THE PERFECT GAME PLAN: MILESTONE

Molly Smith
Dr. Arias
CMPT_220_112_17S
7 April, 2017

Abstract. The application will prompt the user to enter player personal information and player statistical information. It will neatly display the information entered. Theoretically, the application will also prompt the user to enter information about other factors about the players, like what the player ate before the game, how much sleep the player got the night before a game, and recovery time between games. Given the time constraint and knowledge of the creator, only stats and player roster information will be prompted to the user.

Introduction. The purpose of this application is to layout organized stats so that coaches and their players are able to design a game plan that will lead the team to success. Although the application is ultimately supposed to have many different sports available, this project will focus solely on basketball. Often times, coaches look at their players' stats (field goal percentage, assist-to-turnover ratio, minutes played, etc.) to see where improvement needs to be made. However, this application will also address the factors off the court that may influence these stats; these factors include amount of sleep the night before a game, pregame meal, amount of recovery days between days, and many more. Having this information nicely laid out in an application will give the user the ability to see if there is a pattern between these factors and a player's performance.

Detailed System Description. The system prompts the user to enter personal information about the player. This information includes player's first name, last name, position, jersey number,

weight, and height. The system also prompts the user to enter statistical information about the player. This information includes minutes played, field goals made, field goals attempted, field goal percentage, three-pointers made, three-pointers attempted, three-pointers percentage, free throws made, free throws attempted, free throw percentage, offensive rebounds, defensive rebounds, total rebounds, assists, turn overs, blocks, and steals.

The typical user will be coaches or managers in charge of the team's stats. The coach should be able to update players' and team stats after a game. However, this application will just be the beginning of this; it will have the coach create a 5 person roster and enter one set of stats. Ideally, the application will be used by the players, as well, to look at their stats, see upcoming events, and see posted messages from the coaches.

*UML Diagrams are found on Pages 4 and 5

Requirements. This application will address the problem of organizing statistical information for sports teams. Ideally, it will allow coaches to have their team's stats, calendar, messages, and scouting reports for other teams all in one place. In addition, it will allow coaches to see if certain outside factors relate to how his or her team performs.

Literature Survey. There are many apps that provide statistical organization tools. One app in particular is TeamSnap. TeamSnap allows a team to keep track of phone numbers, emails, schedules, and some basic stats.

DakStats is another program dealing with sports and statistics. The program records game, season, and career statistics. It also provides live stats for fans and people following the game online.

User Manual. The user will open the application. The application will prompt the user to enter his or her player's name, position, jersey number, height, and weight. The user will then be prompted to enter players' stats on includes minutes played, field goals made, field goals attempted, field goal percentage, three-pointers made, three-pointers attempted, three-pointers percentage, free throws made, free throws attempted, free throw percentage, offensive rebounds, defensive rebounds, total rebounds, assists, turn overs, blocks, and steals.

Conclusion. Ultimately, this application will be an organization tool for keeping track of team and player statistics, calendars and schedules, personal player information, and scouting reports. Ideally, the coach will be able to enter his or her players' statistics for a game, but they will also be able to make correlations for outside factors, such as food, sleep, and recovery. However, given the skills of the creator, this project will just present the program of entering a team roster and stats for each of the players entered.

References/Bibliography.

Johnson, Richard A. "JAVA DATABASE CONNECTIVITY USING JAVA DB (APACHE DERBY): A TUTORIAL." *International Journal of Information, Business and Management* 8.3 (2016): 295-309. *ProQuest.* Web. 7 Apr. 2017.

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

Player.java	
	firstName: String lastName: String jerseyNumber: int position: String heightInches: int weightPounds: int
+	Player() Player(newFirstName: String, newLastName: String, newJerseyNumber: int, newPosition: String, newHeightInches: int, newWeightPounds: int) getName(): String getNumber(): String getPosition(): String getHeight(): String getWeight(): String

Statistics.java
minutesPlayed: int fieldGoal: int fieldGoalAttempts: int fieldGoalPercentage: double threePoint: int threePointAttempts: int threePointPercentage: double freeThrow: int freeThrowAttempts: int freeThrowPercentage: double offensiveRebounds: int defensiveRebounds: int totalRebounds: int personalFouls: int assists: int turnOvers: int blocks: int

	steals: int
+	Statistics()
+	Statistics(newMinutesPlayed: int, newFieldGoal: int, newFieldGoalAttempts: int, newFieldGoalPercentage: double, newThreePoint: int, newThreePointAttempts: int, newThreePointPercentage: double, newFreeThrow: int, newFreeThrowAttempts: int, newFreeThrowPercentage: double, newOffensiveRebounds: int, newDefensiveRebounds: int, newTotalRebounds: int, newTotalRebounds: int, newPersonalFouls: int, newAssists: int, newTurnOvers: int, newBlocks: int, newSteals: int)
+	getMinutes(): int
+	getFieldGoal(): int
+	getFieldGoalPercentage(): double
+	getThreePoint(): int
+	getThreePointPercentage(): double
+	getFreeThrow(): int
+	getFreeThrowPercentage(): double
+	getRebound(): int
+	getFouls(): int
+	getAssists(): int
+	getTurnOver(): int
+	getBlock(): int
+	getSteal(): int