# Bucket Buddy
# Software Requirements Specifications

*Rafael Braga*

*Robert Howe*

*Jason Nance*

*2/12/2015*

| Revision History | | | |
|---|---|---|---|
| Date | Version | Author | Change Description |
| 1/22/2015 | 0.1 | Jason Nance | Initial skeleton, project description section |
| 1/29/2015 | 0.2 | Robert Howe | Added activity diagram and initial class diagram |
| 2/1/2015 | 0.3 | Jason Nance | Added some use cases and activity diagrams with descriptions of each |
| 2/1/2015 | 0.4 | Rafael Braga | Added the second class diagram and description of some use cases |
| 2/1/2015 | 0.5 | Robert Howe | Added roster use case, roster activity diagram, and descriptions |
| 2/4/2015 | 0.6 | Jason Nance | Added definitions of terms |
| 2/4/2015 | 0.7 | Rafael Braga | Added purpose of project |

# Table of Contents

# Introduction

## Purpose

The main motivation of this project is to build a smooth and efficient application that is pleasant to use. Sometimes you want to keep data from your favorite team and do not wish to use the archaic form of pen and paper. For this reason, we are proposing an easier and more organized new way of following teams. Applications focused in sports are common; however, they usually cost money. "Bucket Buddy" was idealized taking into consideration basketball fans and professionals in the area who want to keep personal data about their favorite teams, players and games but do not wish to spend money on it. Our goal is build an effective implementation which is good enough to equal paid applications in quality.

## Scope

We will need to create multiple activities in our app that will provide an interface for the user to store data relevant to a non-professional basketball game. To that end, we will need to design and create several classes along with their attributes and methods. An efficient database will need to be implemented that stores all of the profile information and statistics in a logical and straightforward manner, making data retrieval easy.

## Goals

The goal of this project will be to have an app that will enable the user to effectively and efficiently store information on basketball players and a team as a whole. The user will be able to retrieve that data in a similar effective manner. The storage and retrieval of the data will take place with an intuitive and sleek interface.

## Definitions

Some definitions of terms useful for understanding the domain of Bucket Buddy follow. A full specification of the rules of basketball is beyond the scope of this document, but it will be useful to define some of the statistics and game information that a user can record.

Scoring - a shot that goes through the hoop (is "made") from within the 3 point line awards the shooter's team 2 points; a shot made from outside the 3 point line is worth 3 points. A made free throw is worth 1 point
Field Goals Attempted (FGA) - the number of shots attempted by a player or team
Field Goals Made (FGM) - the number of shots made by a player or team
3 Pointers Attempted (3PA) - the number of shots attempted from beyond the 3 point line by a player or team
3 Pointers Made (3PM) - the number of shots made from beyond the 3 point line by a player or team

Offensive Rebound - when a player retrieves the ball after a shot attempt on the opponent's basket

Defensive Rebound - when a player retrieves the ball after a shot attempt on their team's basket

Assist - when a player passes to a player on their team, and that player proceeds to score

Turnover - when a player's actions cause the other team to gain possession of the ball via steal, offensive foul, out of bounds, or other event resulting in a change of possession

Steal - when a player takes possession of the ball from a player on the other team

Block - when a player makes contact with the ball during a shot attempt, causing it to miss

Foul - when a player commits one of several actions forbidden by the rules, such as making contact with an opponent while the opponent is shooting or pushing an opponent in an effort to get a rebound

Substitution - switching one or more of the 5 players on the court for one or more players sitting on the team's bench.  Substitutions can only occur at specified times while the ball is dead

Timeout - when a team has possession of the ball, they can call a timeout to stop the game clock.  Each team has a limited number of timeouts per half

# Project Description

## Overall Description

Bucket Buddy will consist of three main components.  The first, the roster menu, will allow a user to create, modify, or delete a roster made up of players, which themselves can be created, modified, and deleted.  This menu should be simple and easy to navigate.  Secondly, the user should be able to select a roster and begin collecting statistics and game information for that roster during a game using an intuitive graphical interface.  The interface will allow the user to undo or correct any mistaken input.  These statistics will be stored in a database on the phone's hard drive for later viewing.  The final component will be a display showing statistics already collected for a given roster or player, allowing the user to quickly assess the performance over time or per game of a team or individual.  This display should be accessible from the roster menu and while recording statistics during a game.

## Functional Requirements

1. The user must be able to create, modify, and delete a roster of players.
   a. The user must be able to create, modify, and delete players within rosters.
2. The user must be able to view statistics for each player on a given roster.
3. The user must be able to view statistics for a given roster as a whole.
4. The user must be able to choose a roster and begin recording statistics and game information in a game with that roster via a graphical interface.

a. Statistics include (but are not limited to) points, field goals attempted/made (including their location), 3 point shots attempted/made, rebounds (offensive and defensive), assists, turnovers, steals, blocks, fouls, and free throws attempted/made.
b. Game information includes (but is not limited to) substitutions, timeouts taken, current period (half or quarter), and the other team's score.
c. The user must be able to undo and/or correct a recent statistic entry in the event of a mistake.
d. The user must be able to view all aforementioned statistics from within the game interface.

## Nonfunctional Requirements

1. The roster menu must be simple and easy to navigate.
2. The statistics collection graphical interface must be intuitive, aesthetically pleasing, and performant.
3. The app must not cause an unnecessarily large drain on the battery.
4. The app must store data efficiently and minimize the amount of hard drive usage.
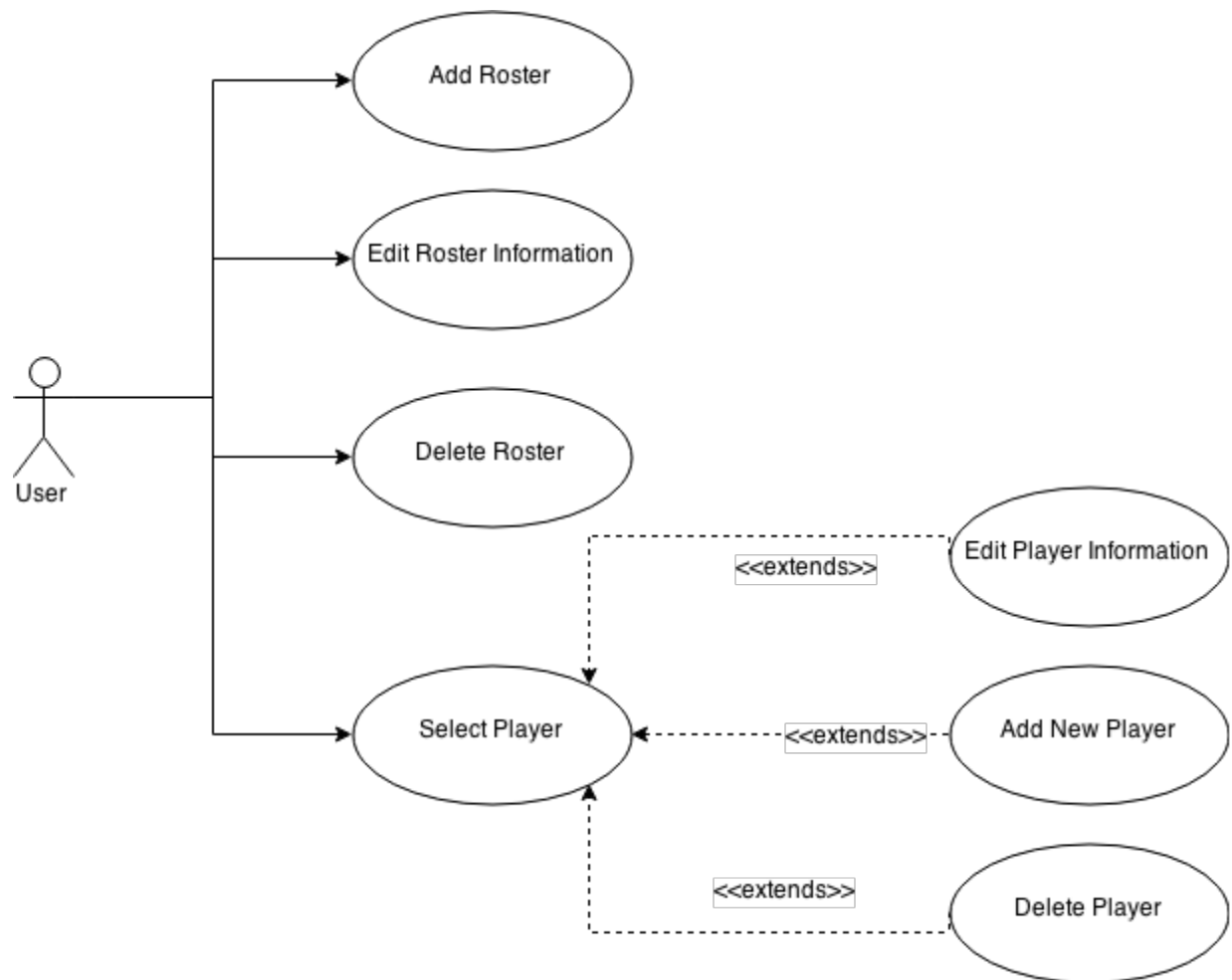5. The statistics reports must be useful and aesthetically pleasing.
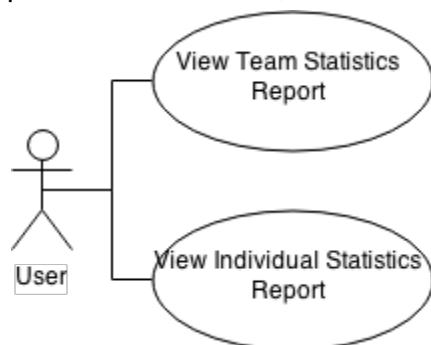
# Diagrams

## Use Case Diagrams
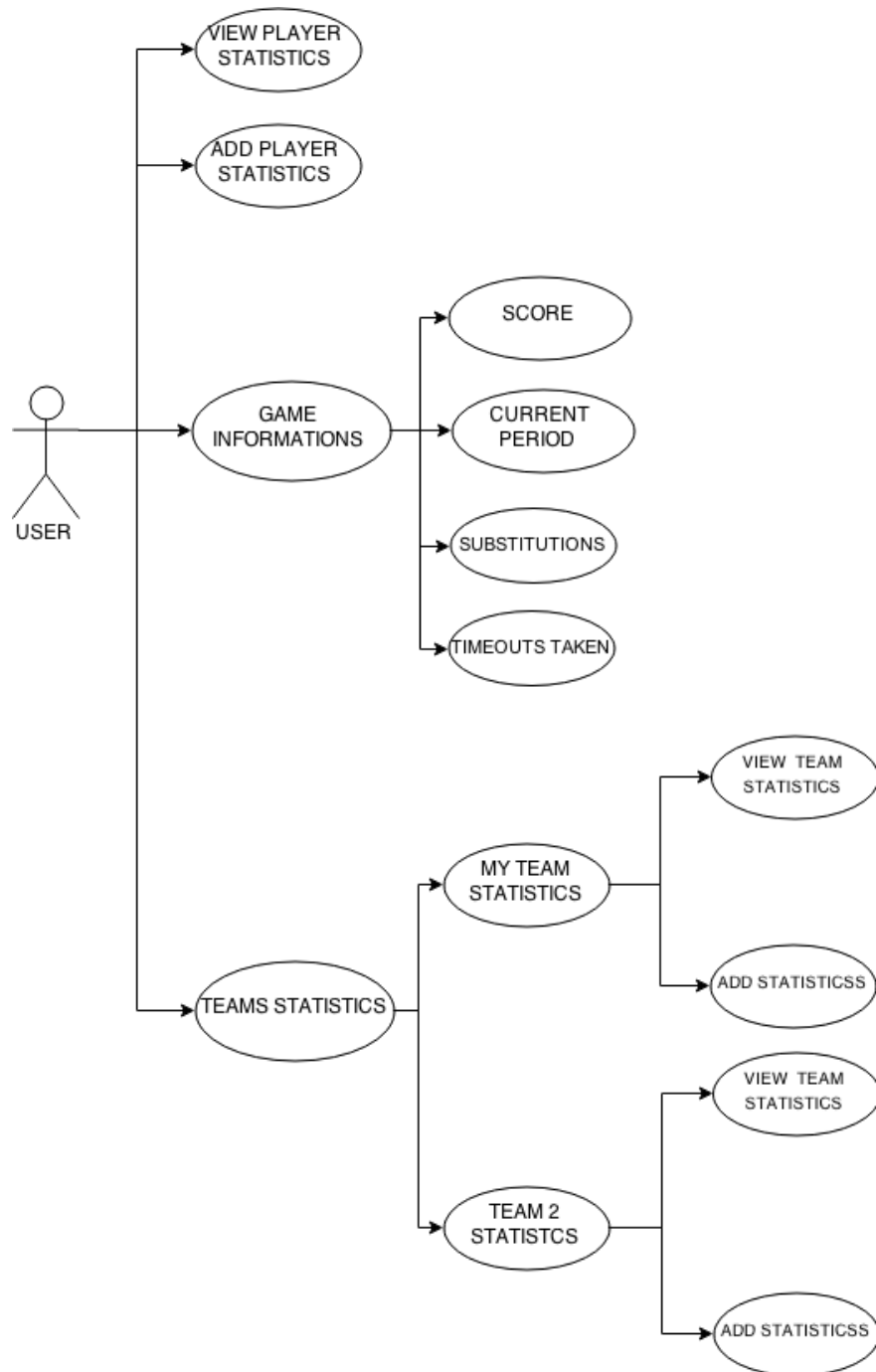
**Use Cases - Game Screen**

For the Game Screen view, the user will be able to view, add and edit information about the current game and players.

**Use Cases - Roster Menu**

From the roster menu, the user can choose to add a new roster, edit a current rosters name, delete a current roster or choose an individual player in which they can edit that players personal information, delete that player, or add a new player.
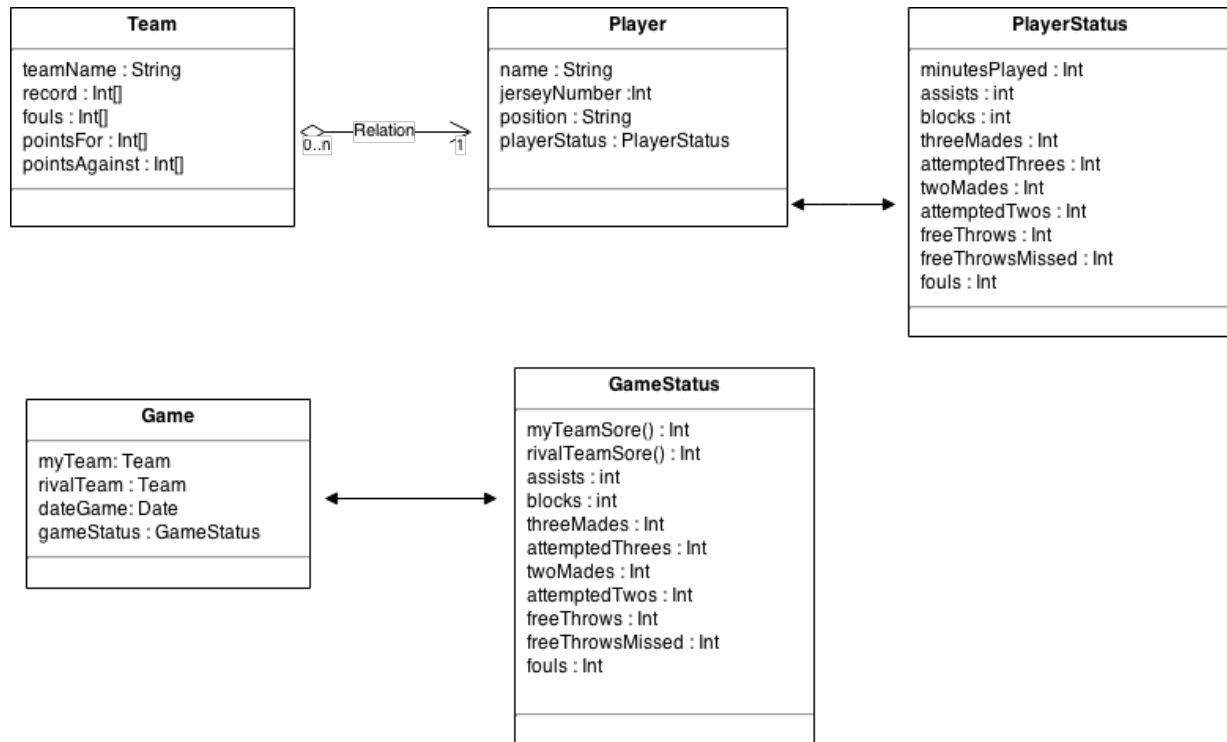


**Use Cases - View Statistics**

From the view statistics screen, the user can choose to view a report of their team's statistics or statistics of an individual.

## Class Diagram

| Team | | |
|---|---|---|
| teamName : String | | |
| record : Int[] | | |
| fouls : Int[] | | |
| pointsFor : Int[] | | |
| pointsAgainst : Int[] | | |

Relation 0..n — 1

| Player |
|---|
| name : String |
| jerseyNumber :Int |
| position : String |
| playerStatus : PlayerStatus |

| PlayerStatus |
|---|
| minutesPlayed : Int |
| assists : int |
| blocks : int |
| threeMades : Int |
| attemptedThrees : Int |
| twoMades : Int |
| attemptedTwos : Int |
| freeThrows : Int |
| freeThrowsMissed : Int |
| fouls : Int |

| Game |
|---|
| myTeam: Team |
| rivalTeam : Team |
| dateGame: Date |
| gameStatus : GameStatus |

| GameStatus |
|---|
| myTeamSore() : Int |
| rivalTeamSore() : Int |
| assists : int |
| blocks : int |
| threeMades : Int |
| attemptedThrees : Int |
| twoMades : Int |
| attemptedTwos : Int |
| freeThrows : Int |
| freeThrowsMissed : Int |
| fouls : Int |

## Class Diagram

The class diagram describes the system, its attributes, and its methods, giving us a wide vision of the software and relation between the objects.
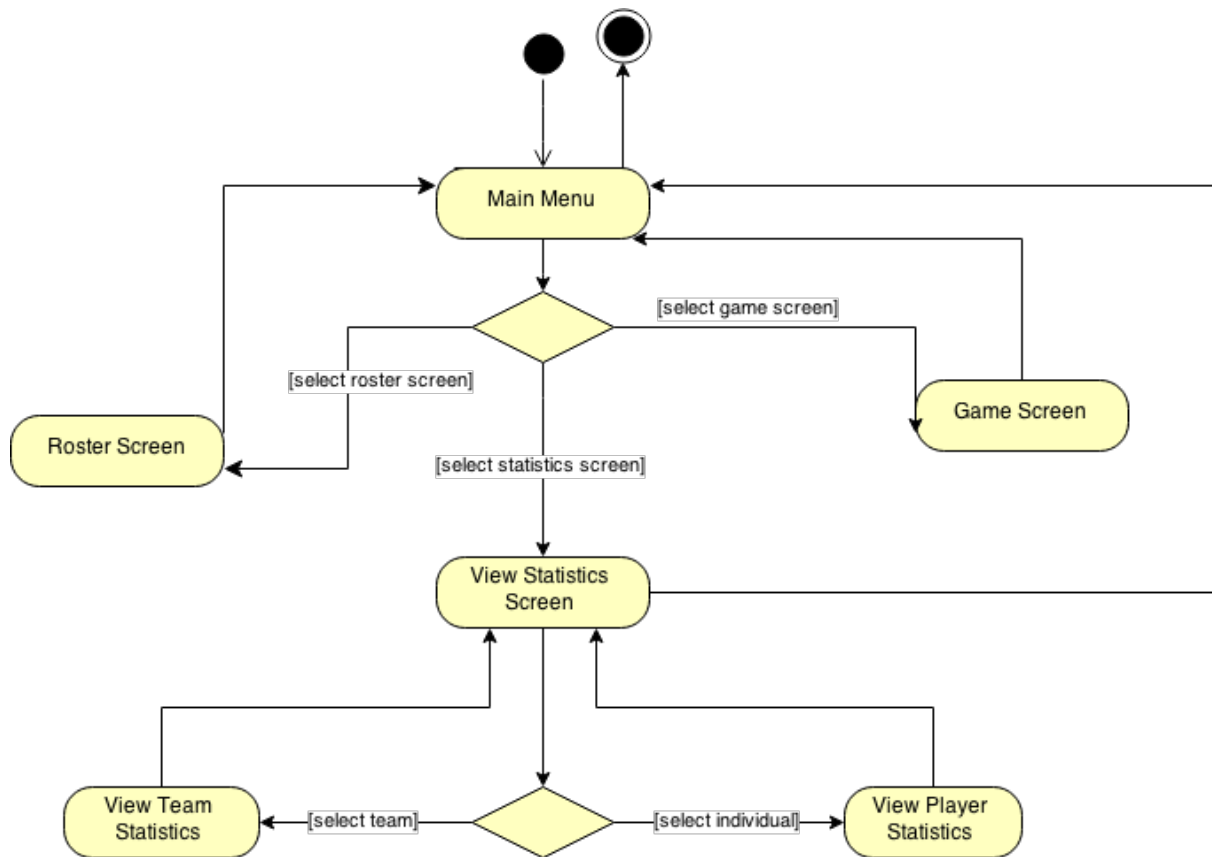
## Activity Diagrams
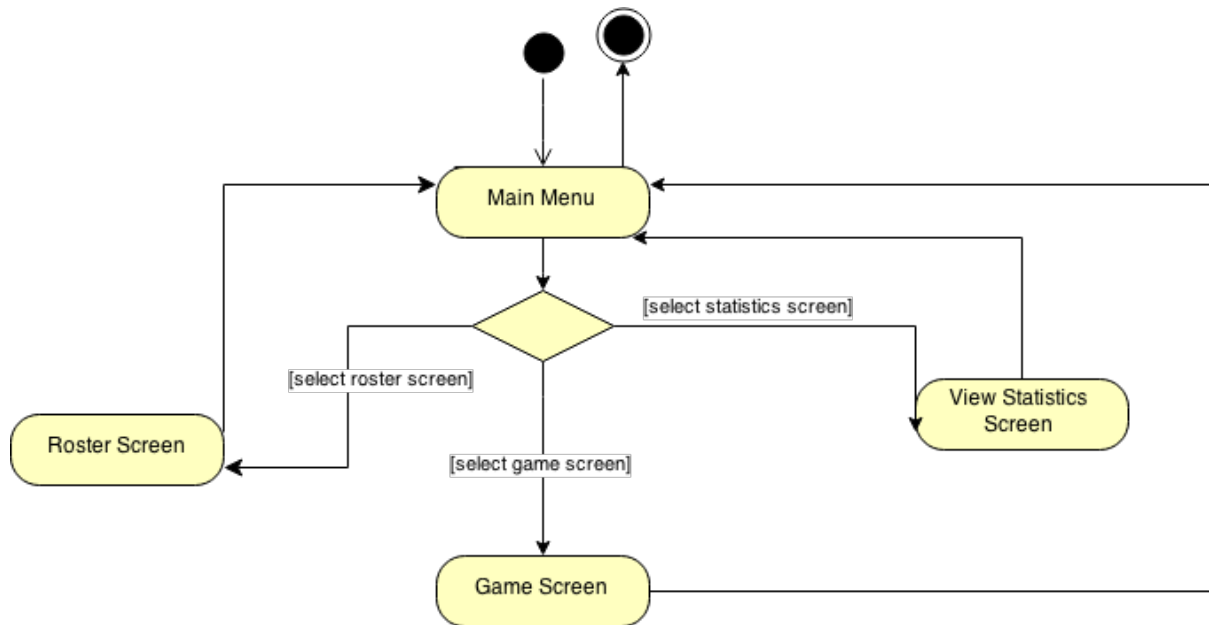


**Activity Diagram - Main Menu**

From the main menu, the user can choose to go to the roster screen, the game screen, or the view statistics screen. Once inside any of those screens, the user can optionally perform a number of actions before exiting back to the main menu and leaving the app.
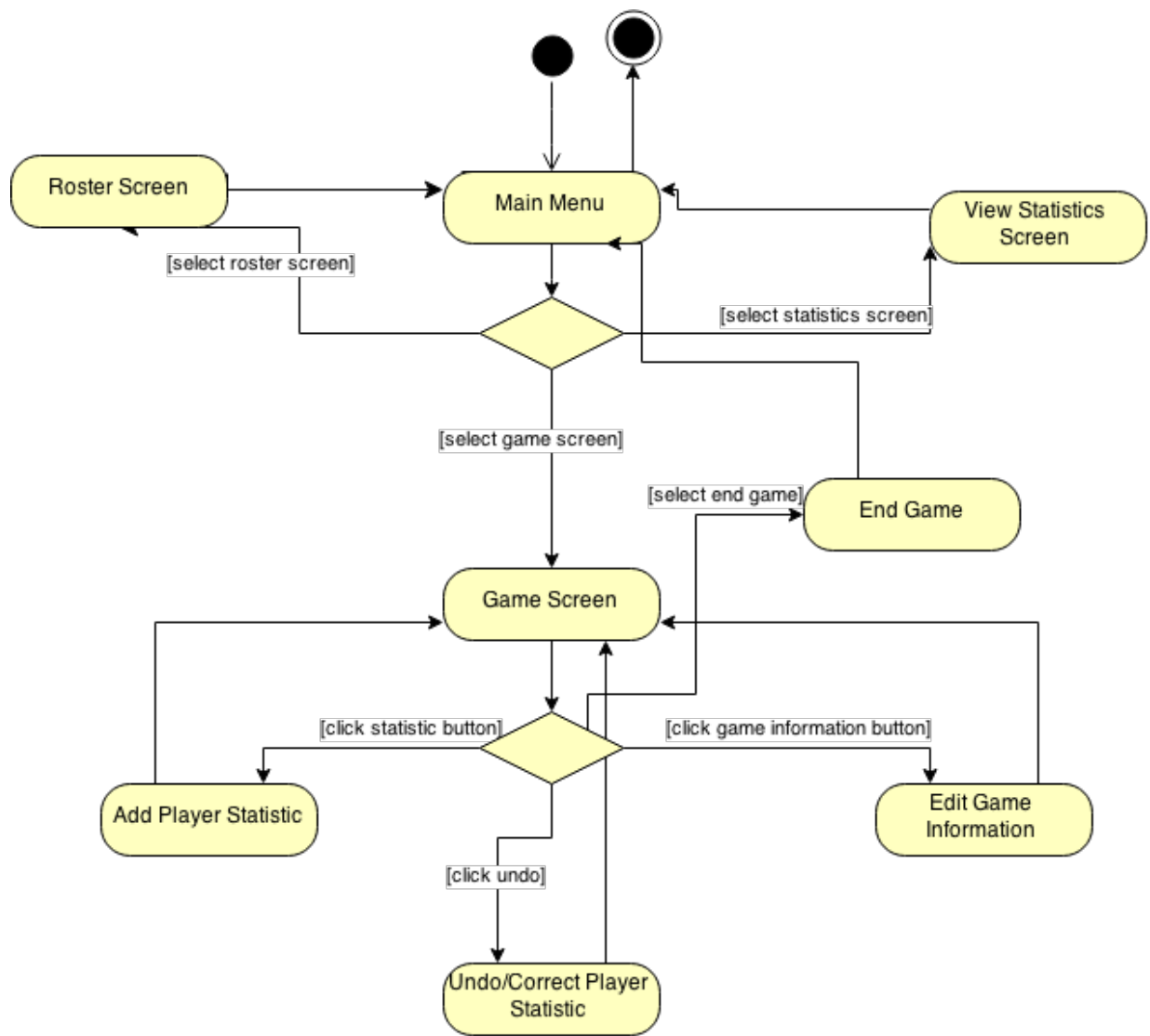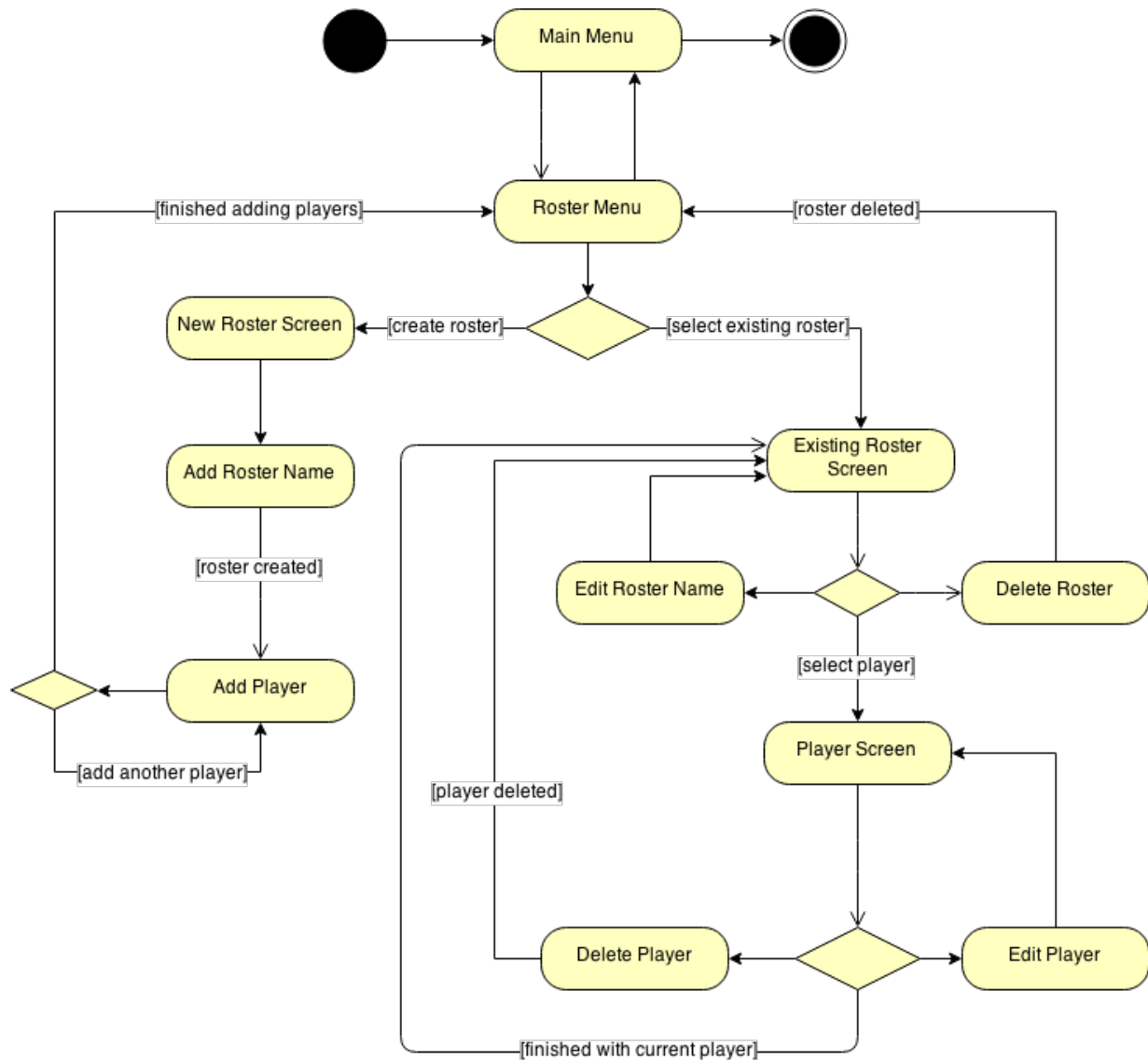
**Activity Diagram - Game Screen**

In the game screen, the user can perform several actions that will assist them in recording statistics.  By clicking any one of the buttons designated for recording statistics, the user can add a player statistic.  The user can also undo or correct a player statistic with the undo button should they make a mistake.  Options available by pressing a game information button will allow the user to edit information about the game (as opposed to individual player statistics).  Finally, once the game is over, the user can choose to end the game and will be returned to the main menu.

**Activity Diagram - View Statistics**

Upon reaching the view statistics screen, a user can choose a team -- allowing them to view statistics about the team -- or they can choose a player to see their individual statistics.  Either of these choices will allow the user the option to return to the view statistics screen, from which they can view other statistics reports or return to the main menu.

**Activity Diagram - Roster Menu**

Upon reaching the roster menu screen, a user can choose an existing team (roster) or choose to create a new team. Choosing an existing team allows them to edit the team name, delete the team, or choose a specific player on that team, at which point they can edit that players personal information or delete that player.  Choosing to create a new roster allows the user to create the team name then add as many players onto the roster as they see fit.