the actual captain) should wear the armband.

Secondary team color · *color*
The team's secondary color (may be used for venue seats, supporter flags and banners, ...).

Team emblem · *texture*
The club's emblem (mostly to be shown in menus and overlays, and to be placed in the team kit).

League · *Unique ID*
The league that the team participates in.
e.g. Andebol 1.

Domestic cup · *Unique ID*
The cup that the team participates in domestically.
e.g. Taça de Portugal.

Continental cup · *Unique ID*
The cup that the team participates in continentally.
e.g. EHF Champions League.

Nation · *Unique ID*
The nation that the team participates in domestically.

City · *string*
The city the team is registered in.

Date founded · *DateTime*
The club’s foundation date.
Mostly used for special occasions such as club anniversaries and for some cheer squads’ signs.

Venue · *Unique ID*
The venue the team plays in regularly.
If, for example, a team is playing a continental cup match and its venue isn’t within regulation, the eligible venue closest to the team’s regular location should be used.

Popularity · *int*
The team’s popularity.
This affects the % of attendance (adjusted to the venue’s capacity and the supporter profile.

Supporter attendance · *int*
The team’s ability to pull supporters to watch a game live.
Until a certain level of attendance, the supporters have a much bigger attendance when the game is a continental cup match or an important league game, and much less attendance when it’s a game against a weaker team.
e.g. Supporter attendance at 13 out of 20
FC Porto v FC Barcelona (100% attendance with Super Dragões attending)
FC Porto v SL Benfica (100% attendance with Super Dragões attending)
FC Porto v Águas Santas (60% attendance without Super Dragões attending)

Supporter patience · *int*
The level of patience the supporters have.
Affects the pressure when the team isn’t satisfying expectations, attendance drops when the team is on a bad form, for example.

Home kit · *Kit*
The kit worn by the team in all games, provided there are no color conflicts with the opposing team’s kit.
When a supporter is wearing a team shirt, it’s also very likely this kit’s shirt.

Away kit · *Kit*
The kit worn by the team in away games when there is a color conflict with the opposing team's home kit.

Third kit · *Kit*
The kit worn by the team in away games when there is a color conflict with the opposing team's home kit in special occasions (such as continental cup games).

Captain's armband · *Captain armband*
The armband worn by the team's captain in matches.

Starting transfer budget · *int*
The amount of money set aside for player transfers that the team starts a career mode with.

Starting contract budget · *int*
The amount of money set aside for player contracts that the team starts a career mode with.

Transfer budget · *int*
The amount of money set aside for player transfers that the team has during a career.

Contract budget · *int*
The amount of money set aside for player contracts that the team has during a career.

History and Statistical Logs

Club trophies · *Dictionary<Competition (Unique ID), amount of times (int)>*
The trophies that the club has won throughout the years.

Rivalries

Attributes

Rivalry type · *Rivalry type*
The type of rivalry between the two teams.

Historical means that there used to be a flaming rivalry between the two teams.
e.g. Fenerbahçe SK v Galatasaray SK
Local means that the two teams are from the same city or neighboring locations.
e.g. Everton FC v Liverpool FC
Competitive means that the two teams have recently developed a competitive rivalry, not associated with history or location.
e.g. FC Porto v SL Benfica

First team involved · *int*
The *Unique ID* of the first team involved in the rivalry.

Second team involved · *int*
The *Unique ID* of the second team involved in the rivalry.

Intensity · *int*
The level of intensity of the rivalry. This can range from a mild provocation between supporter groups during the match up to supporters getting involved in fights before/after the game.
This mainly affects the content of the news generated regarding the match in career modes, for example.

Venues

Attributes

Venue name · *string*

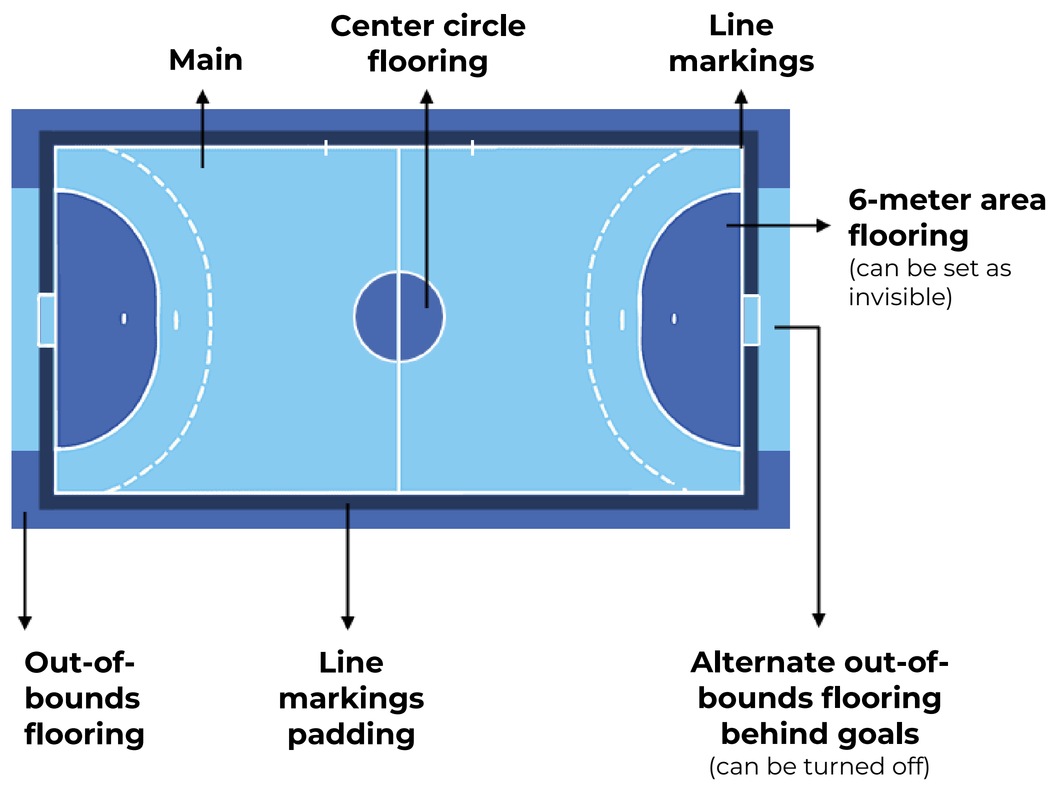
Venue location · *string*
The venue's location (city, country).

Venue prefab · *prefab path*
The path to the venue's prefab.
This will be used to instantiate the venue when a match is starting.

Court floor material · *Floor material*
The physical material that will be applied to the whole play court.

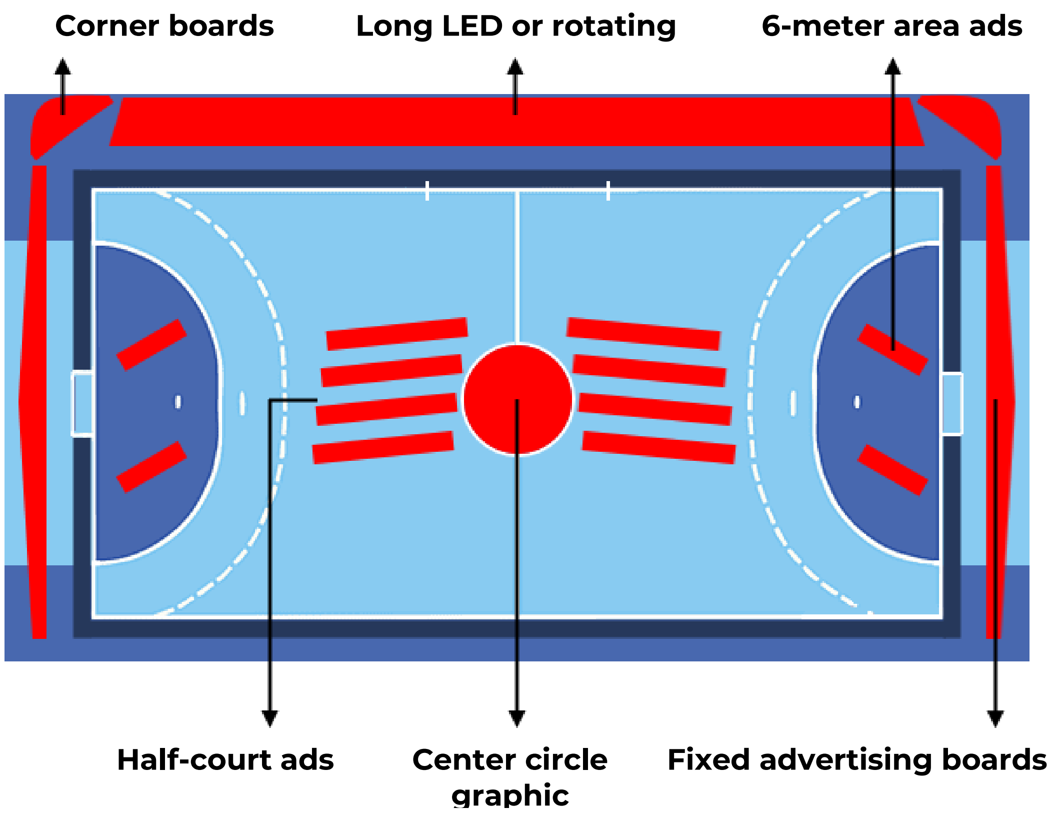
Court flooring colors · *Flooring colors*

The colors that should be applied to the floor, according to the following figure:


Main
Center circle flooring
Line markings
6-meter area flooring (can be set as invisible)
Out-of-bounds flooring
Line markings padding
Alternate out-of-bounds flooring behind goals (can be turned off)

When one of the options specified above is turned off, it should become fully transparent, so that the colors below are visible.
For example, if the 6-meter area's color is invisible, the 6-meter area will have the same color as the main flooring.

Venue branding · *Venue branding elements*
The sponsorship branding elements to be used in a specific match.



Every one of these ads is optional, and the branding doesn't necessarily need to be an advertisement, it may also be branding for the club.
e.g. Dragão Arena, FC Porto's venue, has the Porto logo in the center circle.

Note about venues

Each venue's 3D model should be included in a prefab (that should be referenced in the **Venue prefab**), which will serve as the base to load the venue.
Each of the venue's supporter spots (be it seated and standing) must have an empty GameObject with a specific tag (maybe "Seated" and "Standing"?) so that the supporters can be filled in automatically. Same for media people, photographers, cameramen and other external elements.
The prefab should accommodate the court's dimensions with empty space in the exact middle so as to not spawn anything on top of the court.
The court will be automatically generated (at position (0, 0, 0) and the pivot point being the center) with the following properties:

Court dimensions · *Vector2*

The dimensions of the court.
40m x 20m by default.

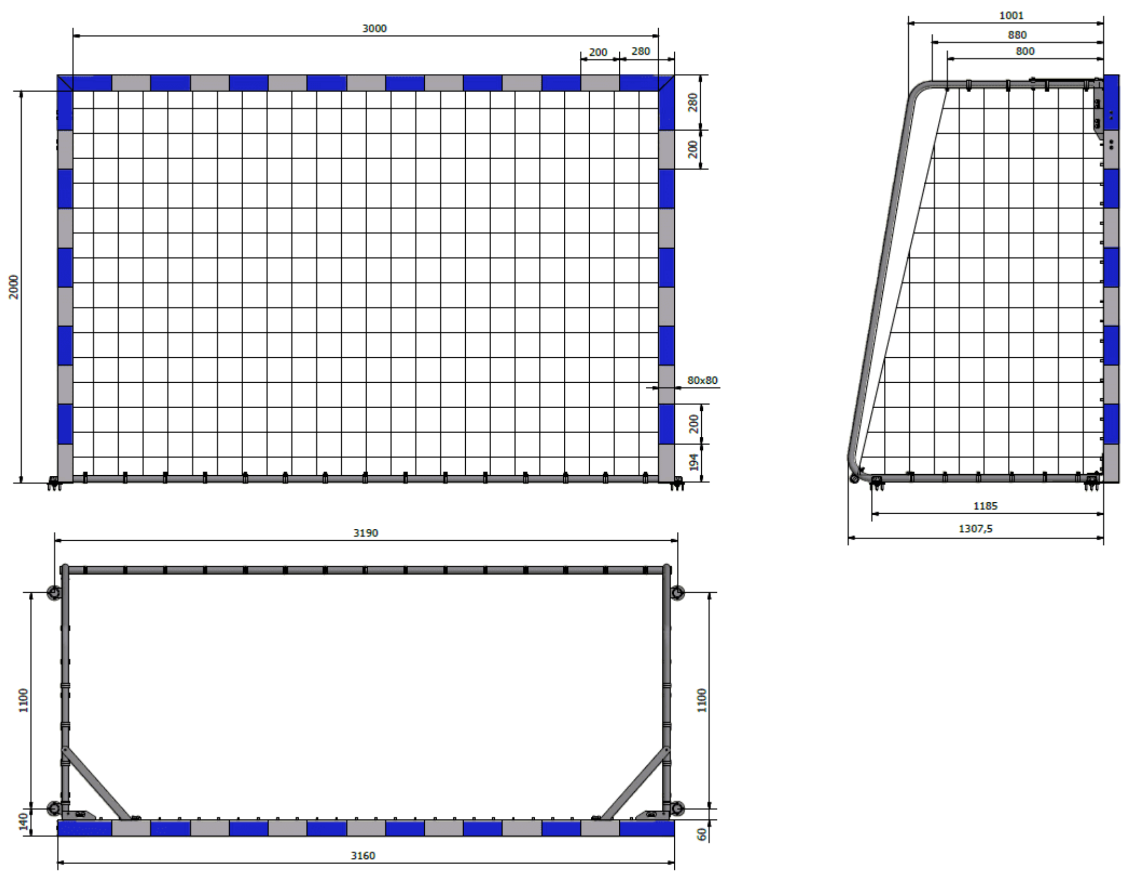
The **lines** should be built into the court, as the dimensions of the court influence the size of the areas.
Default lines:

> **4m line**
> 4m away from the center of the goal, 50cm length
> **6m line**
> 6m away from each point of the goal
> **7m line**
> 7m away from the center of the goal, 1m length
> **9m line**
> 9m away from each point of the goal
> **Center line**
> Going through the length of the court in the middle
> **Center circle**
> 3m diameter circle centered in the exact middle of the court
> **Substitution lines**
> 4.5m away from each side of the middle of the court; always on the substitution benches' side

Out-of-bounds area · *Vector2*
The area of the out-of-bounds markings on the floor. This area is rendered behind the main court, so it should be bigger to accommodate the out-of-bounds area.
45m x 25m by default, which amounts to **2.5m** padding.

Line markings thickness · *int*
The thickness of the line markings on the flooring, accounting for the actual line as the middle and expanding outwards on both sides.
15cm thick by default.

Goal dimensions · *Vector3*
The dimensions of the goals on each side of the court.
3m x 2m x 1m with **8cm goalposts** by default. See the figure below for more specifications.



For the goal to only be counted if the ball fully passes the goal line, the collider may be put with the offset of the ball's diameter; that way, the ball will only hit the collider once it's fully inside.

People

Personal Info

Type of person · *Person type*
The person's type.
Used to determine whether the person is a player, manager, or another type of person within or outside of a club.

First name · *string*
The person's first given name.

Last name · *string*
The person's last given name.

Sex · *Sex*
The person's sex (male or female).

Handle · *string*
The person's social media handle.
Used for career mode news and social media posts.

Player photo · *texture*
The person's photo. Most likely, these will be generated according to the person's in-game appearance and not previously placed.

(optional) Common name · *string*
The name that the person is commonly known as. This may be, for example, a nickname or the player's first name (if they're known by their first name instead of last).

Shirt name · *string*
The name on the player's shirt.

Shirt number · *int*

International shirt number · *int*

Position(s) · *Positions*
The player's level of comfort in each position.

Role(s)/Type(s) · *Role type*
The player's archetype and general role within any team.

Preferred hand · *Preferred hand*
The player's preferred hand while playing.
When the player's level of comfort with one of the hands is between 5 and 12, the player only uses that hand for passing and not shooting.

Date of birth · *DateTime*
The person's date of birth. The DateTime format facilitates date calculation when needed.

Nationality · *Unique ID*

The nation that the player is sworn to. This may be the nation where the player was born or an acquired nationality through migration, but it should always be the nation the player is eligible to play for.

Current ability · *int*
The player's current ability as opposed to their potential ability.

Potential ability · *int*
The ability the player may be able to reach if developed in perfect conditions.

Market value · *int*
The base value this player is worth in the transfer market.

Wage · *int*
The player's salary.

Contract expiration · *DateTime*
The expiration date for the person's contract with their current club.

Physical Info

Height · *float*

Weight · *float*

Body type · *Body type*
The player's base body type (ectomorph, mesomorph, or endomorph).

Psychological Stats

Aggressiveness · *int*
The player's general aggressiveness, both when defending and attacking (players with higher aggressiveness are more prone to causing charge fouls).

Discipline · *int*

The player's ability to stay disciplined in higher-risk games (p.e. important matches) or stressful situations, such as 7-meter throws or games in which there is a lot of physical contact with opponents.

Defensive work rate · *int*
The player's work rate when the team is defending (without ball possession).

Offensive work rate · *int*
The player's work rate when the team is in possession of the ball.

Physical Stats

Balance · *int*
The player's capability to maintain balance or shot/pass accuracy when under physical pressure from an opponent.

Strength · *int*
The player's brute force, for holding/pushing opponents or keeping defenders away.

Acceleration · *int*
The player's general ability to reach peak velocity from a relatively slow start.

Speed · *int*
The player's general peak speed.

Agility · *int*
The player's ability to move their body aerodynamically, both doing body feints or moving while in the air.

Jumping · *int*
The player's power to jump, both vertically (mainly for back-liners) and forward (mainly for wingers).

Stamina · *int*
The player's stamina, affecting multiple factors, such as speed, shot and pass accuracy, decision-making, discipline, among others.

Goalkeeping Stats

Upper reach · *int*
The goalkeeper's ability to defend shots at the upper side of the goal.

Lower reach · *int*
The goalkeeper's ability to defend lower shots.

Elasticity · *int*
The goalkeeper's elasticity when attempting to get to an out-of-reach shot.

Reactions · *int*
The goalkeeper's ability to react to an attacker's unexpected move or shot.

Defensive Stats

Blocking · *int*
The player's skill at blocking shots when defending.

Hand-eye · *int*
The player's coordination from the point they have a thought or reaction to their physical response.

Defensive awareness · *int*
The player's positioning when defending, taking into account all of the attacker's possibilities (to try and block their shooting and passing lanes).

Offensive Stats

Shot power · *int*
The player's raw power when shooting.

Shot accuracy · *int*
The player's accuracy when performing basic shots.

Technique · *int*

The player's general wrist looseness and skill, mainly regarding trick shots (spin shot, lob shot) and passes.

Passing · *int*
The player's skill at passing to other players.

Offensive awareness · *int*
The player's positioning, during free play and, more specifically, during planned plays.

Manager Stats

Patience · *int*
The manager's patience when players commit mistakes.
This affects, for example, the amount of shots that a player can miss and the amount of 'easy' saves a goalie can miss before they're subbed.

Calmness · *int*
The manager's ability to stay calm on the bench when faced with player mistakes or adverse situations in a match.

Tendency to protest · *int*
The manager's tendency to protest decisions, especially 2-minute suspensions or 7-meter penalties.

Motivating · *int*
The manager's skill at motivating the team. This affects mostly game simulations, and applies to major stops in the match (timeouts, half-time, ...).

Refereeing Stats

Line marking tolerance · *int*
The referee's tolerance towards stepping on line markings.
This also affects the referee's tolerance towards players (mainly wingers) holding their jump until they're touching the 6-meter area before shooting or passing.

Roughness tolerance · *int*
The referee's tolerance towards rough physical play, as long as it's within the rules.

Protesting tolerance · *int*
The referee's tolerance towards players protesting their decisions, especially influential decisions such as 2-minute suspensions or 7-meter penalties.

History and Statistical Logs

Club preferences · *Preference[]*
The player's favorite and disliked clubs.

People preferences · *Preference[]*
The player's favorite and disliked people.
This may have an effect on hiring a player. Players may be more eager to move to a team when a preferred person plays for it, and may be hesitant when there's a disliked person.
This will also affect players' social media group appearances (a player will sometimes post with a preferred teammate).

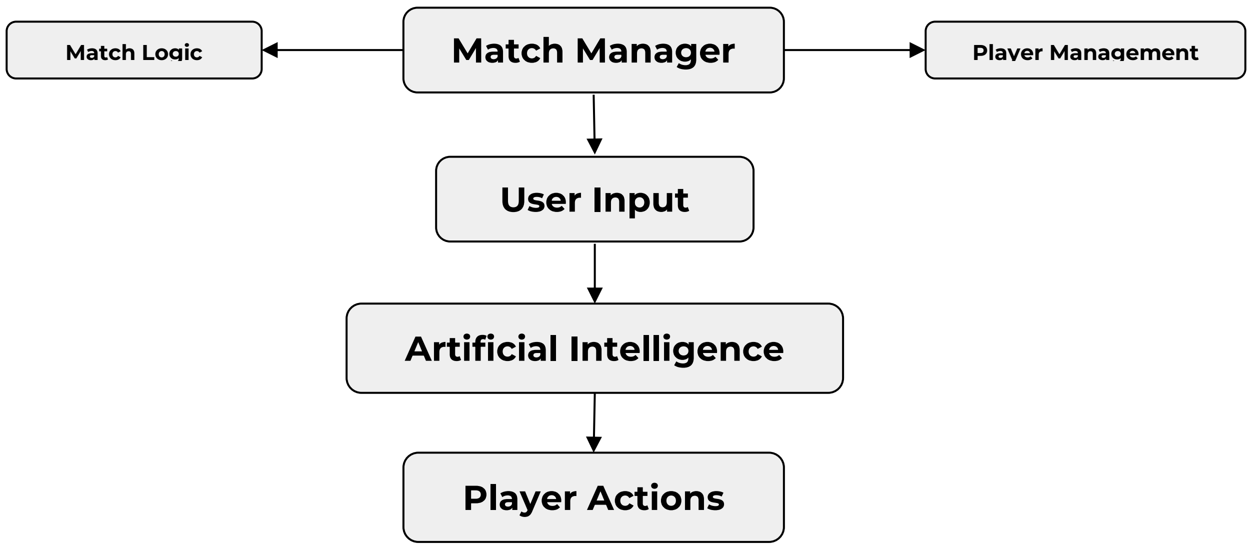
Clubs played for · *Unique ID[]*
The clubs that the player played for in their career.

Statistics · *Statistics[]*
The player's statistics over the years.
This includes the current year.
General statistics should be calculated by adding up the sub-divided statistics (league, cups, ...).

Technical Specifications

Control System

Control System Flowchart



Match Manager

Among other tasks, the *Match Manager* periodically and very frequently cycles through the players in a specific order, defined by the players' stats. This allows for players with faster reactions to act (or react) more quickly than ones with lower reactionary stats.
For example, it gives an edge to a fast-reacting goalkeeper facing a slow or predictable player.

In general, this layer manages the action and reaction cycles during the match, from players and referees to match logic, technical operators and team staff.

User Input

User Input is where the user's actions and inputs in the real world come in.
This layer must come after the *Match Manager* to avoid instant reactions. As such, any reaction by another player will only be started in the next reaction cycle, circumventing action overlaps and increasing enjoyability and realism.
It also means that the player (and AI) is able to use others' reactions to fake or change their move to their own advantage.

This layer directly controls and overrides the next layer, *Artificial Intelligence*.

Artificial Intelligence

Artificial Intelligence is the layer that tells players what to do based on their circumstance, stats and characteristics.
As mentioned above, the *User Input* overrides this layer's behavior. That is, the player's calculated movement is canceled if the user is directly controlling the player.

However, the *Artificial Intelligence* layer still acts when overridden by user input: it's responsible for restricting the *Player Actions* by opposing the user input to the controlled player's stats. That is, if the user decides to have the player jump, the *Artificial Intelligence* layer receives the command and handles the build-up for the jump, as well as how the jump is (if it's vertical or more directed, if it's a higher jump, if the jump happens in perfect conditions, ...) depending on the player's position, situation and stats.

Player Actions

The *Player Actions* are a group of actions, movements and animations – pre-defined *or* programmatically executed –, ordered by the *Artificial Intelligence* layer, with a set of parameters.
These allow the animations to be intertwined and performed seamlessly, with consistency and accuracy according to what the match requires.

Game Modes

DOCUMENT SAMPLE: SECTION NOT INCLUDED

Customization System

DOCUMENT SAMPLE: SECTION NOT INCLUDED

File System Handling

File System

Each save file has its own version of the exposed files, so that they are also flexible even after a saved game has been started and **do not interfere with the main files or other saves**.

The editable parameters should be as centralized as is feasible, within their own categories.

Summarizing, a saved game (career, tournament, ...) should be a folder. All its files have the same name, so as to facilitate obtaining data for a specific save, with the extension referring to the nature of the file, although **none of these files are encrypted and all of them can be edited with a standard text editor**.
Inside the folder are the saved game's core, i.e. the information required for data persistence, along with the files containing the parameters, in a similar fashion to the following example:

```
↳ <saved game folder path>
  ↳ SavedGame1
    ↳ SavedGame1.save
    ↳ SavedGame1.competitions
    ↳ SavedGame1.clubs
    ↳ SavedGame1.rivalries
    ↳ SavedGame1.venues
    ↳ SavedGame1.people
```

The example above should serve only as a model for creating the structure of the generic file system and those of saved games.

Networking Structure

DOCUMENT SAMPLE: SECTION NOT INCLUDED

Design References

DOCUMENT SAMPLE: SECTION NOT INCLUDED

Sport Overview

Sport and Rule Summary

DOCUMENT SAMPLE: SECTION ABRIDGED