|  |  |
| --- | --- |
| GhostRunner Design Document  Design and functional specification |  |

**(covers parts of the system only)**

**Version:** 1.0

**Last Updated:**  August 31, 2017

**Copyright**: GhostRunner.

**Author****:** Alamgir Farouk

**Table of Contents**

1 Overview 5

2 Enitities 5

2.1 Owners 5

2.2 Players 5

2.3 PlayerSeason 5

2.4 Teams 5

2.5 TeamPlayers 6

2.6 TeamLineUp 6

2.7 TeamLineUpPlayers 6

2.8 GameUserLineupPlayers 6

2.9 Game 6

2.10 GameEvents 7

2.11 Viewers 8

3 Database Schema 8

4 Game States and transition 9

4.1 Inviter states 9

5 Game Login 11

6 Game Broker signaling 12

7 Player Importing 14

7.1 Differences between types 14

7.2 Import mechanism 14

8 Invitation cycle 15

8.1 Selecting team (Inviter) 15

8.2 Selecting a pitcher 16

8.3 Selecting the lineup 18

8.4 Game start 19

8.5 In game properties 19

9 (NOT IMPLEMENTED) Team creation UI 21

10 UI for team in offense 22

10.1 Action buttons 22

10.1.1 SWING AWAY (1) 22

10.1.2 BUNT ATTEMPT (2) 22

10.1.3 STEAL A BASE (3) 23

10.1.4 HIT-n-RUN ATTEMPT (4) 23

10.1.5 CLUTCH BAT ATTEMPT (5) 23

10.2 Lineup change buttons 24

10.2.1 PINCH HIT(TER) (6) 24

10.2.2 PINCH RUN(NER) (7) 25

11 UI design for team in defense 26

11.1 Action buttons 26

11.1.1 PITCH TO BATTER (1) 26

11.1.2 INTERNATIONAL WALK (2) 26

11.2 Lineup Change Buttons 27

11.2.1 DEFENSIVE SUBSTITUTION (3) 27

11.2.2 RELIEF PITCHER (4) 28

11.3 Player Cards 29

11.4 Data (TBD) 30

12 Question/answers (excerpts from emails) 31

13 Rough sketches/notes 38

14 Open Issues 39

15 API Samples 40

**Legal Page**

**Confidentiality Agreement**

The undersigned reader acknowledges that the information provided by GhostRunner in this document is confidential; therefore, the reader agrees not to disclose it without the express written permission of GhostRunner Inc.

It is acknowledged by the reader that information to be furnished in this document is in all respects confidential in nature, other than information which is in the public domain through other means and that any disclosure or use of same by the reader may cause serious harm or damage to GhostRunner Inc.

Upon request, this document is to be immediately returned to GhostRunner.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Signature

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Name (typed or printed)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Date

# Overview

This document provides the schema and API details for the Baseball Game engine.

The schema is based on Baseball rules, GhostRunner game and remote interaction requirements. We describe each table separately, and show their combined relationship at the end.

UID (User identifier) represents a user (logged in). Complete description of User entity is omitted in this document.

NOTE: Actual table names may be different based on schema requirements.

# Enitities

These are the high level entites we will use.

## Owners

An Owner is a user in the game, identified by UID and other meta data.

## Players

This represents a player in the ‘pool’ of players. This table captures personal (non-game) related information about a player.

## PlayerSeason

A particular player’s performance during a season is caputured in this table. It is also known as the ‘player card’.

## Teams

One owner can have multiple teams. So, teamId is incrememented. Teams may become dormant or retired. We can decide how many active teams are possible.

## TeamPlayers

This represents a player in the ‘pool’ of players. This table captures all the information about historical players, possibly using auxilliary tables for historic data.

## TeamLineUp

A lineup is created by an owner using his team players. A lineup can have players only from a team. So the Team comes first .

## TeamLineUpPlayers

(not used)

## GameUserLineupPlayers

This table hold the lineup for any particular game and each player related information.

## Game

A game represents a new game being played between two TeamLineUps. At any time, the game entity records the state of the game, who is playing, who is standing where, etc. It should record all information such that a game can be reconstructed and play can continue. Current scores are also cached although they could, in theory, be computed from the playevents entries.

Games

* Home(id)
* Away(id)
* id(id)
* current inning
* player positions
* [game state]
* [ui state]
* [cached score]
* …
* Inninings
* Who is at bat
* Who is pitching
* Out status (0,1 or 2)
* Base status(0 through 7) (who where)

Figure 1 Games table

## GameEvents

Each play event is a chronological record of the events that combine together to form the game. The events maintain a sequence, which is important for multiplayer distributed games.

GameEvents

* gameId(id)
* sequeneId(id)
* inningId
* eventType
* resultType
* …

GameSubEvents

* gameId(id)
* sequeneId(id)
* subSequenceId(id)
* eventType
* resultType
* …

Figure 2 BaseballGamePlays

## Viewers

A viewer is a user who is allowed to observe an ongoing game. A viewer-game relation table will be created (omitted as too straightforward)

# Database Schema

The following diagram shows the overall schema of the game engine. Most attributes are omitted, only the relational model is shown.



Figure 3 ER diagram for game and teams

# Game States and transition

The two teams as well as the game itself go through unambiguous states which are persisted at any time, all transitions between them being atomic.

We omit the game states for simplicity. In general, the game state will capture the ‘turn’ information (Owner1, Owner2 or System), but within the turn, the actual state of team will be maintained in the TeamLineup table, and individual player states will be maintained for the appropriate player in TeamLineUpPlayer table.

## Inviter states

We only consider the state transitions for the Inviter (the one who initiates the game) and the Invitee.

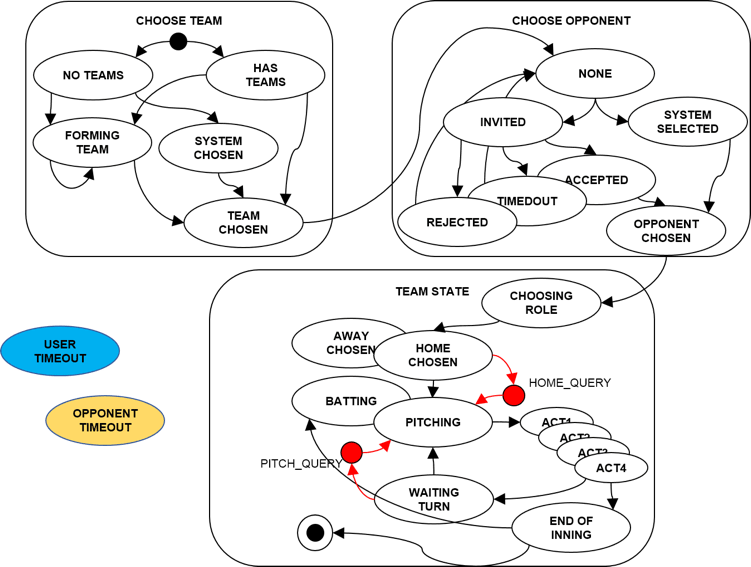


Figure 4 Inviter (the initiator of the game) state transitions.

Invitee states

The person being invited goes through a slightly different set of states at the beginning, but we repeat the full life-cycle for completeness.

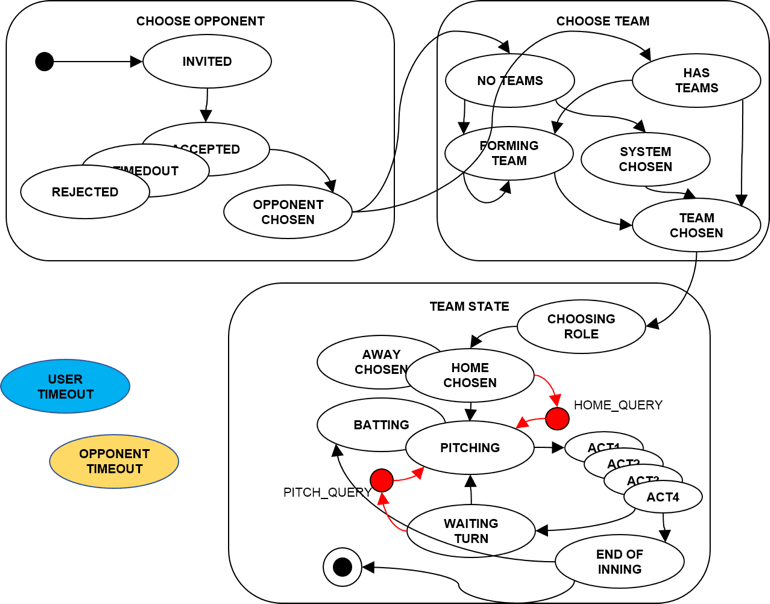


Figure 5 Invitee (person who was invited) state transitions.

Game flow

The game flow itself is modeled as a flow diagram, to better illustrate the actors (owner1, owner2 or system)

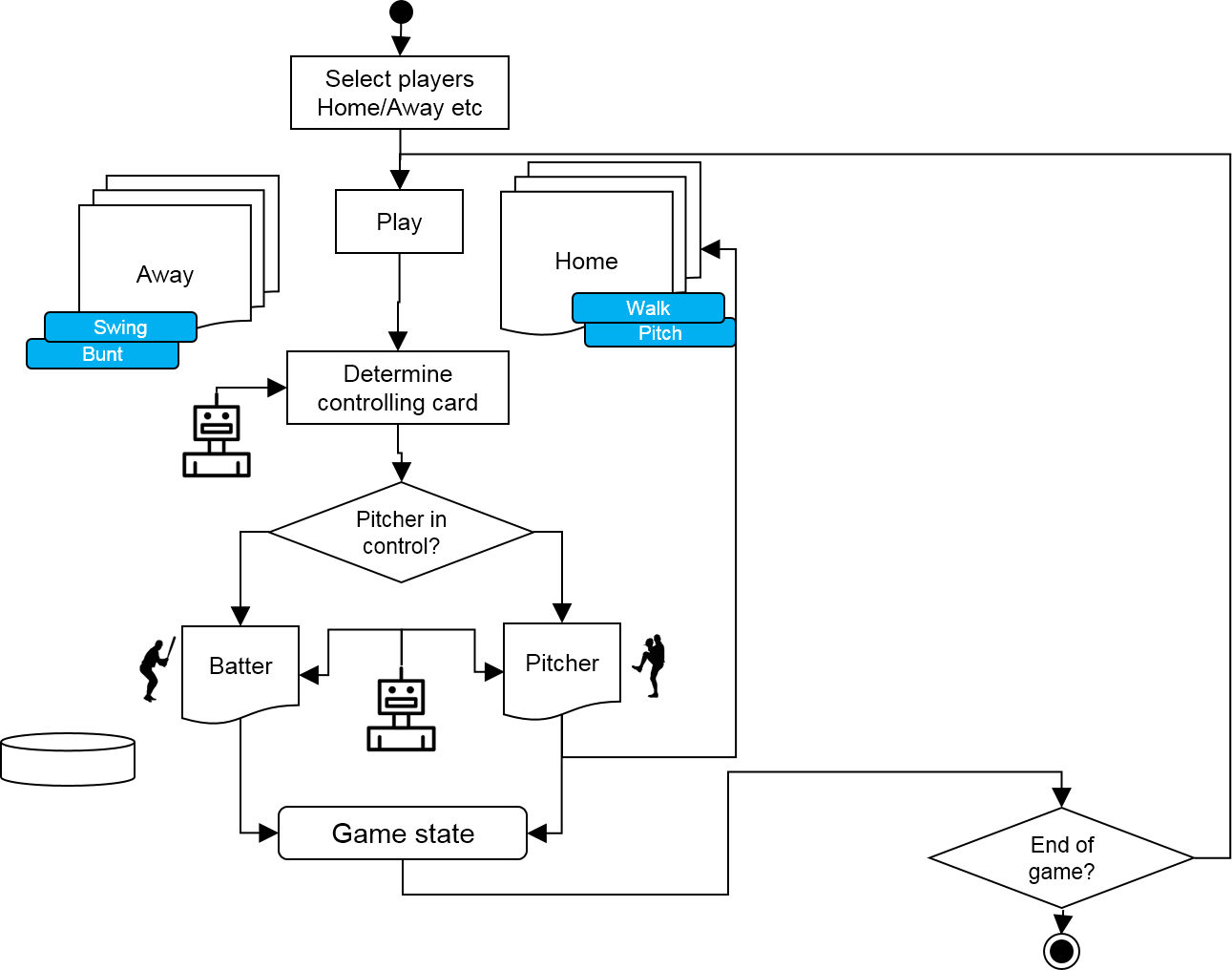


Figure 6 Game flow diagram.

# Game Login

The login will use the standard authentication from our authentication package, with the following requirements

The login component (javascript object?) will be a separate component but it can be built as part of the package.

The login will set a client side cookie (as usual) with the UID as well as fire an event on login and logout and allow attaching listeners to it. Other pages may also want to listen to this event, so it may be a good idea not to have any dependencies other than jquery etc.

The login will maintain all user details (username, UID) by retrieving them from the server. If retrieval fails, state will be logged out, cookie removed (same as now).

The login will parse the URL and use the embedded UID and retrieve the username etc., when the URL contains a UID. Invalid UIDs (when server throws an error) will be ignored silently.

Valid UIDs will result in logged in status as usual.

# Game Broker signaling

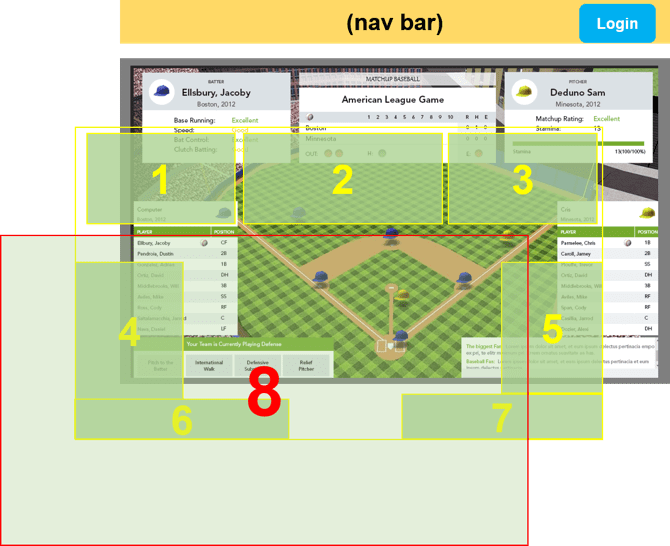


Figure 7 Main UI components

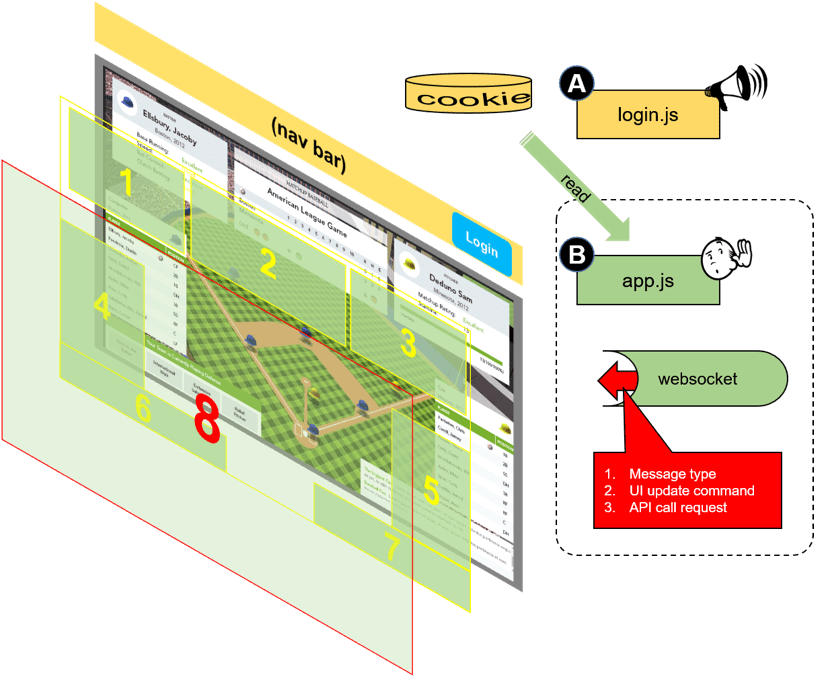


Figure 8 Signaling via websocket will control the masks and UI state. Default state will be full mask (8)

# Player Importing

Two types of players are identified, Pitchers and Fielders. They do not have the same set of data, which means some database will be null/undefined for respective types.

## Differences between types

The main differences between fielders and pitchers will in the way they are rated, and used for lineup creation.

## Import mechanism

The excel worksheets will be read by an importing software which will convert the excel data into json (JSON) files, which, in turn will be used as arguments for the player creation APIs.

# Invitation cycle

The invitation cycle consists of selecting an existing member or entering an email whence an adhoc-member is created and then selected for invitation. Additionally, a ‘starter’ Pitcher must be selected by the inviter. The figure below shows the combined member-selection, and email entering screen.

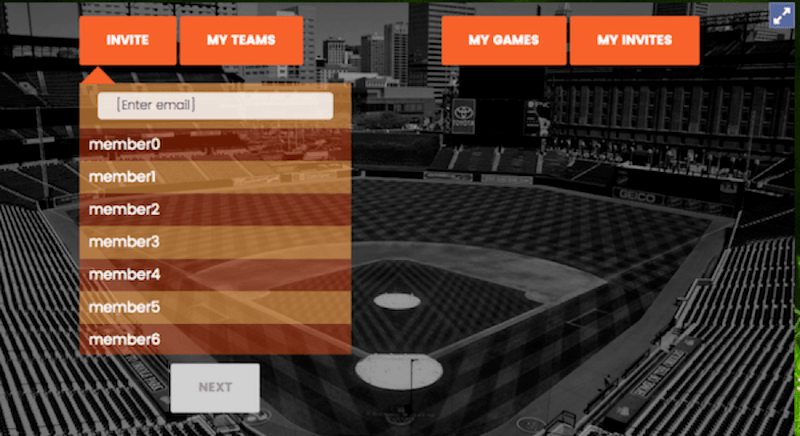


Figure 9 Selecting member or entering email

## Selecting team (Inviter)

The inviter, upon selecting user as described above, will be presented with a UI for selecting a team.

1. He can choose from exiting (free) teams
2. or use a team he has created from a pool of available players.

Option one will be implemented first.

The figure below shows the view for selecting the team. NOTE: Only free teams are shown in the example. Teams that the user owns will have ‘edit’ and ‘delete’ buttons next to them.

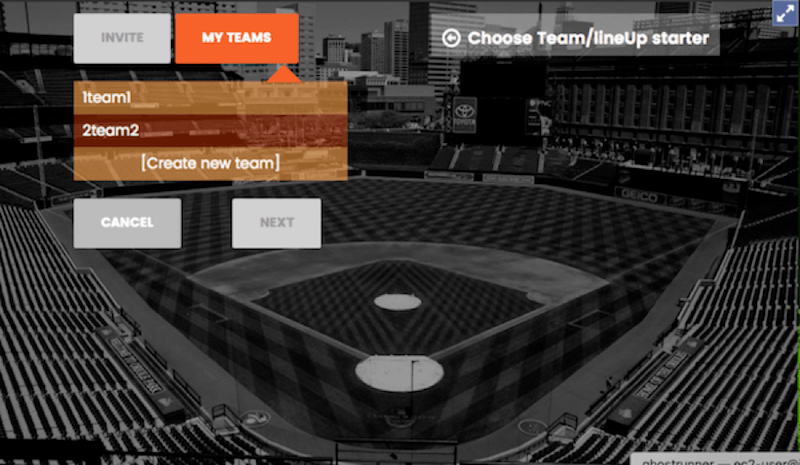


Figure 10 Team selection screen.

Note that the current UI also provides an option for creation of a team. This may be hidden or removed during deployment as it may be a paid function.

## Selecting a pitcher

Once a team has been chosen, we will show only the pitchers from that team and allow only one selection (radio button).

**NOTE**: A pitcher is identified as a player that was imported from the ‘Pitcher’ worksheets. No other variable is considered in determining who should be included in the pitcher list.

**NOTE**: the user must enter a lineup name first. This lineup name is for display purposes only, it has no significance otherwise.

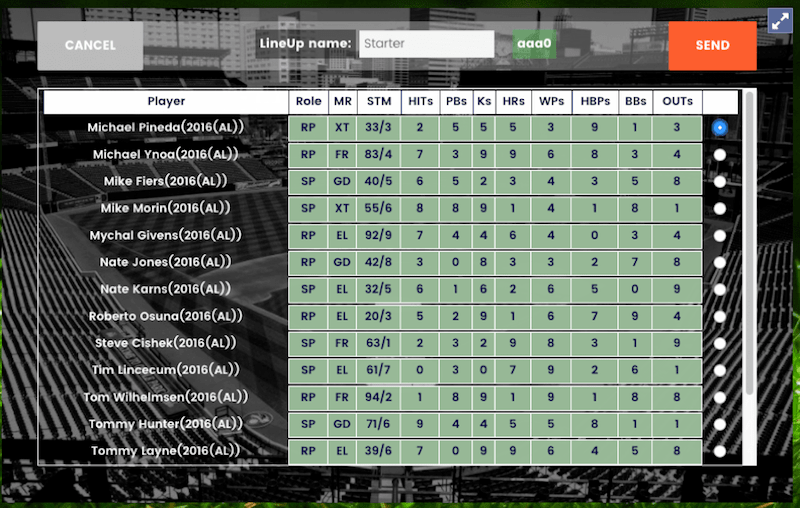


Figure 11 Selecting starter pitcher from team.

Upon successful submission of the invitation, the ‘inviter’ will have to go through the same steps.

Once the inviter has also selected his starter, the invitee will have to select his lineup (from the same team).

## Selecting the lineup

The user is presented with a list of ALL players in his team (chosen earlier).

1. Only fielders will be shown.
2. Fielders will be shown choice of field positions based on P1, P2 etc values.
3. Fielders will be shown choice of Batting order (BO)
4. All players will be shown selection checkbox.

The figure below shows the UI for lineup selection.

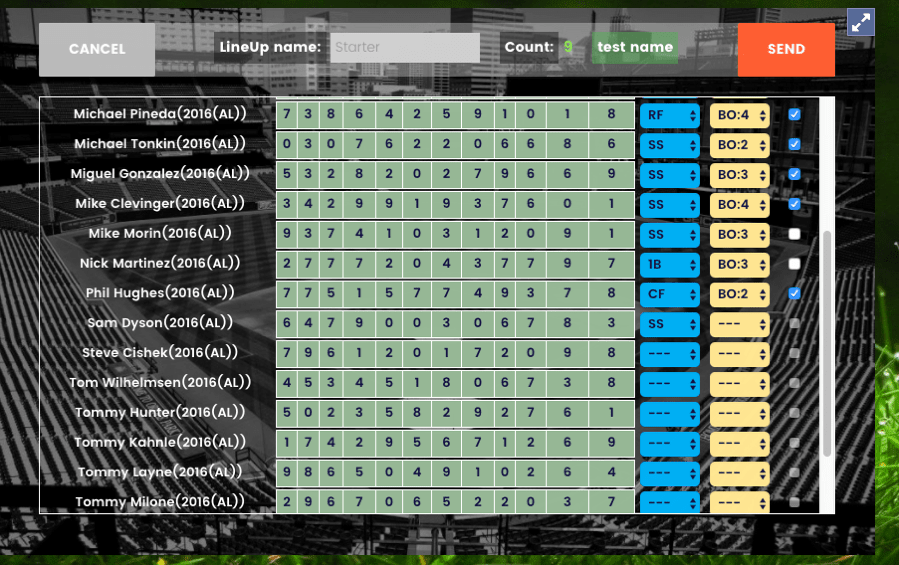


Figure 12 Lineup selection UI. (see modifications proposed below)

The lineup selection UI will show ALL available players from a team. The team name and lineup name will be displayed on top for reference.

1. ONLY fielders from the team will be shown in this screen.
2. The checkboxes on the far right.
3. A drop down will be provided for each player for the properties which must be configured during line up selection. .
   1. For FIELDERs it will be populated by P1, P2 etc. values from the worksheet. Every entry the FIELDER has in the worksheet will allow a new entry in the dropdown. This results in the ‘POSITION’ value for this FIELDER.
4. The total number of players will be 9 (need to verify this)

## Game start

Once both players have selected lineups, the inviter is notified that the invitee has accepted his invitation and he can start the game. Once a Game is started it is switched to RUNNING state.

NOTE: A user can have only one RUNNING game at a time. But a user can have unlimited number of games in any other state, such as INVITING, PAUSED, etc.

The invitation cycle automatically pauses any RUNNING game, so that when this game is RUNNING other games will be paused. Other users will be notified without option.

## In game properties

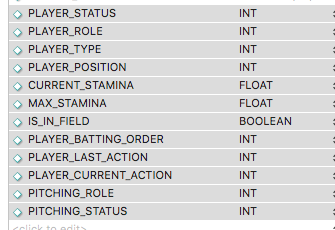


Figure 13 In-game player variables

PLAYER\_STATUS: General purpose variable we can use for maintaining the status of a player (we define possible values as needed)

PLAYER\_ROLE: For pitchers this will be the typical pitch roles as recognized in Baseball terminology. For fielders they will be defined by us, and may simply be one constant (‘fielding’)

PLAYER\_TYPE: PITCHER or FIELDER to enable overriding their original roles. Initially they will match original player roles.

PLAYER\_POSITION: For fielders, these will be standard Baseball terminology (e.g. SS etc). For Offense players, they will be positions defined by us to indicate base, home plate etc. Players not in game or not playing will have special positions defined by us.

CURRENT\_STAMINA: used for pitchers while pitching, and ignored (negative or null) for all other players.

IS\_IN\_FIELD : a Boolean to indicate if the player is physically in the field or not.

PLAYER\_BATTING\_ORDER: Used for capturing the current batting order.

PLAYER\_LAST\_ACTION: We will maintain the last action taken by a player mostly for calculations or game display.

PLAYER\_CURRENT\_ACTION: This is the action the player is taking at the moment. May be null or not needed depending on whether we make all moves ‘atomic’ (so moves must end) or not.

PITCHING\_ROLE: We use standard baseball terminology to capture current pitching role for pitchers. For Fielders, we will use derived placeholders.

PITCHING\_STATUS: We may not need this, based on whether we consider pitches as atomic or not.

**Please add any other in-game (RUNNING) state variable we may need for EVERY player (fielders, pitchers, batters).**

# (NOT IMPLEMENTED) Team creation UI

A user can create a team from available players. While this may be a paid function later, for now we are implementing it as a free/available function to get it working.

The UI is shown below.

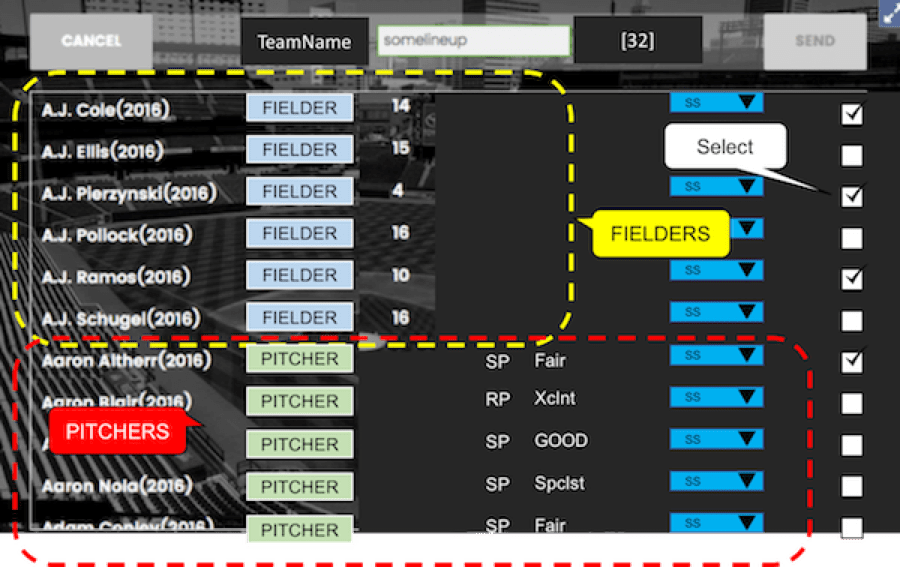


Figure 14 We show costs, which will be added for total of 75. Similar restrictions for pitchers.

Once the game is started (RUNNING) the UI switches to the ‘field’ mode as shown below.

# UI for team in offense

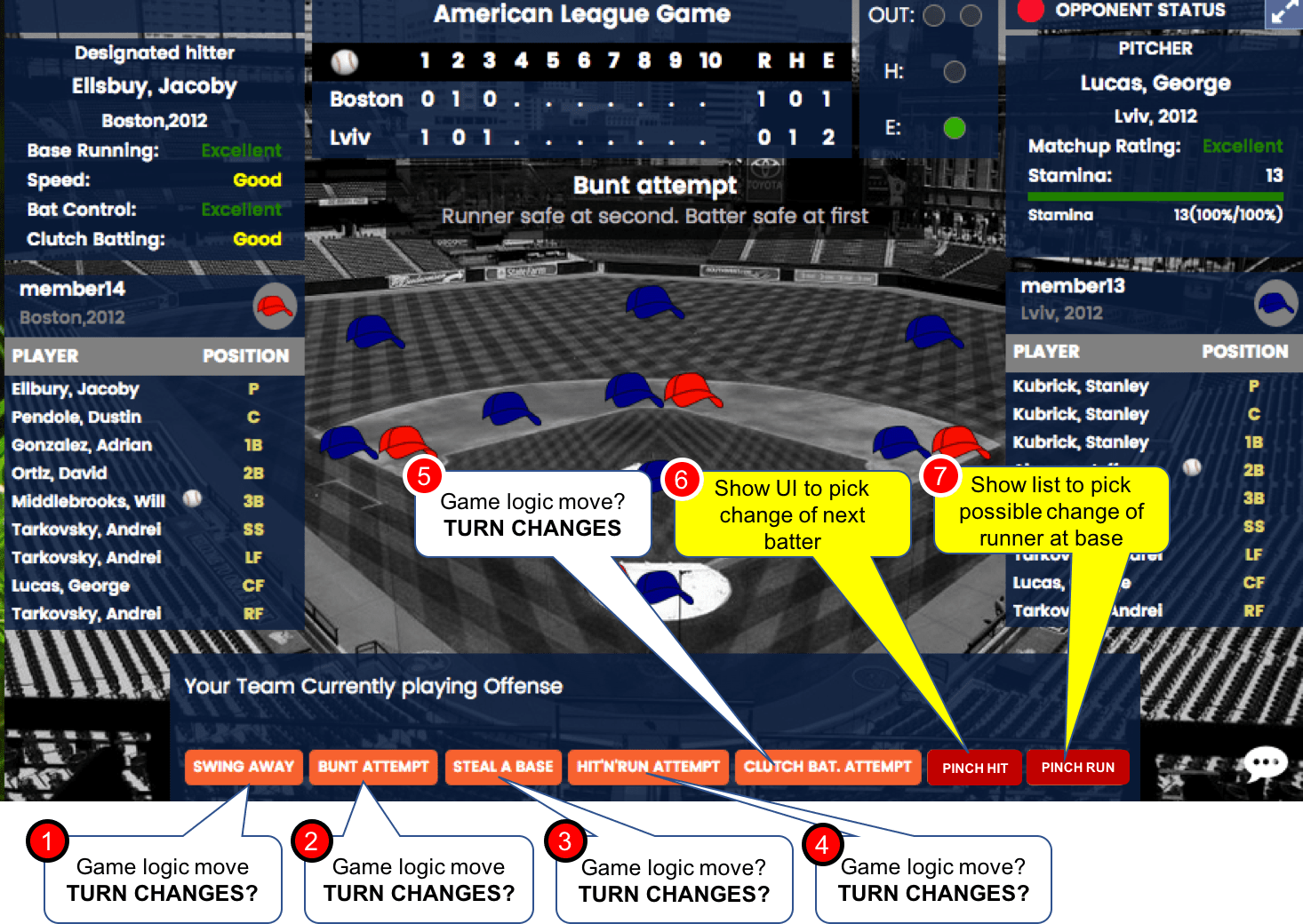


Figure 15 In-game OFFENSE UI during a RUNNING game. (NOTE FICTIOUS DATA USED)

## Action buttons

### SWING AWAY (1)

* 1. API call,
  2. no UI,
  3. turn changes,
  4. possible secondary move,
  5. button text never changes.

### BUNT ATTEMPT (2)

1. API call,
2. no UI,
3. turn changes,
4. possible secondary move,
5. button text never changes.

### STEAL A BASE (3)

1. API call,
2. no UI,
3. turn changes,
4. possible secondary move,
5. button text never changes.

### HIT-n-RUN ATTEMPT (4)

1. API call,
2. no UI,
3. turn changes,
4. possible secondary move,
5. button text never changes.

### CLUTCH BAT ATTEMPT (5)

1. API call,
2. no UI,
3. turn changes,
4. possible secondary move,
5. button text never changes.

## Lineup change buttons

### PINCH HIT(TER) (6)

1. ***Purpose*** - Replace next batter with new fielder from team.
2. Pull up list of available players, to replace ‘next’ batter, show name of batter being replaced.   
   **NOTE: current batter will not be replaced**
3. API call on save to make lineup change.
4. Signal to both users to refresh screen.
5. The position of this player will be same as player he replaced during fielding later.
6. **Turn DOES NOT change**
7. no secondary move,
8. button text never changes.

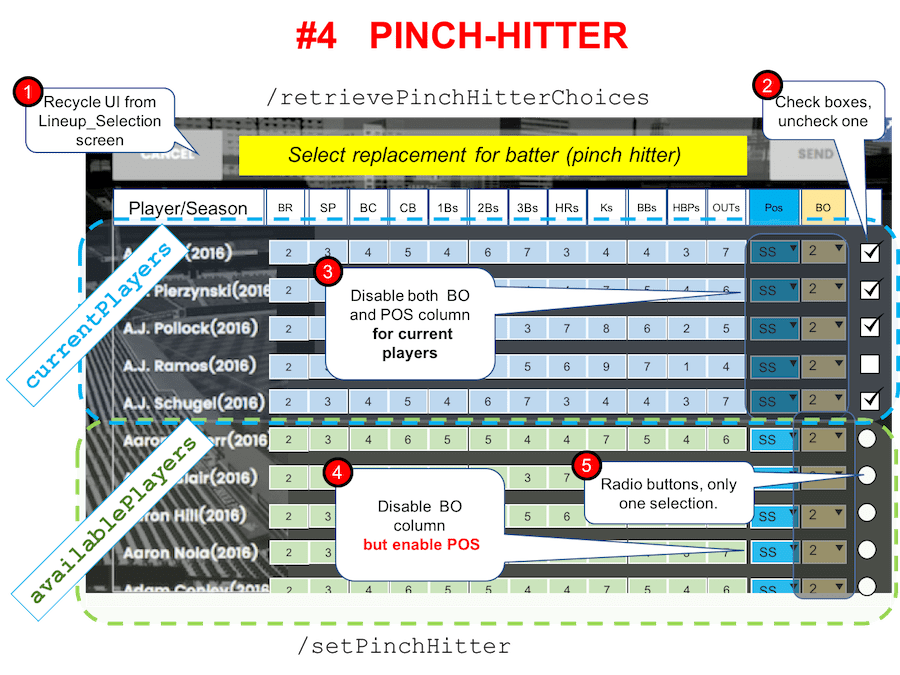


Figure 16 Pinch Hitter selection UI (for next batter)

### PINCH RUN(NER) (7)

1. ***Purpose*** - Replace some runner at base with new fielder from team
2. Pull up list of current runners at base, pull up list of available players, to replace ‘selected’ runner.
3. API call on save to make lineup change.
4. Signal to both users to refresh screen.
5. **Turn DOES NOT change**
6. no secondary move,
7. button text never changes.

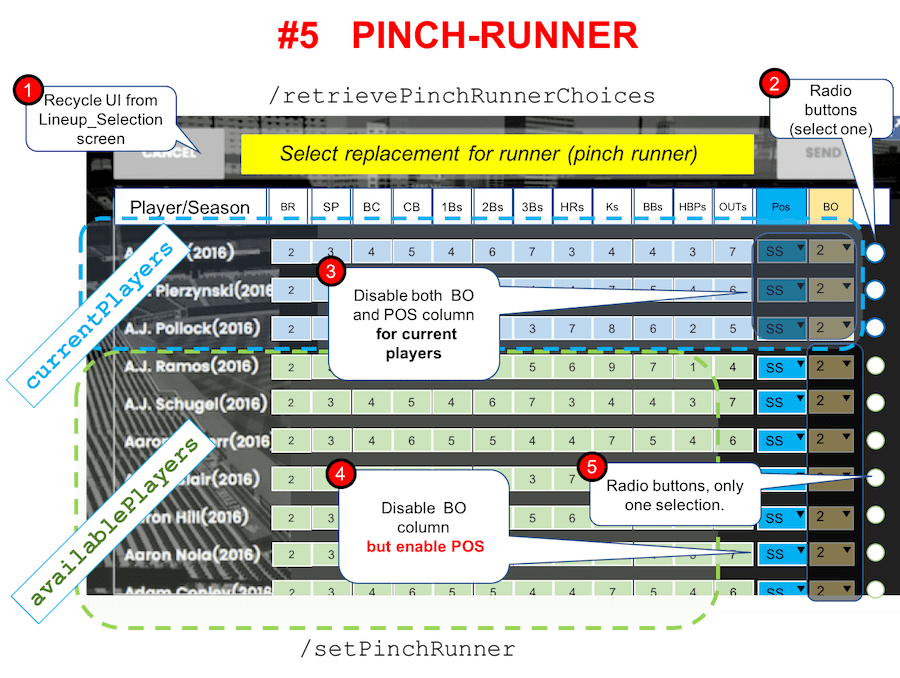


Figure 17 Pinch runner selection UI

# UI design for team in defense

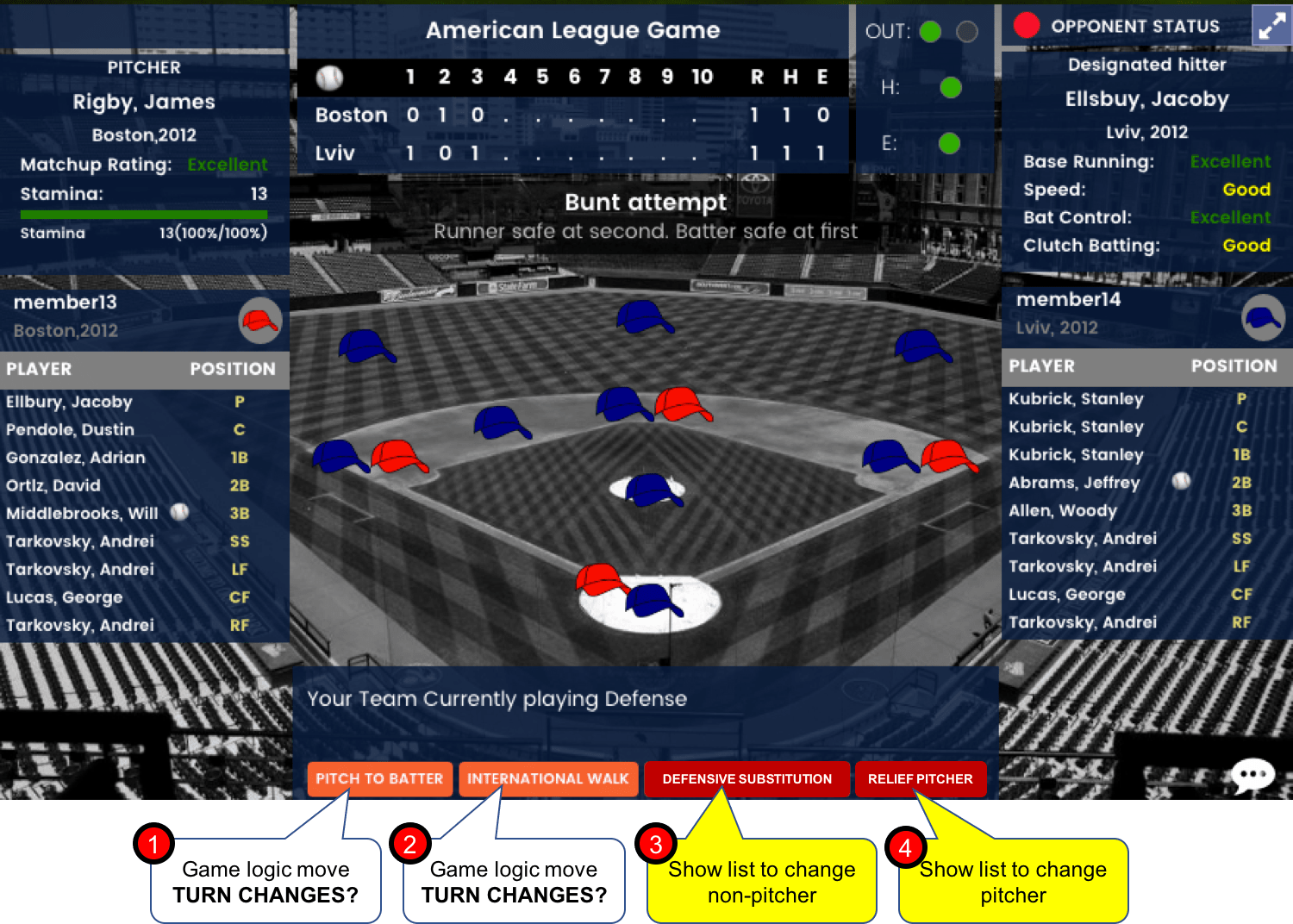


Figure 18 UI for team playing defense.

## Action buttons

### PITCH TO BATTER (1)

* 1. API call,
  2. no UI,
  3. turn changes,
  4. no secondary move,
  5. button text never changes.

### INTENTIONAL WALK (2)

* 1. API call,
  2. no UI,
  3. turn changes,
  4. no secondary move,
  5. button text never changes.

## Lineup Change Buttons

### DEFENSIVE SUBSTITUTION (3)

* 1. ***Purpose*** - Replace some current fielder with new fielder from team.
  2. Pull up list of current fielders (all in field except pitcher), pull up list of available players, select (radio) of player being replaced, select (radio) of player who is replacing.
  3. API call on save to make lineup change.
  4. Signal to both users to refresh screen.
  5. **Turn DOES NOT change**
  6. no secondary move,
  7. button text never changes.

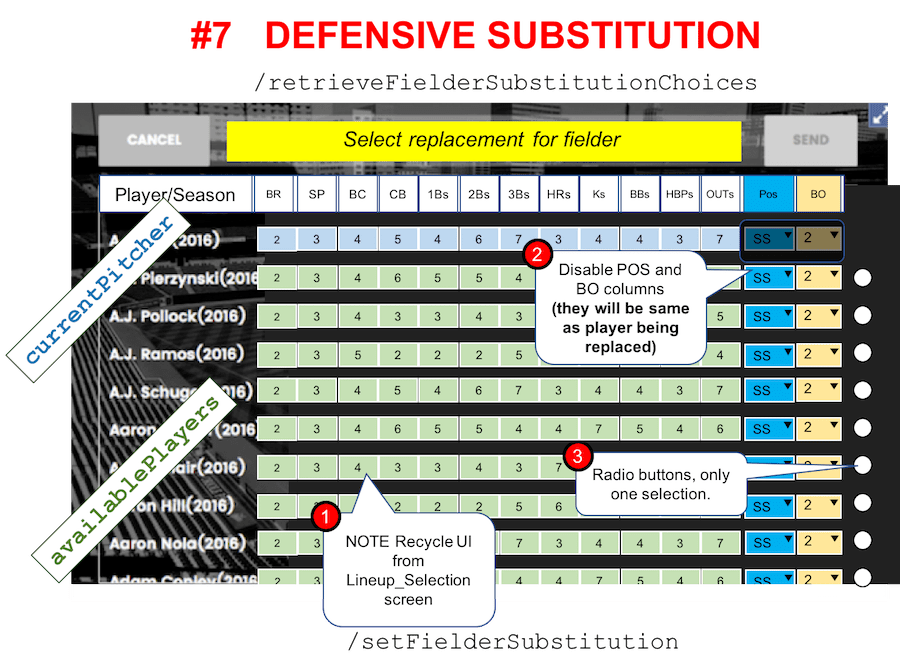


Figure 19 Defensive Substitution UI

### RELIEF PITCHER (4)

* 1. ***Purpose*** - Replace some current fielder with new fielder from team.
  2. Pull up list of current fielders (all in field except pitcher), pull up list of available players, select (radio) of player being replaced, select (radio) of player who is replacing.
  3. API call on save to make lineup change.
  4. Signal to both users to refresh screen.
  5. **Turn DOES NOT change**
  6. no secondary move,
  7. button text never changes.

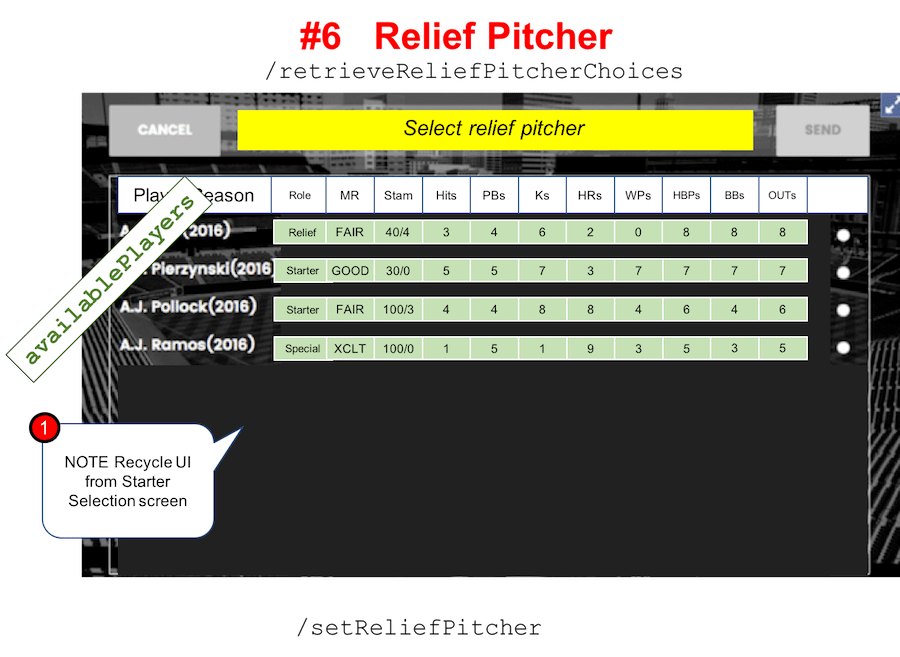


Figure 20 Relief Pitcher selection UI

## Player Cards

Right clicking on any player on the field will display a player card. The design will be derived from the following sketches.

We identify the following:

The player card depends on the field position of a player (not the type or role etc).

Following four positions are possible

1. Fielder (DEFENSE)
2. Runner (OFFENSE)
3. Batter (OFFENSE)
4. Pitcher (DEFENSE)

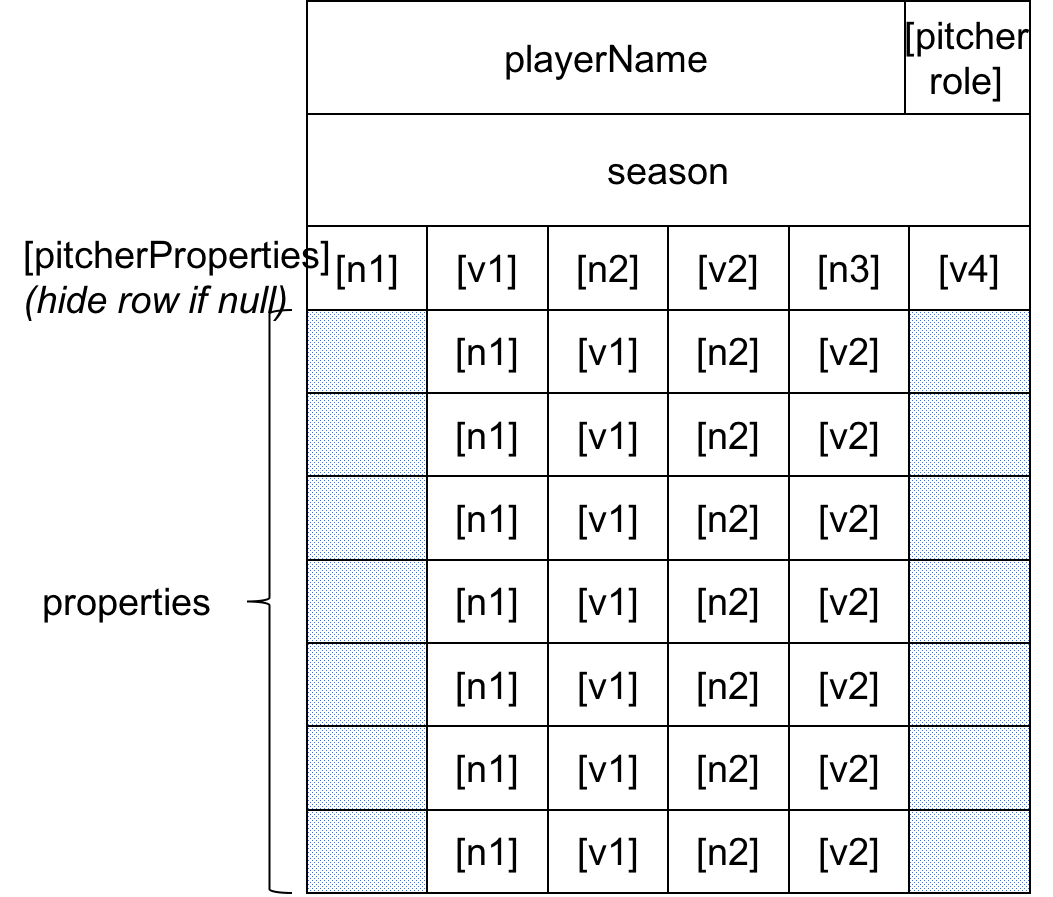


Figure 21 Player card structure

## Data (TBD)

# Question/answers (excerpts from emails)

**AF: Does the term Role only apply to pitchers? So SP, etc.   
the user change the role of a pitcher (say from SP to RP etc)? (yes/no?)**

JG: The term "role" really only applies to pitchers.  For batters / fielders, it is typically called, "position."  For example, "What position does Joe Morgan play?  Second Base."  and "Have you decided yet on Riviera's role?  Yes, he'll be exclusively used in relief."

**a) For starters, we will declare the 'type' of a player as either fielder or pitcher. So ALL players are either of type PITCHER or FIELDER. (they can cross over).**

**b) For 'PITCHER's we will have SP, RP etc as 'roles'. So, during selection (lineup), the role can be decided as long as he is a player of type pitcher. For fielders the role will be 'UNRESTRICTED' (meaning there are no restrictions). (YES/NO)?**

***AF:***

***c) Role is decided during lineup creation, but we show the role from the database (yes/no)?***

***d) Role is shown ONLY for pitchers. (for fielders we will have the role of 'UNRESTRICTED' but we will not show it).***

***e) For 'FIELDERS' we will have field positions while playing (so these can change). These are dynamic and changeable. Positions are required for every player in the field, so we need a position for the pitcher as well otherwise we describe the field. So we will assign a fictitious position for the pitcher (pitcher).***

***f) When is a batter decided during lineup creation? How is he decided? (We need an order value, how do we offer choices). Currently, when the user is creating his lineup, we have no design for how these (batting order) are decided.***

***(AF: These questions were not answered)***

**AF: So, batters have no 'role' value, nor do 'fielders' unless they are all in the role of 'fielder'?**

JG: Now, as to the second question... this can be summarized as, "Starters can be used in relief, but relievers can never start."  So, effectively, yes... an owner can change the role of a pitcher **- in the course of a game.**  Essentially, giving a pitcher the "S" designation is like giving them a privilege that those with "R" do not have.  So, we don't really get hung up on whether or not an owner will "change" a pitcher from SP to RP (or from S2 to RP)... we just check to see whether or not a pitcher has the "S."    
**(AF: in game rule. Did not answer the question if being a ‘fielder’ is a role by itself.)**

There are some UI implications to this... When deciding on your starter, the pitcher who started the previous game and the pitcher who started the one before that are both "red."  That is, they cannot be used in any way (start or relief).  On the other hand, the pitcher who started the third game back is "yellow," which means that he can relieve, but not start.  I bring this up in recognition of the fact that Vasyl is gone soon. :-)   
***(AF: Starter selection rule: both yellow and red are ineligible, in game rule: yellow can)***

***AF: We have received no feedback on UI for lineup creation so far. Please see the lineup creation UIs we sent you earlier. Based on our current UI, here is what I can propose:***

***During startup selection, we only show players of type PITCHER from the team.***

***IF a user repeats using a trial team, we will have no memory of previous games.***

***When a user uses a trial team, we clone the team for him (shows up as 'cloned'). Only if he uses his cloned team can we maintain history.***

***After each game we will update the status of the pitchers in his 'cloned' team only and cannot update 'trial teams' as they are visible to others.***

***For the next game, we will use the properties from this team (so history is relevant) only if the user uses one of his 'cloned' teams. We can stop showing the Trial Teams once he has a cloned team so this decision can be transparent to the user, but we need some clarification on this.***

**AF: When creating lineup, can the user change the 'position' of the fielders? (Yes/no?) Is there a default position of any fielder? (yes/no, if yes, which column defines it)**

JG: Absolutely, the owner is free at all times to place a batter / fielder into any position that they are eligible to play (as mentioned before, these - up to five positions, but usually only one or two - are listed in the spreadsheet, right after the name?).    
***(AF: Did not answer the question about whether the position needs to be assigned and changed during line-up creation. The answer appears to be an in-game action. We have not prepared for in-game line up editing.)***

For example, I sent you a picture of the player card for Kris Bryant as a fielder.  Please note that he has the ability to play third base, left field, or second base.  Thus, the owner - when constructing his lineup - can use Kris Bryant at any of those three positions.  In addition, at any point during a game, the owner can move Kris Bryant from one of those positions to another of those three... and then move him back again later if he wants.  It's rare, but it DOES happen.  But recognize... the owner can ONLY play a batter / fielder at a position listed on his card.  Incidentally, the five positions are listed in order of "most frequently used at" - P1 - to "least frequently used at" - Px (where x = the highest number for that player).  So, P1 is generally considered the "default" position of a player.  Even more incidentally, if the player has no P1 value, the he is only eligible as a Pinch Hitter or a Designated Hitter... that is, he cannot play in the field.  Oops... one more item: Any batter / fielder can pinch hit, pinch run, or be a designated hitter.  Those "positions" are not specified.

***AF: a) See attached FielderPositions from excel sheet. If a fielder has only one entry, he gives only one choice.***

***b) The 'pinch hit' etc, are not bonafide positions, right? Once again, we have the problem of 'rules' vs actual positions. A designated hitter, is a 'batter' as far as field position is concerned. The fact that he is a designated hitter has nothing to do with his current position on the field, but it has to do with the fact that he is the 10th player so it has to do with lineups. We have to clarify these, especially differentiate between actual 'physical positional' attributes as opposed to attributes that dictate whether he can take up a physical position.***

JG***:*** 1a - yes.

1b - yes.

1c - The role is clear during lineup creation. At that point, you only want to select the starting pitcher - and he will either be SP or S2 (from the database). Most DEFINTELY we only show the role from the database.

1d - YES! I like your idea of "just define something but don't show it."

1e - yes. Create a fictitious position for pitcher called "pitcher." While we do not (today) care about a pitcher's DEF, maybe we will in the future.

1f - this is a VERY good question. I had sent some screen shots in the past and described some examples, but obviously these all went into the "wait until later" file. No sweat... I will review what you have for Vasyl and reply VERY soon.

my 1b ("the second question") - You are absolutely accurate when you point out that, if a user is playing a trial team, he will not have any records kept. This is the key element of this item. The fact had eluded me because my focus has always been on the paying customer (more specifically on the maintenance paying customer). Yes, no records will be kept, so that player can repeat his same pitcher over and over. Whatever... he still has to make out a lineup for the fielders. On the other hand, this "cloning" exercise is not readily understandable to me tonight. Hmmm... Frankly, I don't really care if a trail user has his history tracked or not. Let them use the same pitcher over if they want to... our choice of pitchers for the team will make this element moot.

2a - correct... and correct. These are the positions, P1 being the most common (default), and DO use them to allow or define options during lineup creation and during game play. Yes.

2b - right. The "pinch hitter" and "pinch runner" are not bonafide "defined" positions. And, although "designated hitter" is, they can be treated the same (that is, no position) from a program perspective (to your "rules vs. programming" point). I am probably not making sense here, but the gist is that you are correct... we have to define a 10th position, even though it is not "actually" a position.

3 (ahh, the definition of "cards."). - Sorry. I was under the impression that I had clarified what I meant by a "card" for this game. Because we began this game by playing with index cards and dice (and hand-calculating the results!), the legacy is to call the definition of a given player his "card." In reality, nowadays, a player's "card" is simply his row in the spreadsheet that has all of the calculated and defined parameters that define him for that given season. Now, a quick glance at the pitchers' spreadsheet vs. the fielders' points out that the values are different. Pitchers have certain parameters that fielders don't have... and vice-versa. So, to re-phrase, "if a given player does not have a row defined in the pitchers spreadsheet for a given season, then he cannot pitch for that season." And, sorry about the extraneous comments about what a pitcher can and cannot do... I was mostly just trying to be holistic.

AF: A pitcher cannot be added to the lineup as a fielder (yes/no?) (and what about the reverse)

JG: If a player does not have a "pitcher card" defined for a given season, he cannot pitch for that season.  Similarly, we do not define a DEFensive Rating for a pitcher.  Maybe later - not now.  Also, a pitcher does not have a "batter card."  Yes, in the NL, a pitcher does bat... but when pitcher bats, the opposing pitcher automatically has control and the result of the matchup is always taken from the defensive pitcher's card.  Finally, except for bunt attempts, a pitcher cannot utilize any of the strategy options (e.g., steal, hit and run, clutch bat).

JG: Sorry... one more item...

Also, I am mostly trying to point out that there are two types of "selection" processes... that is, the lineup selection before a game is a separate process from the selection of players to create a team. Consequently, the screens to do one of the processes are different than those for the other. This is an extremely important point that I fear has not been clarified before.

***(AF: Yes, we are still not clear what belongs to team creation vs what belongs to line-up selection.)***

JG: OK, here is the "vision" that I have for the pitcher selection page.

The "Dugout" on top should not be there, and the image in the background is different (I am trying to find the specific picture in my files - will send it later for inclusion).

MOST IMPORTANTLY, the table "style" is what I imagine being used.

***AF: What is a ‘pitcher-selection’ page? We have no such page nor do have any provision for it. Is this the same as the line-up selection page or is it the ‘starter’ selection page?***

***AF: Currently I understand that (for the lineup creation window)***

***a) for fielders they will be from P1...'***

***b) for pitchers they will be sp, rp etc***

***c) I have no information on 'batters'***

JG: My "vision" is that someone launches the game and they are presented with

1. An initial screen with login (I plan to use a ballpark view for the background of this screen), then

2. A screen with team selection and invitation stuff (like you have defined), then

3. A pitcher selection screen (the one I just sent), with the "Send invitation" button, then

4. They wait for the invitee to accept and fill out their lineup (I'll find a background picture for this), then

5. The indication that the invitee has completed his preparation, with a table of their fielders (for selection of their own lineup) - I'll find a picture for the background, then

6. A confirmation screen with their lineup as it would appear in the game screen, then

PLAY BALL!!

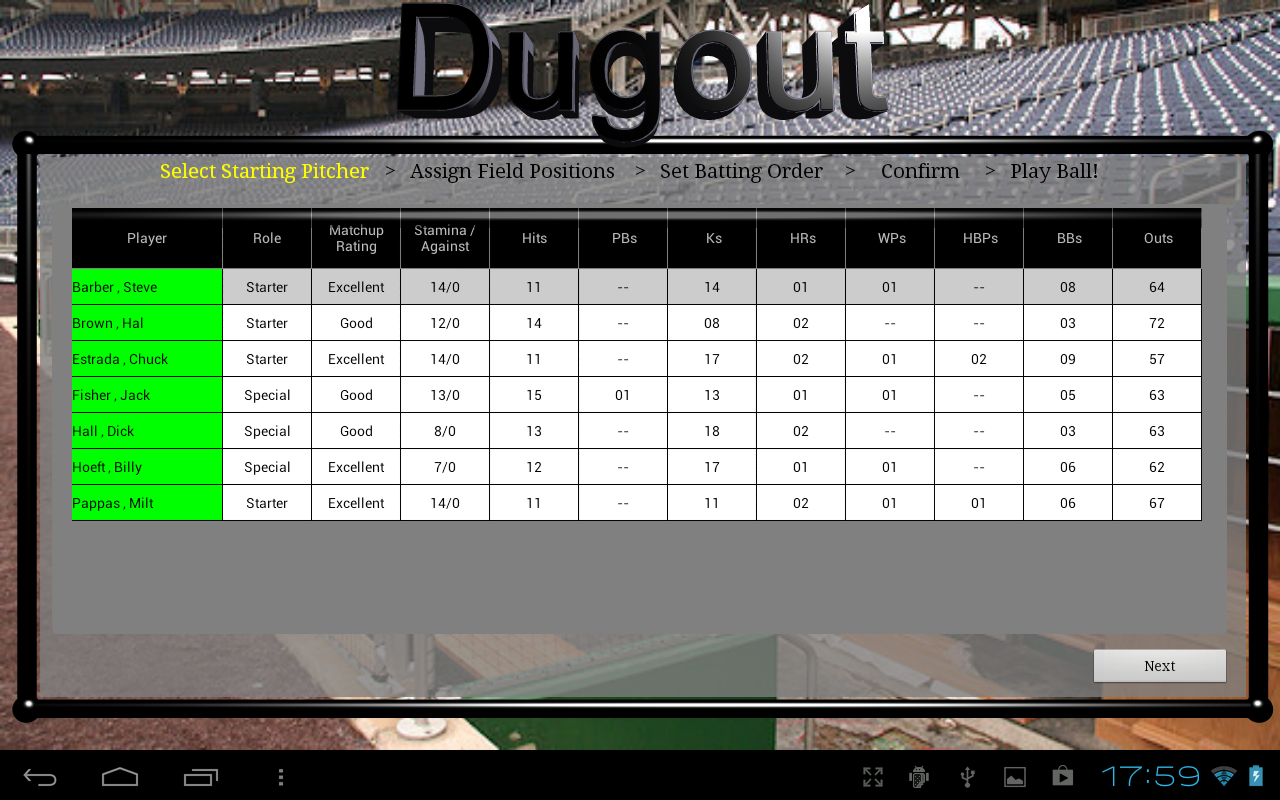
Are any of these steps unclear?

**AF: *The steps themselves are clear. However, such descriptions or visions are not implementable as they are incomplete and/or ambiguous. For example, if you say, ‘wait for the invitee…’ it needs to be specified whether we lock the screen (prevent the user from doing anything else), how is notified, what happens if he misses the notification. Therefore, a descriptive sentence hides all the specific implementation details, which remain unspecified, and guaranteed to cause confusion or issues later.***

JG: OK, here is the "vision" that I have for the pitcher selection page.

The "Dugout" on top should not be there, and the image in the background is different (I am trying to find the specific picture in my files - will send it later for inclusion).

MOST IMPORTANTLY, the table "style" is what I imagine being used.



First, I have attached the "table" style view of the fielders (please note that I am wanting the table in the back... I do not care about the popup window in the foreground. If you cannot identify the columns in the table, reply and I will send you a list) ...

OK, next, an explanation of why the table view:

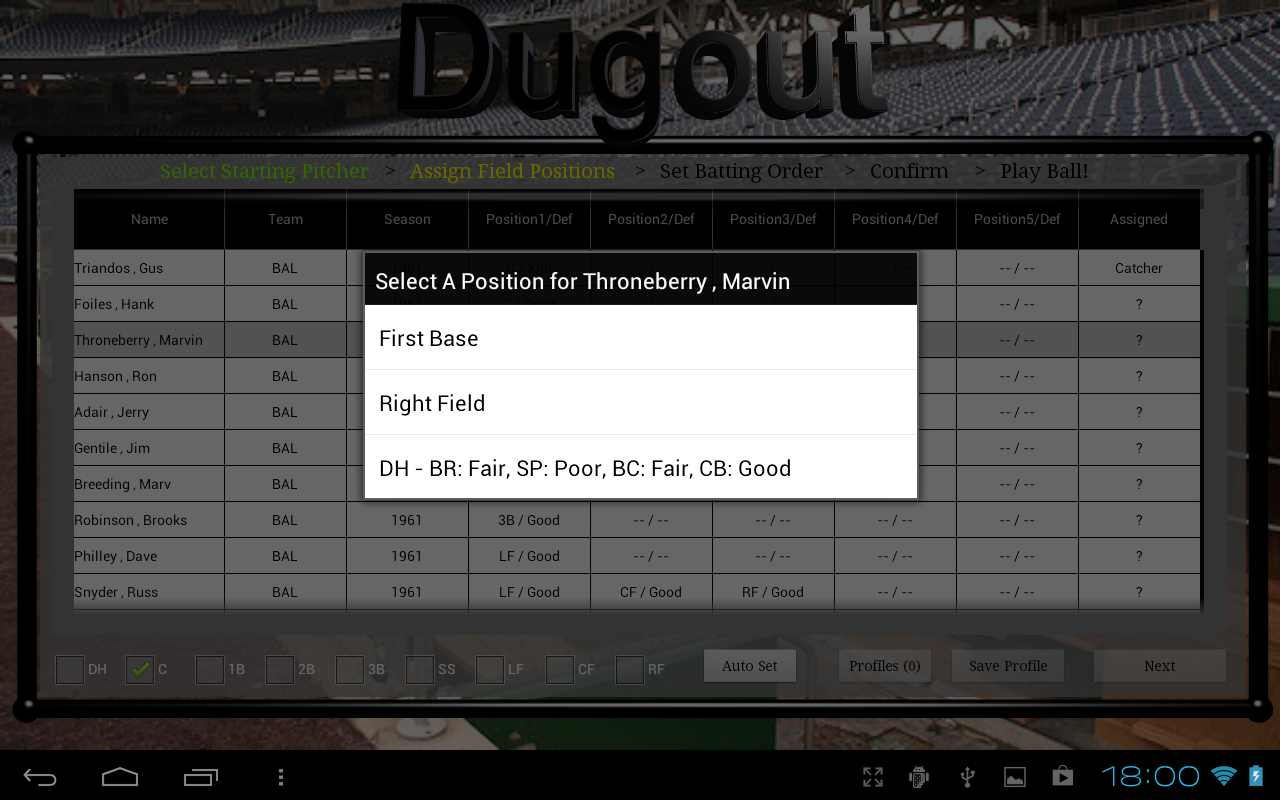
1. The owner needs to see the information that is listed in the table. All of that is important to the owner.

2. You are mostly on it with your proposed design, except that more information than you have listed is necessary.

3. When the owner is selecting the players to use in his lineup (for a specific game), only the fielders are needed in the selection screen.

4. I like your dropdown indicating which position (from among the player's possible positions).

5. I also like your dropdown for the batting spot.



I just looked at it again and realized that the screen shot I sent has the wrong columns.

The columns for the table when selecting the batters / fielders for a game should have the following headers:

Name

Team, Season

BR Rating

SP Rating

BC Rating

CB Rating

1Bs

2Bs

3Bs

HRs

K!s

BBs

HBPs

Outs

The owner needs this info to make his selections for the game.

I know that these are a lot of columns - I welcome your input for getting them all on one screen and legible. But the table style view is sensible to offer the owner all of the info at a glance.

***AF: NOTE: This is the first time I am seeing these screens/views. It seems this design was to have the lineup selection done in 3 steps, 1. select pitcher, 2. assign positions, 3. set batting order. We were not aware of this 3 step requirement. Our current design is based on two screens (starter selection, lineup selection). It seems the most relevant possible way to adapt our current UI is as follows:***

1. ***There is no UI for selecting the fielders in this design. Assuming we are doing things differently.***
2. ***Add fields to the starter selection screen (aka pitcher selection), but have no other variables in this UI. (so, step 1)***
3. ***Add fields to the lineup selection UI. Fold batting order and position selection together as two separate columns, both with drop downs***. (step 2 and 3)

***AF: Currently we have no requirement for the UI to enforce that all required positions in a field have been satisfied. I am assuming this may be a requirement. If we have to include such logic in the UI we will need clarification. Ideally, it seems, this should be a requirement for team selection, since a team where there are no fielders for some position can never be used to create a lineup etc. It would seem then that this should be a rule for team selection but currently we have no such requirement provided for team selection either.***

***AF: Is there are a rule that only a pitcher with ‘role’ S2 or S can be selected for starters? The above descriptions are too ambiguous. We synthesize the following rules for starter selections.***

1. ***Some will be unavailable based on history. We will not implement this for public/free teams. They will be implemented by the API by setting appropriate flags to data sent to UI.***
2. ***Pitchers with S2 and S rating can be used for starter selection. We will restrict the data sent to the UI by filtering out relief pitchers (role=RP).***

***Please confirm.***

***AF: The UI samples you have provided are from a table at full width. However, when you requested us to add the blog on the right side of the game, this reduced the width available to us and the current width is less than a full width tablet (table: approx. 1024px, current UI approx. 600 px. Therefore, fitting this information will require abbreviations and may not look too legible.***

# Rough sketches/notes

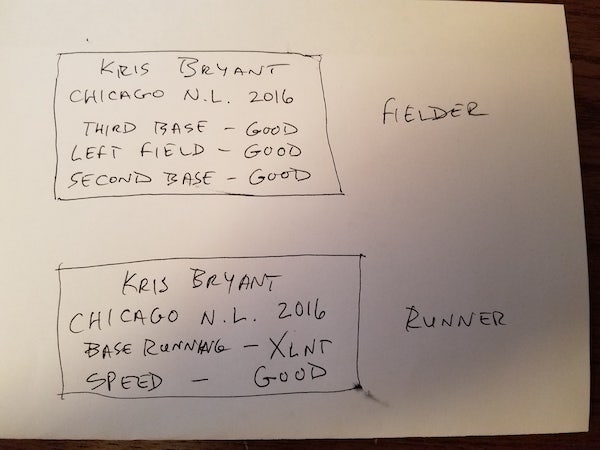


Figure 22 card sketch 1

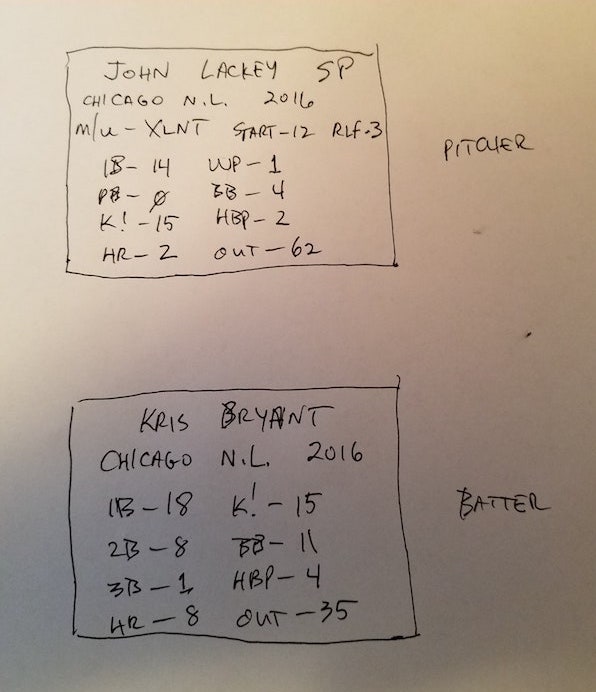


Figure 23 card sketch 2

# Open Issues

Following issues will have to be resolved. These issues mean that some functions will not be implemented in the first version:

1. In the current in-game UI, there are seven/five ‘move’ or action buttons only. There is no UI design for the owner to change the lineup during the game or change the player positions or batting order etc.. It was not indicated that this can be a requirement, but some of the descriptions in the emails indicate that this may be needed. To implement this, we need a design for this, with limits/bounds, etc., and API. May not be possible in the current cycle.

# API Samples

This is a sample API description

***isUserNameAvailable***

|  |  |
| --- | --- |
| **REST Mapping:** | *[GET] /authentication/isUserNameAvailable/?username=username0* |
| **Description:** | *Check if the username is available or not.* |
| **Parameters:** | *username* |
| **Return Value:** | *Result object, success must be true.* |
| ***Exceptions:*** | *UnableToComplyException is thrown if the supplied arguments are invalid. PanicException is thrown if there is an internal server error.* |