Claire Oberdorf
Dr. Arias
CMPT 220
26 April 2019

Abstract. This application, The Softball Game Stat Tracker will prompt a user, such as a coach, to enter the teams of a game and the statistics as the game progresses. It will neatly display all the statistics of the games as soon as they are entered. This is a great application where coaches and players can follow along with the statistics to determine how well the players are doing during the game and during the season. It is also an application that can save the data and statistics of one game and add it to the rest during the season for overall statistics. Because of the time and knowledge of the programmer, this application will only calculate the statistics of one singular game.

Introduction. The motivation of this application is to create a program that coaching staffs and players can use to organize the statistics from games where they can make decisions to help the team succeed. This application could be turned into one that organizes the statistics of all sports, but for this project it will only focus on a softball stat tracker. In higher level softball, coaches often rely on the numbers their players create. These numbers determine the performance of the players on the field, such as their batting average, on-base percentage, ERA, innings played, etc. With these specific numbers, coaches and players are able to see where there needs to be improvement and make tough decisions with the support of the numbers. Having all of this information to each specific type of player (offense, defense, pitching) keeps everything organized in order to help the team succeed and improve.

Detailed System Description. (UMLs on pages 4, 5, and 6) This application system, the Softball Game Stat Tracker will prompt the user such as a coach or stat keeper to enter the roster of their team. When entering the roster, the information included will be the first and last name of the player and their jersey number. Next, the user will choose the starting lineup by entering the names of the players again, along with the position they play (1 is the pitcher, 2 is the catcher, 3 is the third baseman, etc.). Ideally a coach or stat keeper for the team would want to start this application at the beginning of the season because it will keep all of the statistics throughout the season. These statistics are included in the three different categories of fielding, pitching, and hitting. The statistics for fielding include assists (A), errors (E), fielding percentage (FLD%), innings played (INN), putouts (PO), and total chances (TC). The pitching statistics include base on balls (BB), complete game (CG), earned runs (ER), earned run average (ERA), innings pitched (INN), loses (L), strikeouts (SO), saves (SV), save opportunities (SVO), wins (W), and hits against (H). Lastly, the hitting statistics include at bats (AB), batting average (AVG), base on balls (BB), caught stealing (CS), doubles (2B), hits (H), hit by pitch (HBP), home runs (HR), on base percentage (OBP), plate appearances (PA), runs (R), runs batted in (RBI), sacrifice bunt/fly (SAC), stolen bases (SB), strikeout (SO), and triple (3B).

The coach or stat keeper of the team will be able to update the statistics as the games are happening, or if they can enter them while the game is happening they can be entered after the game is compete. This application will only be able to be used by one user at a time but, ideally it can become an application that has a passcode and username where players and all coaches can log in and view the statistics for the player, overall season, or separate games.

Requirements. This application will address the problem of organizing the statistical information for sports teams and specifically softball. In the perfect application, the coaches will be able to store more than just a games statistics, such as the season schedule, statistics from all games during the season, and be able to write notes or messages on other teams scouting reports.

Literature Survey. There are a lot of other applications that organize and keep statistics of softball games. One application is the FreeSoftballStats. The FreeSoftballStats is a free application that keeps track of all types of softball (slow pitch, fast pitch, beer league, high school, or competitive softball team's). This application keeps track of the batting, hitting, and pitching stats. Another application through TeamSnap is called Softball Statistics Software. The Softball Statistics Software keeps all offensive, defensive, and pitching statistics along with letting the user create their own calculated statistical formulas.

User Manual. The user of the Softball Game Stat Tracker will open the application and first enter their teams roster if it is the first time they are opening the application for the new season. After the roster is entered into the application the user can choose the starting lineup for that game by entering the names and positions of the players. As the game is happening the user will enter the statistics such as a hit, stolen base, an earned run, or walk.

Conclusion. This application is a tool used to organize a softball teams' statistical data in order to make improvements on their performances individually, and as a whole. Ideally, the coach will be able to enter all player information and then the statistics for the game. At the end of games, the application will add and summarize the game statistics for that game and add them to the season statistics. Because of time and the skills of the programmer, the Softball Game Stat

Tracker will just present the application of entering the team roster and lineup for a game and then the stats of each player for that specific game.

References.

Liang, Y. Daniel. Introduction to Java Programming: Comprehensive Version. Boston: Pearson, 2015. Print.

UMLs.

Player.java
firstName: String lastName: String jerseyNumber: int position: String starter: String
+ Player() +Player(newFirstName: String, newLastName: String, newJerseyNumber: int, newPosition: String, newStarter: String) +getName(): String +getJerseyNumber(): String +getPosition(): String +getStarter(): String

Fielding.java
assists: int errors: int putOuts: int fieldingPercentage: double inningsPlayed: int totalChances: int
+ Statistics() +Statistics(newAssists: int, newError: int, newPutOuts: int, newFieldingPercentage: double, newInningsPlayed: int, newTotalChances: int) +getAssists(): int +getErrors(): int +getPutOuts(): int +getFieldingPercentage(): double +getInningsPlayed(): int +getTotalChances(): int

Pitching.java
baseOnBalls: int completeGames: int earnedRuns: int earnedRunAverage: double inningsPitched: double losses: int opponentBattingAverage: double strikeOuts: int saves: int saveOpportunites: int wins: int hitsAgainst: int
+ Statistics() +Statistics(newBaseOnBalls: int, newCompleteGames: int, newEarnedRuns: int, newEarnedRunAverage: double, newInningsPitched: double, newLosses: int, newOpponentBattingAverage: double, newStrikeOuts: int, newSaves: int, newSaveOpportunities: int, newWins: int, newHitsAgainst: int) +getBaseOnBalls(): int +getCompleteGames(): int +getEarnedRuns(): int +getEarnedRunAverage(): double +getInningsPitched(): double +getLosses(): int +getOpponentBattingAverage(): double +getStrikeOuts(): int +getSaves(): int +getSaveOpportunites(): int +getWins(): int +getHitsAgainst(): int

Hitting.java
atBat: int battingAverage: double baseOnBalls: int caughtStealing: int doubles: int hits: int hitByPitch: int

homeRuns: int onBasePercentage: double plateAppearences: int runs: int runsBattedIn: int sacrifice: int stolenBases: int strikeout: int triple: int
+ Statistics() +Statistics(newAtBat: int, newBattingAverage: double, newBaseOnBalls: int, newCaughtStealing: int, newDoubles: int, newHits: int, newHitByPitch: int, newHomeRun: int, newOnBasePercentage: double, newPlateAppearence: int, newRuns: int, newRunsBattedIn: int, newSacrafice: int, newStolenBases: int, newStrikeout: int, newTriple: int) + getAtBat(): int +getBattingAverage(): double +getBaseOnBalls(): int +getCaughtStealing(): int +getDoubles(): int +getHits(): int +getHitByPitch(): int +getHomeRuns(): int +getOnBasePercentage(): double +getPlateAppearences(): int +getRuns(): int +getRunsBattedIn(): int +getSacrifice(): int +getStolenBases(): int +getStrikeout(): int +getTriple(): int