# Sportimo Database Documentation

Contents

[1 Sportimo Database Documentation 1](#_Toc515026833)

[1.1 Collections 1](#_Toc515026834)

[1.2 Collection detailed 2](#_Toc515026835)

[1.3 Data Models 7](#_Toc515026836)

[1.3.1 Answer 8](#_Toc515026837)

[1.3.2 Article 8](#_Toc515026838)

[1.3.3 Competition 8](#_Toc515026839)

[1.3.4 FavQuestion 8](#_Toc515026840)

[1.3.5 GamecardDefinition 9](#_Toc515026841)

[1.3.6 GamecardTemplate 10](#_Toc515026842)

[1.3.7 Message 11](#_Toc515026843)

[1.3.8 Player 12](#_Toc515026844)

[1.3.9 Poll 12](#_Toc515026845)

[1.3.10 Question 13](#_Toc515026846)

[1.3.11 Scheduled\_Matches 13](#_Toc515026847)

[1.3.12 Sponsor 15](#_Toc515026848)

[1.3.13 Standing 16](#_Toc515026849)

[1.3.14 Stats-mod 16](#_Toc515026850)

[1.3.15 Team 16](#_Toc515026851)

[1.3.16 User 17](#_Toc515026852)

[1.3.17 Achievement 19](#_Toc515026853)

[1.3.18 UserStats 19](#_Toc515026854)

[1.3.19 UserActivity 19](#_Toc515026855)

[1.3.20 UserGamecard 20](#_Toc515026856)

[*1.3.21* Star 22](#_Toc515026857)

[1.3.22 Purchase 22](#_Toc515026858)

[*1.3.23* Taunt 22](#_Toc515026859)

[1.3.24 UserTaunt 23](#_Toc515026860)

[1.4 Database Footprint and Size projection 23](#_Toc515026861)

## Collections

A grouping of **MongoDB** documents. A **collection** is the equivalent of an RDBMS table. A **collection** exists within a single **database**. **Collections** do not enforce a schema.

The Sportimo database includes the following collections and each one will be explained in section 6.2:

* Achievements
* Answers
* Articles
* Competitions
* Earlyaccessusers
* Favquestions
* Gamecarddefinitions
* Gamecardtemplates
* Gameserversettings
* Introquestions
* Messages
* Players
* Polls
* Pools
* Prizes
* Questions
* Scheduled\_matches
* Scores
* Settings
* Standings
* Subscriptions
* Taunts
* Teans
* Useracctivities
* Usergamecards
* Users
* Usertaunts

## Collection detailed

|  |
| --- |
| Achievements |

The list of the game achievements.

|  |
| --- |
| Answers [Not Used Yet] |

Placeholder for future use

|  |
| --- |
| Articles |

The list of news articles to view in the client news section

|  |
| --- |
| Competitions |

All leagues viewable in the game.

|  |
| --- |
| Earlyaccessusers |

All leagues viewable in the game.

|  |  |
| --- | --- |
| email: | The email of the user |
| email\_sent: | Have we sent invite |
| code: | The early access code |

|  |
| --- |
| Gamecarddefinitions |

The game cards that are available for each match. Created separate in order for the moderator to make custom tailored changes for each match.

|  |  |
| --- | --- |
| matchid: | The id of the match it relates to. |
| gamecardTemplateId: | The id of the template that propagates from |
| creationTime: | The utc time the card was played |
| primaryStatistic: | The primary statistic that this card references. |
| activationTime: | The utc time the card was activated. |
| terminationTime: | The utc time the card was terminated. |
| cardType: | The type of the card. |
| Options: | List of predefined options, regarding teams and win conditions. |

|  |
| --- |
| Gamecardtemplates |

Game card templates are the initial setup for the game cards that will later on be cloned for use in each match. They have the basic setup for all game cards that will be in play. For property details see: Gamecarddefinitions.

|  |
| --- |
| Introquestions |

List of skill questions asked before match in order for the users to be able to win prizes.

|  |
| --- |
| Messages |

The list of messages the dashboard moderators have sent to the users.

|  |
| --- |
| Players |

This is all the data stored from stats regarding player info. Warning these are not the users of the game. These are the football players and their stats.

|  |
| --- |
| Polls |

The polls create from the dashboard.

|  |  |
| --- | --- |
| tags: | To what section this poll is attached |
| text: | The question of the poll |
| total\_votes: | The total votes on the poll |
| Cerated: | The utc time that the poll was created |
| answers: | The list of poll answers |
| answers.text: | The answer’s text |
| answers.img: | The answer’s image |
| answers.percent: | The percent that has this answer |
| answers.votes: | The votes received by this answer |

|  |
| --- |
| Pools |

Internal data used to create leaderboards.

|  |
| --- |
| Prizes |

List of prizes used for assigning them in leaderboards

|  |
| --- |
| Questions [Not Used Yet] |

Placeholder for future use

|  |
| --- |
| Scheduled\_matches |

This is the list of the scheduled matched data. All matches scheduled in the dashboard get an entry here and consecutively are handled by the service or a moderator directly.

|  |  |
| --- | --- |
| home\_team: | The id of the home team |
| away\_team: | The id of the away team |
| start: | The utc time that the match starts |
| competition: | The Id of the competition that this match belongs to |
| settings: | Settings for the client. |
| moderation: | Sets various settings regarding moderation and if it will be based on a feed or manual handling. |
| timeline: | The list of timeline events |
| headtohead: | The results of previous classes between these two teams |
| stats: | Stats created by the timeline events. These are the stats checked in order to validate user wins. |
| Completed: | Whether the match has ended or not |
| State: | The segment the match currently is |
| Time: | The time of the match. This is the game clock not actual time. |
| Away\_score: | The score for the away team |
| Home\_score: | The score for the home team |
| Visiblein: | Countries this match is visible |
| guruStats | The guru stats as gathered by the service |

|  |
| --- |
| Scores |

The list of score that each user has for each match that has participated in.

|  |
| --- |
| Settings |

The settings object that initializes the clients.

|  |
| --- |
| Standings |

The standings for each league as received by the stats.com service

|  |
| --- |
| Subscriptions [Not Used Yet] |

Placeholder for future use

|  |
| --- |
| Taunts |

The list of the taunts available for users to use as send messages in game.

|  |
| --- |
| Settings |

The settings object that initializes the clients.

|  |
| --- |
| Teams |

The team data as received by the stats.com service.

|  |
| --- |
| Useractivities |

This collection tracks users’ movements. An activity is recorder each time the user visits a match.

|  |  |
| --- | --- |
| room: | The match id |
| user: | The user id |
| createdAt: | The utc time of the first visit |
| lastActive: | The last utc time that the user was active |
| overallCardsPlayed: | How many overall cards the user has played |
| cardsPlayed: | How many total cards the user has played |
| instantCardsPlayed: | How many instant cards the user has played |

|  |
| --- |
| Usergamecards |

The list of user played cards.

|  |
| --- |
| Users |

The list of all the game’s users’ data.

|  |  |
| --- | --- |
| lastActive: | The utc time that the user was last |
| createdAt: | The utc time that this user was created |
| admin: | If the user is an administrator/moderator |
| username: | The username of the user |
| password: | The hashed password of the user |
| isOnline: | If the user is presently online |
| favoriteteams: | The list the user’s favorite teams |
| blockedusers: | The list of users this user has blocked |
| achievements: | The list of this user’s achievements |
| level: | The level of the user. This is calculated based on achievements |
| rankingStats: | The best ranking stats for the user |
| subscriptionEnd: | The utc time this user’s subscription will end |
| customerType: | The type of user. Paid or free |
| country: | The country of this user |
| pushSettings: | List of various push permissions |
| pushToken | The push token of the user |
| Unread: | Number of unread messaged the user has |
| Inbox: | List of ids of messages directed to this user |
| Stats.matchesvisited: | User Stats: How many matches the user has visited |

|  |
| --- |
| Usertaunts |

The list of all the user sent taunts and messages.

## Data Models

A set of schemata have been defined in Mongoose to serve the data access layer of the Game server. Most data types are determined, except in those cases where a certain degree of extensibility is sought, where the property type is set to Mixed.

The schemata are listed below in alphabetical order as a reference. These are not full Mongoose schema definitions but rather partial code snippets, however they contain all the properties and data types, plus the sub-models, where those exist.

### Answer

Not actively used.

*{*

*userid: {*

*type:String,*

*ref:'users'*

*},*

*questionid: String,*

*matchid: String,*

*answerid: String,*

*created: { type: Date, default: Date.now }*

*}*

### Article

Represents a news item regarding a league, a team or a player.

*{*

*publication: { type: Schema.Types.Mixed },*

*publishDate: { type: Date},*

*type: {type: String},*

*photo: { type: String },*

*tags: { type: Schema.Types.Mixed },*

*created: { type: Date, default: Date.now }*

*}*

### Competition

Represents a league or cup (national or international) soccer competition. An instance (document) refers to a specific season (e.g. 2017 – 2018)

*{*

*name: { type: Schema.Types.Mixed },*

*logo: { type: String },*

*parserids: { type: Schema.Types.Mixed },*

*season: String,*

*graphics: { type: Schema.Types.Mixed },*

*visiblein: [String],*

*status: {type:String},*

*created: { type: Date, default: Date.now }*

*}*

### FavQuestion

Not actively used. Reserved for future use.

*{*

*info: String,*

*text: { type: Schema.Types.Mixed },*

*answers: [answer],*

*type: {type: String},*

*img: { type: String },*

*status: {type:Number, default: 0},*

*sponsor: { type: Schema.Types.Mixed },*

*}*

#### Answer

Sub-schema of FavQuestion, holds a possible choice to a multiple choice FavQuestion.

*{*

*text: { type: Schema.Types.Mixed },*

*img: String,*

*points: Number*

*}*

### GamecardDefinition

Represents a definition for a gamecard that the user can play, in a certain match. User gamecards result from inheriting from the documents of this collection.

*{*

*matchid: String,*

*gamecardTemplateId: String, // reference to the gamecard template that this definition represents, optional*

*title: Schema.Types.Mixed, // card title*

*image: Schema.Types.Mixed, // icon image*

*text: Schema.Types.Mixed,*

*primaryStatistic: String, // the primary statistic that this card is affected from, in order to be shown on the card (averages for each team)*

*guruAction: String, // Handling action of guru stats*

*// Trigger specifications*

*activationLatency: Number,*

*specialActivationLatency: { DoublePoints: Number, DoubleTime: Number },*

*duration: Number, // instant gamecards only;*

*appearConditions: [Schema.Types.Mixed],*

*winConditions: [Schema.Types.Mixed],*

*terminationConditions: [Schema.Types.Mixed], // when a played card is terminated and pending resolution before put out of play*

*options: [optionDefinition], // mainly instant gamecards*

*// Specs for awarding points to winning cards*

*pointsPerMinute: Number, // overall gamecards only; the rate by which the startPoints get increased or decreased in time*

*startPoints: Number,*

*endPoints: Number,*

*// States and state times*

*cardType: { type: String, enum: ['Instant', 'Overall', 'PresetInstant'] },*

*maxUserInstances: Number, // maximum number of times a user may play this card*

*creationTime: Date,*

*activationTime: Date,*

*terminationTime: Date,*

*isActive: { type: Boolean, default: true },*

*isVisible: { type: Boolean, default: true }, // overall cards only; true if it can appear on clients' list of gamecard, false if it can't*

*status: 0 // 0: pending activation, 1: active, 2: terminated (dead)*

*}*

#### OptionDefinition

A sub-schema of GamecardDefinition, represents an option of a multiple-options game card.

*{*

*isVisible: { type: Boolean, default: true },*

*optionId: String,*

*text: Schema.Types.Mixed,*

*startPoints: Number,*

*endPoints: Number,*

*pointsPerMinute: Number,*

*activationLatency: Number,*

*duration: Number,*

*winConditions: [Schema.Types.Mixed],*

*terminationConditions: [Schema.Types.Mixed]*

*}*

### GamecardTemplate

Represents a template for all gamecard definitions. When a match is scheduled, a gamecard definition is created out the respective document of this collection, that serves as a prototype.

*{*

*isActive: Boolean,*

*title: Schema.Types.Mixed, // card title*

*image: Schema.Types.Mixed, // icon image*

*text: Schema.Types.Mixed, // text template with placeholders: [[player]] for player name, [[team]] for team name*

*primaryStatistic: String, // the primary statistic that this card is affected from, in order to be shown on the card (averages for each team)*

*guruAction: String, // Handling action of guru stats*

*// Trigger specifications*

*activationLatency: Number, // seconds between the gamecard's creation and activation*

*specialActivationLatency: specialActivationLatencyType, // seconds between the gamecard's special ability creation (double time, double points) and activation*

*duration: Number, // seconds between the wildcard's activation and termination*

*appearConditions: [Schema.Types.Mixed], // the card will appear (start its lifetime in a pending state 0) when all the conditionsToAppear are met.*

*winConditions: [Schema.Types.Mixed], // the wildcard wins when all win conditions are met*

*terminationConditions: [Schema.Types.Mixed], // the card is terminated when any of the terminationConditions is met, or the duration is over (if not null).*

*options: [optionTemplate],*

*isVisible: { type: Boolean, default: true },*

*// Awarded points specs*

*pointsPerMinute: Number,*

*startPoints: Number,*

*endPoints: Number,*

*cardType: { type: String, enum: ['Instant', 'Overall', 'PresetInstant'] },*

*}*

#### OptionTemplate

Represents an option of the gamecard template schema and is a sub-schema of the GamecardTemplate schema.

*{*

*isVisible: Boolean,*

*optionId: String,*

*text: Schema.Types.Mixed,*

*startPoints: Number,*

*endPoints: Number,*

*pointsPerMinute: Number,*

*activationLatency: Number,*

*duration: Number,*

*specialActivationLatency: specialActivationLatencyType,*

*winConditions: [Schema.Types.Mixed],*

*terminationConditions: [Schema.Types.Mixed]*

*}*

### Message

Represents the structure of a message that is sent as an in-app message to one or multiple users.

*{*

*sender:{*

*type:String,*

*ref:'users'*

*},*

*recipients:[{*

*type:String,*

*ref:'users'*

*}],*

*img: { type: String },*

*title: {type:mongoose.Schema.Types.Mixed},*

*msg: {type:mongoose.Schema.Types.Mixed, required:true},*

*data: {type:String},*

*read: {type:Number},*

*created: { type: Date, default: Date.now }*

*}*

### Player

Represents a player of a soccer team. Players participate in the match events in the scheduled\_match timeline, and are also returned as team members in a team view.

*{*

*name: { type: Schema.Types.Mixed },*

*firstName: { type: Schema.Types.Mixed },*

*lastName: { type: Schema.Types.Mixed },*

*uniformNumber : { type: String },*

*stats: { type: Schema.Types.Mixed },*

*pic: { type: String },*

*position: { type: String },*

*personalData: { type: Schema.Types.Mixed },*

*parserids: { type: Schema.Types.Mixed },*

*teamId: {*

*type: String,*

*ref: 'teams'*

*},*

*created: { type: Date, default: Date.now },*

*updated: { type: Date }*

*}*

### Poll

Represents a poll that the app is presenting to users for them to complete. Polls are usually referring to soccer competitions, matches, teams or players.

*{*

*text: { type: Schema.Types.Mixed },*

*answers: [answer],*

*matchid: String,*

*type: {type: String},*

*img: { type: String },*

*total\_votes: {type:Number, default: 0},*

*hasAlreadyVoted: Number,*

*hasAnswered: String,*

*voters: [Schema.Types.Mixed],*

*status: Number,*

*tags: { type: Schema.Types.Mixed },*

*sponsor: { type: Schema.Types.Mixed },*

*created: { type: Date, default: Date.now }*

*}*

#### Answer

Sub-schema of the Poll schema, encodes a possible choice for the poll question to choose from.

*{*

*text: { type: Schema.Types.Mixed },*

*img: String,*

*votes: {type:Number,default:0},*

*percent: {type:Number,default:0}*

*}*

### Question

Similar to the Poll concept, represents a multiple choice type of question that the moderator may ask the users to answer.

*{*

*text: { type: Schema.Types.Mixed },*

*answers: [answer],*

*matchid: String,*

*type: {type: String},*

*img: { type: String },*

*status: Number,*

*correct: ObjectId,*

*sponsor: { type: Schema.Types.Mixed },*

*userAnswer: String,*

*created: { type: Date, default: Date.now }*

*}*

#### Answer

Sub-schema of the Question schema, represents a choice to the respective multiple choice type of question.

*{*

*text: { type: Schema.Types.Mixed },*

*img: String,*

*points: Number,*

*answered: {type:Number,default:0}*

*}*

### Scheduled\_Matches

Represents a central concept of the database overall schema, the soccer match. Its initial state is scheduled in the future, but may also be in a live state, where a moderator or an automatic match feed parser dispatches match events into its timeline or completed. References to the competition where it belongs, and participating teams are contained.

*{*

*sport: {type:String, default:'soccer'},*

*home\_team: {*

*type: String,*

*ref: 'teams'*

*},*

*away\_team: {*

*type: String,*

*ref: 'teams'*

*},*

*start: Date,*

*color: String,*

*competition: {*

*type: String,*

*ref: 'competitions'*

*},*

*name: String, // a match name generated from team names, mainly for tracing and debugging reasons*

*disabled: {type: Boolean, default: true},*

*donttouch: Boolean,*

*visiblein: [String],*

*isTimeCounting: { type: Boolean, default: false },*

*home\_score: {type:Number, default:0},*

*away\_score: {type:Number, default:0},*

*match\_date: Date,*

*time: {type:Number, default:0},*

*state: {type:Number, default:0},*

*completed: {type: Boolean, default: false},*

*stats: [mongoose.Schema.Types.Mixed],*

*guruStats: mongoose.Schema.Types.Mixed,*

*headtohead: {type:Array, default: ["W","W","D","L","L"]},*

*timeline: [segment],*

*settings: mongoose.Schema.Types.Mixed,*

*moderation: [moderationService],*

*guruStatsChecked: {type: Boolean, default: false},*

*updatedAt: Date,*

*createdAt: Date,*

*server\_time:{type:Date}*

*}*

#### ModerationService

Sub-schema of the Scheduled\_Matches schema, represents a service that handles the match feed events moderation. Usually it is linked to a third-party service that the app is integrated with, in order to receive match –related messages.

*{*

*type: String,*

*parserid: String,*

*parsername: String,*

*start: String,*

*active: Boolean,*

*scheduled: Boolean,*

*interval: Number*

*}*

#### Segment

Sub-schema of the Scheduled\_Matches schema, represents a soccer match segment such as first or second half.

*{*

*start: Date,*

*// The time in sport time that this segment starts e.g. 46' for second half*

*sport\_start\_time: Number,*

*end: Date,*

*timed: Boolean,*

*text: mongoose.Schema.Types.Mixed,*

*// time duration that the segment was on hold*

*break\_duration: Number,*

*events: [matchEvent]*

*}*

#### MatchEvent

Sub-schema of the Scheduled\_Matches schema. Represents one match event, as they are gathered inside the timeline field of the parent schema, grouped by each match period.

*{*

*match\_id: String,*

*parserids: mongoose.Schema.Types.Mixed, // one id per sender parser*

*type: String,*

*stats: mongoose.Schema.Types.Mixed,*

*playerscount: Number,*

*status: String,*

*timeline\_event: Boolean,*

*state: Number,*

*sender: String,*

*time: Number,*

*team: String,*

*description: mongoose.Schema.Types.Mixed, // one description per language*

*// extra info property to store general references*

*extrainfo: String,*

*team\_id: String,*

*complete: Boolean,*

*playerSelected: String,*

*players: [mongoose.Schema.Types.Mixed],*

*linked\_mods: [String],*

*created: { type: Date, default: Date.now }*

*}*

### Sponsor

Represents a sponsor of a leaderboard or similar ranking.

*{*

*company: { type: String },*

*name: { type: String },*

*banner: { type: String },*

*video: { type: String },*

*created: { type: Date, default: Date.now }*

*}*

### Standing

Represents a ranking of teams in a given competition, usually based on the included teams’ points.

*{*

*identity: { type: String, required: true, unique: true },*

*season: {type: Number, required: true},*

*competitionid: { type: String, ref: 'competitions' },*

*name: { type: Schema.Types.Mixed, required: true },*

*teams: [{ type: Schema.Types.Mixed }],*

*visiblein: [String],*

*parserids: { type: Array },*

*created: {type:Date, default:Date.now},*

*lastupdate: {type:Date, default:Date.now}*

*}*

### Stats-mod

Represents a certain match statistics modification, and is occurred whenever an event is coming for the match for which the given statistic depends on.

*{*

*match\_id: String,*

*stat\_for: String,*

*stat: String,*

*by: Number,*

*was: Number,*

*is: Number,*

*segment: Number,*

*linked\_event: String,*

*created: Date*

*}*

### Team

A fundamental concept along with the Competition, Player, and Scheduled\_Matches. Represents a soccer team for a specific competition.

*{*

*name: { type: Schema.Types.Mixed },*

*logo: { type: String },*

*color: { type: String },*

*stats: { type: Schema.Types.Mixed },*

*parserids: { type: Schema.Types.Mixed },*

*leagueids: { type: Schema.Types.Mixed },*

*competitionid: { type: String, ref: 'competitions' },*

*recentform: [String], // an array of String of type "W","L","D"*

*nextmatch: Schema.Types.Mixed,*

*lastmatch: Schema.Types.Mixed,*

*standing: {*

*type: Schema.Types.Mixed, default: {*

*"rank": 0,*

*"points": 0,*

*"pointsPerGame": "0",*

*"penaltyPoints": 0,*

*"wins": 0,*

*"losses": 0,*

*"ties": 0,*

*"gamesPlayed": 0,*

*"goalsFor": 0,*

*"goalsAgainst": 0*

*}*

*},*

*topscorer: { type: String, ref: 'players' },*

*topassister: { type: String, ref: 'players' },*

*players: [Schema.Types.Mixed],*

*created: { type: Date, default: Date.now },*

*updated: { type: Date }*

*}*

### User

Another fundamental schema for the app user Entity, encodes all login and profile information about the app user, personal information is anonymized (passwords are hashed).

*{*

*name: {*

*type: String*

*// ,required: true*

*},*

*username: {*

*type: String,*

*unique: true,*

*required: true*

*},*

*password: {*

*type: String,*

*required: true*

*},*

*email: {*

*type: String,*

*required: true,*

*unique: true*

*},*

*picture: String,*

*inbox: [{*

*type: String,*

*ref: 'messages'*

*}],*

*unread: {type: Number, default: 1},*

*social\_id: {*

*type: String,*

*unique: true,*

*required: false*

*},*

*// The following field is going to be used for the single frictionless sign on*

*social\_ids: {*

*type: mongoose.Schema.Types.Mixed,*

*default: {},*

*required: false*

*},*

*pushToken: { type: String, default: "NoPustTokenYet" },*

*pushSettings: {*

*type: mongoose.Schema.Types.Mixed, default: {*

*all: true,*

*new\_message: true,*

*match\_reminder: true,*

*kick\_off: true,*

*goals: true,*

*won\_cards: true,*

*final\_result: true*

*}*

*},*

*resetToken: String,*

*country: { type: String, required: false, default: "GR" },*

*msisdn: String,*

*customerType: {type: String, default:"free"},*

*subscriptionEnd: {type:Date, default: "02/28/2017"},*

*subscriptionContractId: String,*

*subscription: {type: mongoose.Schema.Types.Mixed},*

*pinCode: String,*

*birth: String,*

*gender: String,*

*admin: Boolean,*

*rankingStats: {*

*type: mongoose.Schema.Types.Mixed,*

*default: {*

*bestRank: 9999,*

*bestRankMatch: null,*

*bestScore: 0,*

*bestScoreMatch: null*

*}*

*},*

*stats: mongoose.Schema.Types.Mixed,*

*level: { type: Number, default: 0 },*

*achievements: [achievement],*

*blockedusers: [String],*

*favoriteteams: [String],*

*unlockedmatches: [String],*

*isOnline: { type: Boolean, default: false },*

*deletedAt: { type: Date },*

*deletionReason: { type: String },*

*lastLoginAt: { type: Date }*

*}*

### Achievement

Represents an app achievement definition that all users are trying to complete. Actual user achievement completion statuses are kept in the User schema.

*{*

*uniqueid: String,*

*icon: String,*

*title: mongoose.Schema.Types.Mixed,*

*text: mongoose.Schema.Types.Mixed,*

*has: Number,*

*total: Number,*

*completed: Boolean*

*}*

### UserStats

Obsolete. Not actively used.

*{*

*matchesVisited: { type: Number, default: 0 },*

*matchesPlayed: { type: Number, default: 0 },*

*cardsPlayed: { type: Number, default: 0 },*

*cardsWon: { type: Number, default: 0 },*

*prizesWon: { type: Number, default: 0 }*

*}*

### UserActivity

Represents the overall activity of a user in a given soccer match.

*{*

*user: {*

*type: String,*

*ref: 'users'*

*},*

*room: String,*

*matchesPlayed: Number,*

*cardsPlayed: Number,*

*cardsWon: Number,*

*instantCardsPlayed: Number,*

*instantCardsWon: Number,*

*presetinstantCardsPlayed: Number,*

*presetinstantCardsWon: Number,*

*overallCardsPlayed: Number,*

*overallCardsWon: Number,*

*lastActive: Date,*

*isPresent: Boolean*

*}*

### UserGamecard

Represents a game card in-play for a specific match and user. Keeps all properties for state and card management and maintenance. There are 3 types: Instant, Overall, Preset-Instant.

*{*

*userid: String,*

*gamecardDefinitionId: {*

*type: String,*

*ref: 'gamecardDefinitions'*

*},*

*optionId: String, // valid only if the definition includdes options.*

*pointsAwarded: Number,*

*pointsAwardedInitially: Number,*

*matchid: String,*

*title: Schema.Types.Mixed, // card title*

*image: Schema.Types.Mixed, // icon image*

*text: Schema.Types.Mixed,*

*primaryStatistic: String, // the primary statistic that this card is affected from, in order to be shown on the card (averages for each team)*

*guruAction: {type:String , default: "Sum"}, // Handling action of guru stats*

*// Trigger specifications*

*minute: Number,*

*segment: Number,*

*activationLatency: Number,*

*duration: Number,*

*winConditions: [condition],*

*terminationConditions: [condition],*

*pointsPerMinute: Number,*

*startPoints: Number,*

*endPoints: Number,*

*// States and state times*

*cardType: { type: String, enum: ['Instant', 'Overall', 'PresetInstant']},*

*maxUserInstances: Number, // maximum number of times a user may play this card*

*//remainingUserInstances: Number,*

*creationTime: Date,*

*activationTime: Date,*

*pauseTime: Date, // when the gamecard is suspended because the current segment ends*

*resumeTime: Date, // when the gamecard is resumed with the next segment start, after being paused*

*terminationTime: Date,*

*specials: { DoublePoints: special, DoubleTime: special },*

*isDoubleTime: { type: Boolean, default: false },*

*isDoublePoints: { type: Boolean, default: false },*

*wonTime: Date,*

*status: { type: Number, default: 0 }, // 0: pending activation, 1: active, 2: terminated (dead), 3: paused*

*// finally an array of event ids that have modified this userGamecard document since its instantiation, useful for modifying its state when an event is updated/removed*

*contributingEventIds: [String]*

*}*

#### Condition

Sub-schema of the UserGamecard, defines a condition for the final card resolution (whether the user wins or loses the played card).

*{*

*text: Schema.Types.Mixed,*

*stat: String,*

*target: Number,*

*remaining: Number,*

*teamid: String,*

*playerid: String,*

*comparativeTeamid: String,*

*comparativePlayerid: String,*

*comparisonOperator: { type: String, enum: ['gt', 'lt', 'eq']},*

*startPoints: Number,*

*endPoints: Number,*

*pointsPerMinute: Number,*

*conditionNegation: { type: Boolean, default: false }*

*}*

#### Special

Sub-schema of the UserGamecard that represents an extension of a gamecard with special abilities like time extension or playing for double points.

*{*

*creationTime: Date,*

*activationTime: Date,*

*activationLatency: Number,*

*status: { type: Number, default: 0 } // 0: not enabled, 1: pending activation 2: activated*

*}*

### Star

*Represents a view of the app top users (the stars), a special leaderboard.*

*{*

*users: [userSchema]*

*}*

#### UserSchema

Sub-schema of the Star, for holding top ranking details for one –star- *user*

*{*

*rank: { type: Number },*

*user: { type: String, ref: 'users', required: true },*

*titles: [titleSchema]*

*}*

#### TitleSchema

Sub-schema of UserSchema, represents one top ranking details for a specific user

*{*

*iconUrl: { type: String },*

*date: { type: String },*

*text: { type: Schema.Types.Mixed }*

*}*

### Purchase

Represents the paid subscription status for a user

*{*

*status: { type: String }, // "Inited" / "Pending" / "Completed"*

*user: { type: String },*

*type: { type: String }, // "Subscription" / "Match"*

*info: { type: String }, // "Weekly Subscription" / MatchID*

*provider: { type: String },*

*method: { type: String },*

*receiptid: { type: String },*

*providerMessage: {type: String},*

*created: {type:Date, default: Date.now()}*

*}*

### Taunt

*Represents a definition for predefined provocatory messages that a user may send to a friend*

*{*

*type: { type: String },*

*term: { type: String},*

*imgurl: { type: String },*

*sprite: { type: String },*

*text: { type: mongoose.Schema.Types.Mixed },*

*animation: { type: String }*

*}*

### UserTaunt

Represents instances of Taunt actually sent by users, with a reference to the taunt definition

*{*

*sender: {*

*type: String,*

*ref: 'users'*

*},*

*recipient: {*

*type: String,*

*ref: 'users'*

*},*

*taunt: {*

*type: mongoose.Schema.Types.Mixed*

*},*

*created: { type: Date, default: Date.now }*

*}*

## Database Footprint and Size projection

As of 25/5/2018, the database size is 869572992 Bytes, containing 111929 documents with an average document size of 7768 Bytes. 146 matches have been played from the start of 2018, which contribute to half a season for about 7 national leagues.

The 1-day footprint corresponding to 1 match played in the app is approximately 6 Mbytes. In order to acquire a size required to run x matches in a season for various competitions, this has to be multiplied by x.