Capstone Analytics

Software Application Programming Guide

Version 1.3

Table of Contents

1 Introduction

1.1 Scope of the Product

The product has extracted annual statistics and salaries from publicly provided databases. With both performance based statistics and current salaries, we have done statistical analysis to determine the fair market value of a player. Once the statistical analysis is complete, players that are underpaid based on performance would be identified as optimal candidates for acquisition. Conversely overpaid players could be avoided. Ultimately the user could use these tools to maximize their profits by ensuring they avoid signing bad contracts.

1.2 Definitions, acronyms, and abbreviations

1.2.1 LAMP

LAMP is an acronym for a web service stack named originally for its four main components: Linux, Apache Web Server, MySQL, and PHP. LAMP is used to develop complete website solutions, and may include additional software packages dependent on user preference.

1.2.2 Regression

Regression, or linear regression, is a term in statistics for a method of modeling the relationship between two variables. Linear regression is the primary function used for the statistical analysis of the dataset.

1.2.3 Mean

Mean is the mathematical average of a variable in question.

1.2.4 Regression

Standard Deviation is a mathematical is used to measure the amount of dispersion of a set of values over a population. The closer standard deviation is to zero, the closer the population is to the mean value.

1.2.5 Sum

Sum is a synonym for total. Summing a dataset is the first step in finding the mean.

1.2.6 Intercept

In Mathematics, the intercept is the point on the Y-Axis that a line crosses, and is used for the slope intercept formula for a line. $y=mx+b$ where b is the intercept.

1.2.7 Slope

In Mathematics, the slope of a line is defined as the rise over the run where the rise is the change in y and the run is the change in x between two points.
$(y2-y1)/(x2-x1)$

1.3 References

http://www.seanlahman.com/baseball-archive/statistics
Used for statistics references.

2 General Description

2.1 Product Perspective

Forbes evaluates Major League Baseball at approximately 36 billion dollars and the average team being worth 1.2 billion dollars. Even with these large values, payroll for players make up a large part of the expenses for baseball clubs. Depending on revenue and revenue sharing amongst the teams, MLB has attempted to minimize the advantage larger clubs have over small clubs. However, the 2014 payroll shows a gap of 190 million dollars between the highest and lowest payroll. This product would help smaller clubs evaluate players on an objective basis in order to identify players that should be acquired at a minimal cost.

2.2 Product Functions

The software will allow users to select up to four statistical categories to evaluate. The user will be able to weigh the importance of each statistic being evaluated. This will allow the user to increase the importance of power (homeruns) or contact (hits) hitting styles as appropriate to find their ideal replacement player. After the search is complete, the program will return the players the most undervalued players in a list.

2.3 User Characteristics

The initial market for users of this program would be teams and agents involved in MLB. However, expansions should be made into the fantasy sports market, as there is a much larger audience that could offset the cost of production. Further releases could focus on development of a mobile specific web application or smart phone application using the same design to evaluate players.

2.4 General Constraints

The current version 1.3 will be a web based only application. The user will require internet access in order to use the application. Other limitations are that the user will require their own web browser in order to access the site.

2.5 Assumptions and Dependencies

The assumptions will be that the user will require a compatible web browser. Web browsers supported include Chrome, Internet Explorer, Safari and Firefox. Versions updated as of the 15th of April 2015 should be compatible. View of the website will also depend on firewall settings and not being blocked by other safe viewing mechanisms.

3 Specific Requirements

3.1 Database

~~A data base will be real-time and stored on another website the front end web application will access the statistics of each player to be evaluated.~~ A database will need to be stored on the web application server in order to allow the front end to access statistics of players to be evaluated.

3.1.1 Master

Master will account for one member fields in the database and will contain information to identify the baseball player.

3.1.2 Salaries

Salaries will list the salaries that the players received during the 2014 season, and will be a primary dataset for regression analysis.

3.1.3 Teams

Teams will be a member field in the database to account for the team a player is currently under contract with.

3.1.4 Appearances

Plate appearance will be a member field in the database to account for how many times the player has appeared for home plate.

3.1.5 Batting

Batting will account for multiple member fields in the database and will contain multiple variables that are standard to baseball. This will be an integral part of the linear regression calculation.

3.1.6 Fielding

Fielding will account for multiple member fields in the database and will contain multiple variables that are standard to baseball.

3.1.7 Pitching

Pitching will account for multiple member fields in the database and will contain multiple variables that are standard to baseball.

3.1.8 Users

The Users table in the database will contain registered users of the system and their credentials.

3.2 Security

The web application will require security to ensure that subscription services are being used. Username and Password will be required. Those will be obtained via e-mail through subscription services.

3.2.1 Username

Usernames will be verified as a unique id prior to being issued by subscription services. Once username is established a field will be provided for login and correct entry will be required to gain access to web application.

3.2.2 Password

Passwords will be issued by subscription services. Passwords can be changed but must be verified as 8 characters at least one upper and lowercase letter, one number, and a special character (!@#$%^&*()_-+/\':,?{}[]~). Password entry field should be next to or below Username.

3.3 Application

The web application will provide analysis of baseball players using statistics with salary to determine what the player's market value is.

3.3.1 Linear Regression

Linear regression is an approach for modeling the relationship between a scalar dependent variable y (salary) and one or more explanatory variables or independent variable (this will be a baseball statistic) denoted X. The application will utilize linear regression to determine fair market value.

~~3.3.2 Standard Deviation~~

~~The standard deviation is a measure that is used to quantify the amount of variation or dispersion of a set of data values. The application will apply this variation to determine the variation in salary. This will also be based upon the choice of statistic.~~

3.3.3 Minimum Plate Appearances

This will be used as population to control to provide more meaningful data sets. It will serve to remove players who did not play a statistically significant amount of time.

3.4 Application GUI

The web application GUI will provide choice and feedback to the user to manipulate the data for their use.

3.4.1 Statistic Drop Down Menu

This will provide a drop down menu to choose from batting statistics in order to perform analysis on the database and provide feedback to the user.

3.4.2 Minimum Plate Appearances Drop Down Menu

This will provide a drop down menu to choose minimum plate appearances in order to filter players who did not have a significant amount of plate appearances. This will remove players from consideration to provide tailored results.

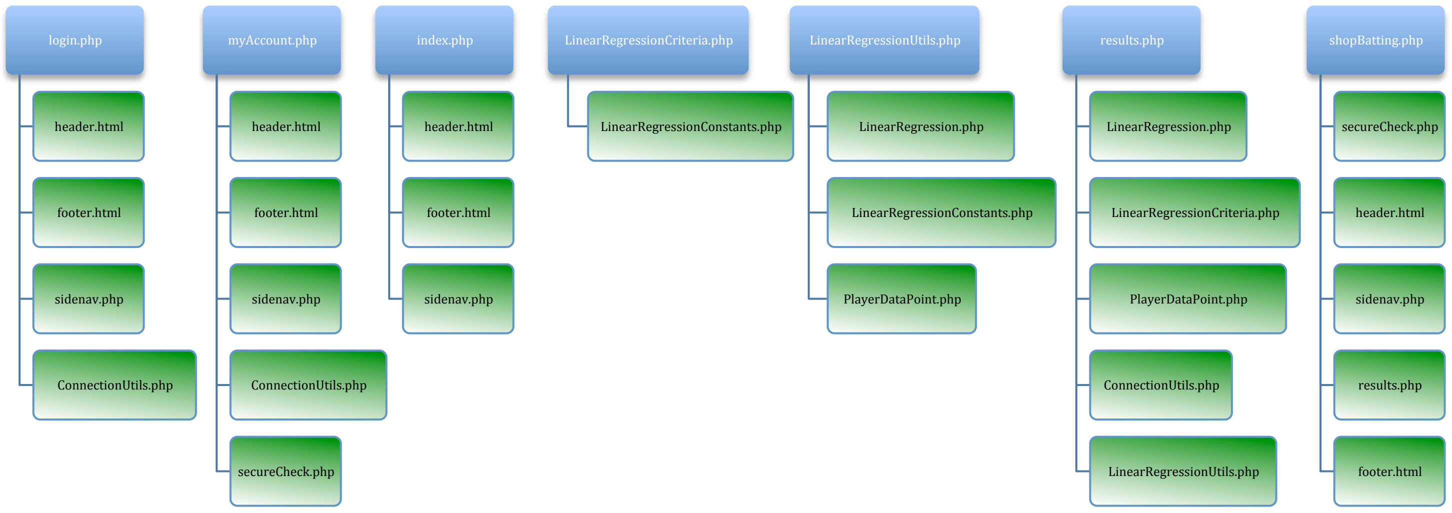
3.4.3 Data Results Return

Results will return players sorted by the difference between salary and expected salary. Players with the largest negative difference will be listed first. Data table will provide player name, statistic being used, salary, and expected salary.

4 Diagrams

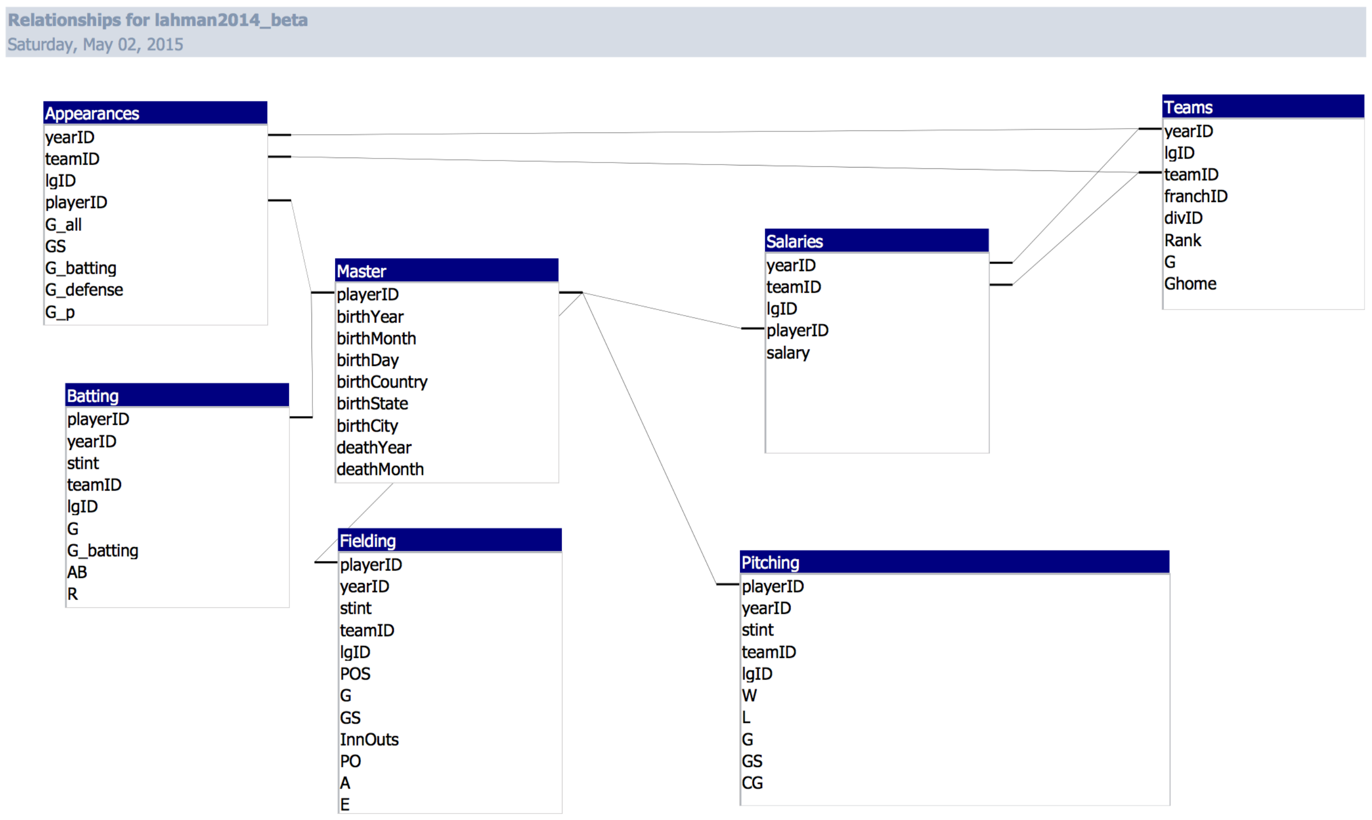
4.1 File Relationships

In the following diagram, main files are represented by blue and the files that are included by it are represented by green. Some files are not included, as main files if they do not have includes.



4.2 Database Table Relationships

The following table shows the relationship between the database tables.



5 LinearRegression.php

LinearRegression.php holds the definition of the LinearRegression class.

5.1 Variables

Variables for this class are set to private and are accessible through setter and getter functions and are declared private.

5.1.1 meanX

The variable meanX is used to hold the value of the mean or mathematical average of the set of values of the set X.

5.1.2 stddevX

The variable stddevX is used to hold the value of the standard deviation of the values in the set X.

5.1.3 sumX

The variable sumX is holds the value of the sum of all the numbers in the set X.

5.1.4 sumXsqr

The variable sumXSqr holds the sum of the squared difference between the mean and the value X.

5.1.5 meanY
The variable meanY is used to hold the value of the mean or mathematical average of the set of values of the set Y.

5.1.6 stddevY
The variable stddevY is used to hold the value of the standard deviation of the values in the set Y.

5.1.7 sumY
The variable sumY is holds the value of the sum of all the numbers in the set Y.

5.1.8 sumYsqr
The variable sumYSqr holds the sum of the squared difference between the mean and the value Y.

5.1.9 psum
The variable psum holds the sum of the players in the dataset.

5.1.10 n
The variable n holds the count.

5.1.11 slope
The variable slope holds the slope of the linear regression line for the slope intercept equation

5.1.13 intercept
The variable intercept holds the Y intercept of the slope intercept equation for the linear regression function.

5.2 Class Functions

5.2.1 getMeanX()

Summary
A getter function for the variable meanX

Definition
public function getMeanX()

Parameters
None

Return Value
The return value is that of meanX, and is dependent upon it.

5.2.2 getStdDevX()

Summary
A getter function for the variable stddevX

Definition
public function getStdDevX()

Parameters
None

Return Value
The return value is that of stddevX, and is dependent upon it.

5.2.3 getSumX()

Summary
A getter function for the variable sumX

Definition
public function getSumX()

Parameters
None

Return Value
The return value is that of sumX, and is dependent upon it.

5.2.4 getSumXSquared()

Summary
A getter function for the variable sumXsq

Definition
public function getSumXSquared()

Parameters
None

Return Value
The return value is that of sumXsq, and is dependent upon it.

5.2.5 getMeanY()

Summary
A getter function for the variable meanY

Definition
public function getMeanY()

Parameters
None

Return Value
The return value is that of meanY, and is dependent upon it.

5.2.6 getStdDevY()

Summary
A getter function for the variable stddevY

Definition
public function getStdDevY()

Parameters
None

Return Value
The return value is that of stddevY, and is dependent upon it.

5.2.7 getSumY()

Summary
A getter function for the variable sumY

Definition
public function getSumY()

Parameters
None

Return Value
The return value is that of sumY, and is dependent upon it.

5.2.8 getSumYSquared()

Summary
A getter function for the variable sumYsq

Definition
public function getSumYSquared()

Parameters
None

Return Value
The return value is that of sumYsq, and is dependent upon it.

5.2.9 getPsum()

Summary
A getter function for the variable psum.

Definition
public function getPsum()

Parameters
None

Return Value
The return value is that of psum, and is dependent upon it.

5.2.10 getCount()

Summary
A getter function for the variable n.

Definition
public function getCount()

Parameters
None

Return Value
The return value is that of n, and is dependent upon it.

5.2.11 setMeanX()

Summary
A setter function for the variable meanX

Definition
public function setMeanX($mean)

Parameters
$mean - a variable defined to accept the value which meanX will be set to.

Return Value
None

5.2.12 setStdDevX()

Summary
A setter function for the variable stddevX

Definition
public function setStdDevX($stdDev)

Parameters
stdDev – The variable defined to accept the value which will be stored in stdDevX

Return Value
None

5.2.13 setSumX()

Summary
A setter function for the variable sumX

Definition
public function setSumX($sum)

Parameters
sum – Variable defined to accept the value that will be stored in sumX

Return Value
None

5.2.14 setSumXSquared()

Summary
A setter function for the variable sumXsq

Definition
public function setSumXSquared($sumSquared)

Parameters
sumSquared – Variable defined to accept the value that will be stored in sumXsq.

Return Value
None

5.2.15 setMeanY()

Summary
A setter function for the variable meanY

Definition
public function setMeanY($mean)

Parameters
mean – variable to accept the value to be stored in meanY

Return Value
None

5.2.16 setStdDevY()

Summary
A setter function for the variable stddevY

Definition
public function setStdDevY($stdDev)

Parameters
stdDev – Variable to accept the value that will be stored in stdDevY

Return Value
None

5.2.17 setSumY()

Summary
A setter function for the variable sumY

Definition
public function setSumY($sum)

Parameters
sum – Variable to accept the value that will be stored in sumY;

Return Value
None

5.2.18 setSumYSquared()

Summary
A setter function for the variable sumYsq

Definition
public function setSumYSquared($sumSquared)

Parameters
sumSquared – Variable to accept the value that will be stored in sumYsq.

Return Value
None

5.2.19 setPsum()

Summary
A setter function for the variable psum.

Definition
public function setPsum($pSum)

Parameters
pSum – Variable to accept the value that will be store in psum.

Return Value
None.

5.2.20 setCount()

Summary
A setter function for the variable n.

Definition
public function setCount($count)

Parameters
count – Variable to accept the value that will be stored in n.

Return Value
None.

5.2.21 getCorrelationCoefficient()

Summary
A getter function for the variable r. The correlation coefficient is calculated within this function.

Definition
public function getCorrelationCoefficient()

Parameters
None

Return Value
Returns the calculated value of the correlation coefficient.

5.2.22 getSlope()

Summary
A getter function for the variable slope. The value is calculated within this function and represents the slope of the linear regression line.

Definition
public function getSlope()

Parameters
None

Return Value
Returns the value of the slope for the linear regression function.

5.2.23 getIntercept()

Summary
A getter function for the variable r. The correlation coefficient is calculated within this function.

Definition
public function getIntercept()

Parameters
None

Return Value
Returns the value of the intercept for the Linear regression function

5.2.24 getY()

Summary
A function to calculate the Y value in the regression function for any value of X input based on the slope and intercept.

Definition
public function getY($x)

Parameters
x – A variable to hold the x value that will be used to calculate the Y value of the function.

Return Value
This function returns the Y value associated or expected for the input value of x.

5.2.25 toString()

Summary
This function provides a method to output an element

Definition
public function toString()

Parameters
None

Return Value
A String representing the data in an element

6 LinearRegressionConstants.php
LinearRegressionConstants.php holds the definition of the LinearRegressionConstants, and PlayerDataPointConstants classes.

6.1 class LinearRegressionConstants
The LinearRegressionConstants.php holds the definition of the LinearRegression class.

Summary
This class will hold the calculated regression variables for analysis and selection of players

Definition
public class LinearRegressionConstants()

Parameters
None

Return Value
None

6.2 class PlayerDataPointConstants
The LinearRegressionConstants.php holds the definition of the PlayerDataPointConstants class.

Summary
This class will hold information related to the player

Definition
public class LinearRegressionConstants()

Parameters
None

Return Value
None

7 LinearRegressionCriteria.php
LinearRegressionCriteria.php is a utility class for building SQL statements for harvesting LinearRegression records/objects.

7.1 Variables

Variables for this class are set to private

7.1.1 tableX

The variable tableX is used to store the name of the desired table X, used for joining

7.1.2 tableY

The variable tableY is used to store the name of the desired table Y

7.1.3 rowX

The variable rowX is used to store the rowX value

7.1.4 rowY

The variable rowY is used to store the rowY value

7.1.5 jTables

The array jTables holds the names of the join tables.

7.1.6 joinTablesSql

The joinTablesSql variable holds the sql statement for the join tables, table names are added to the statements.

7.1.7 joinSql

The joinSql variable holds a string with the sql statement for the joins, created by the functions

7.1.8 playerJoinTables

The playerJoinTables array holds the tableName for the player join tables. The first entry is master, for the master table. This array tells the sql where to find the information in the joins.

7.1.9 playerJoinTablesSql

The playerJoinTablesSql is a variable that holds the sql statement for the player data joins.

7.1.10 comparisonSql

The comparisonSql variable holds the sql statement for the search criteria for a given table, column, value, and comparison.

7.2 Class Functions

7.2.1 _construct()

Summary

The _construct function is a non-default constructor for the class which has 4 *parameters* and serves to set up the variable definitions.

Definition

public function getMeanX($tableX, $rowX, $tableY, $rowY)

Parameters
tableX – Variable that holds the name of the table for the x axis
rowX – Variable that holds the name of the row in table x of interest
tableY – Variable that holds the name of the table for the y axis
rowY- Variable that holds the name of the row in the y table of interest

Return Value
None

7.2.2 addJoin()

Summary
The addJoin function is used to adding a join for two given tables on the left and right columns.

Definition
public function addJoin($tableLeft, $columnLeft, $tableRight, $columnRight)

Parameters
tableLeft – This variable is the name of the left table for the left column in the join.
columnLeft – This variable is the name of the column which is included in the join from the tableLeft table.
tableRight – This variable is the name of the right table for the right column in the join.
columnRight – This variable is the name of the column which is included in the join from the tableRight table.

Return Value
None

7.2.3 addComparison()

Summary
The addComparison function is used to add search criteria for a given table, column, value, and comparison, generating an sql statement.

Definition
public function addComparison ($table, $column, $value, $comparison)

Parameters
table – This variable will hold the name of the table used to make the comparison
column – This variable will hold the name of the column used to make the comparison
value – This variable holds the value that the column data is going to be compared to
comparison – This variable holds the comparison operator.

Return Value
None

7.2.4 getTableAlias()

Summary
The getTableAlias function is used to get the table alias if one has already been created for a table. This function is marked as private.

Definition
private function getTableAlias ($tableName)

Parameters
tableName – The name of the table that we are searching for the alias of.

Return Value
Returns the alias of the table if one is found or the original table name if no alias is found.

7.2.5 updateJoinTables()

Summary
The updateJoinTables function is used to update tables that need to be added to the query. This function is set to private.

Definition
private function updateJoinTables ($tableName)

Parameters
tableName – The name of the table that needs to be updated.

Return Value
None

7.2.6 getLinearRegressionSql()

Summary
The getLinearRegressionSql function is used to create the sql statement

Definition
public function getLinearRegressionSql()

Parameters
None

Return Value
A string representing the SQL statement for the linear regression function

7.2.7 getPlayerDataPointSql()

Summary
The getPlayerDataPointSql function is a utility function to build an SQL statement to gather

records for PlayerDataPoints

Definition
public function getPlayerDataPointSql()

Parameters
None

Return Value
A string representing the SQL statement for player data points.

8 LinearRegressionUtils.php

LinearRegressionUtils.php has several utility functions related to linear regression.

8.1 rowToLinearRegression()

Summary
This function will take the result row that utilizes the constant names and builds a LinearRegression object from it.

Definition
public function LinearRegressionConstants($row)

Parameters
row – This variable represents the row that utilizes the constant names.

Return Value
This function returns a LinearRegression object.

8.2 rowsToPlayerDataPoints()

Summary
This function takes rows that utilize the constant names and build an array of PlayerDataPoint Objects from them.

Definition
public function rowsToPlayerDataPoints($result, $linearRegression)

Parameters
result – This variable represents the data that is being searched for in the linear regression, rows will be compared to these and added if they are a match.
linearRegression- This is the linear regression object and represents a snapshot of what the system is looking for.

Return Value
This function returns an array of PlayerDataPoint objects

9 PlayerDataPoint.php

PlayerDataPoint.php contains the definition for the PlayerDataPoint class

9.1 Variables

Variables for this class are set to private

9.1.1 playerId

The variable playerId holds the playerId that will be obtained from the database.

9.1.2 firstName

The variable firstName holds the first name of the player matching the playerId obtained from the database.

9.1.3 lastName

The variable lastName holds the last name of the player matching the playerId obtained from the database.

9.1.4 x

The variable x holds the x value for the player

9.1.5 y

The variable y holds the y value for the player

9.1.6 expectedX

The variable expectedX holds the expected X value for the player in the database

9.1.7 expectedY

The variable expectedY holds the expected Y value for the player in the database

9.2 Class Functions

9.2.1 getPlayerId()

Summary

This is a getter function to return the playerID in the object.

Definition

public function getPlayerId()

Parameters

None

Return Value

This function returns the playerId from the object.

9.2.2 getFirstName()

Summary

This is a getter function to return the firstName variable in the object.

Definition
public function getFirstName()

Parameters
None

Return Value
This function returns the firstName variable from the object.

9.2.3 getLastName()

Summary
This is a getter function to return the lastName variable in the object.

Definition
public function getLastName()

Parameters
None

Return Value
This function returns the lastName variable from the object.

9.2.4 getXValue()

Summary
This is a getter function to return the x variable from the object.

Definition
public function getXValue()

Parameters
None

Return Value
This function returns the x variable from the object.

9.2.5 getYValue()

Summary
This is a getter function to return the y variable from the object.

Definition
public function getYValue()

Parameters
None

Return Value
This function returns the y variable from the object.

9.2.6 getExpectedX()

Summary
This is a getter function to return the x variable from the object.

Definition
public function getXExpectedX()

Parameters
None

Return Value
This function returns the expectedX variable from the object.

9.2.7 getExpectedY()

Summary
This is a getter function to return the y variable from the object.

Definition
public function getExpectedY()

Parameters
None

Return Value
This function returns the expectedY variable from the object.

9.2.8 getDifferenceX()

Summary
This is a getter function to return the difference between the x variable and the expected x variable from the object.

Definition
public function getDifferenceX()

Parameters
None

Return Value
This function returns the difference between the x variable and the expectedX variable.

9.2.9 getDifferenceY()

Summary
This is a getter function to return the difference between the y variable and the expectedY variable from the object.

Definition
public function getDifferenceY()

Parameters
None

Return Value
This function returns the difference between the y variable and the expectedY variable.

9.2.10 setPlayerId()

Summary
This is a setter function for the playerId variable

Definition
public function setPlayerId($id)

Parameters
id – This variable holds the id of the player to be set in the function.

Return Value
None

9.2.11 setFirstName()

Summary
This is a setter function for the firstName variable in the object.

Definition
public function setFirstName($firstName)

Parameters
firstName – This variable holds the first name of the player that will be set within the function

Return Value
None

9.2.12 setLastName()

Summary
This is a setter function for the lastName variable in the object.

Definition
public function setLastName($lastName)

Parameters
lastName –This variable holds the last name of the player that will be set within the function.

Return Value
None

9.2.13 setXValue()

Summary
This is a setter function for the x variable in the object.

Definition
public function setXValue($x)

Parameters
x – This variable holds the value that the x variable in the object is to be set to.

Return Value
None

9.2.14 setYValue()

Summary
This is a setter function for the y variable in the object.

Definition
public function setYValue($y)

Parameters
y – This variable holds the value that the y value in the object will be set to

Return Value
None

9.2.15 setExpectedX()

Summary
This is a setter function for the expectedX variable in the object.

Definition
public function setXExpectedX($expectedX)

Parameters
expectedX – This variable holds the external value that the expectedX variable in the object will be set to.

Return Value
None

9.2.16 setExpectedY()

Summary
This is a setter function for the expectedY variable in the object.

Definition
public function setExpectedY($expectedY)

Parameters

expectedY – This variable holds the external value that the expectedY variable in the object will be set to.

Return Value

None

10 ConnectionUtils.php

ConnectionUtils.php contains the function getConnection, which is responsible for database connection resolution.

10.1 Variables

The variables are contained within the function and are therefore private by scope.

10.1.1 servername

The variable servername holds the address for the MySQL server that will be connected to. In this case it is set to localhost as it will be local to the application.

10.1.2 username

The variable username holds the login id for the MySQL server that will be connected to.

10.1.3 password

The variable password is used to hold the password that matches the login id for the MySQL server that we are connecting to.

10.1.4 dbName

The variable dbName holds the name of the database that we are connecting to on the MySQL server.

10.2 Functions

10.2.1 getConnection()

Summary

This function attempts to connect to the MySQL server with the credentials provided in its internal variables and if successful returns a connection, otherwise it returns an error.

Definition

public function getConnection()

Parameters

none

Return Value

A MySQL connection if successful, a connect_error if failed.

11 login.php

LoginAction.php is called when the user tries to login to their account. The first action is to determine if a database connection has been made, if not a connection is made.

11.1 Variables

The variables are contained within the function and are therefore private by scope.

11.1.1 dbc

The variable dbc holds the database connection returned from calling the getConnection method in ConnectionUtils.php This variable is a local variable to LoginAction.php.

11.1.2 username

The variable username holds the login id for the website and is captured from the POST method. This variable is a local variable to LoginAction.php.

11.1.3 password

The variable password is used to hold the password for the website and is captured from the POST method. This variable is a local variable to LoginAction.php.

11.1.4 sql

The variable sql is internal to the login function described below and its scope will be private by default. This variable is used to hold the sql statement for querying the database for the username and password combination for the user login.

11.1.5 result

The variable result is internal to the login function described below and its scope will be private by default. This variable is used to hold the result of querying the database for the username and password combination for the user login. Based on the result variable the function will establish the validity of the login.

11.2 Functions

11.2.1 login()

Summary

This function will take the variables passed to it and check for a valid login by the user onto the website.

Definition

public function login($username, $password, $dbc)

Parameters

username – This variable is passed in and is the username that will be used to determine user access.
password – This variable is passed in and is the password that will be used to determine user access.
dbc – This is the database connection that will be used to access the database and determine user credentials.

Return Value
True – The user credentials have been validated.
False – The user credentials are not found. (Incorrect username or password)

12 logout.php
logout.php is called when the user logs out of the system. The logout ends the session and sends the client back to the index screen

12.1 Variables

12.1.1 _SESSION
The _SESSION is set to an empty array, thus clearing its information. The session is then destroyed.

13 myAccount.php
myAccount.php is a utility page used to allow the user to change their password.

13.1 Variables

13.1.1 dbc
The variable dbc holds the database connection returned from calling the getConnection method in ConnectionUtils.php

13.1.2 username
The variable username holds the login id for the website and is captured from the _SESSION array.

13.1.3 oldPw
The variable oldPw is used to hold the old password for the website and is captured from the POST method.

13.1.4 newPw
The variable newPw is used to hold the new user password for the website and is captured from the POST method.

13.1.5 message
The variable message is used to hold a feedback message to the user based on the input of the oldPw, newPw and username variables and their subsequent validation or invalidation.

13.2 Functions

13.2.1 validatePassword()

Summary
This function will take the variables passed to it and check for a valid login by the user onto the website.

Definition
function validatePassowrd($username, $password, $dbc)

Parameters
username – This variable is passed in and is the username that will be checked
password – This variable is passed in and is the password that will be checked
dbc – This is the database connection that will be used to access the database and determine user credentials.

Return Value
True – The user credentials have been validated.
False – The user credentials are not found. (Incorrect username or password)

13.2.2 updatePassword()

Summary
This function will take the variables passed to it and update the user password in the database with the password variable.

Definition
function updatePassword($username, $password, $dbc)

Parameters
username – This variable is passed in and is the username that the associated password will be changed for
password – This variable is passed in and is the value that the user password will change to
dbc – This is the database connection that will be used to change the user's password

Return Value
This function returns the result of the UPDATE query used to change the users password.

14 results.php
Results.php provide up to the top 50 most cost effective players using the criteria from the shopBatting screen. It does this by creating a linear regression object, fetching the data, sorting the data, and then printing the data.

14.1 Variables

14.1.1 conn
The variable conn holds the returned connection object from the getConnection() function.

14.1.2 stat
The variable stat holds the stat option selected on the shopBatting screen. It is received from the _POST variable. The choices are Hits, Home Runs, and Doubles.

14.1.3 minAB
The variable minAB holds the minimum at bats selected from the shopBatting screen. It is received from the _POST variable. The choices are 0, 162, and 502.

14.1.4 minBirthYear
The variable minBirthYear holds the minimum birth year selected from the shopBatting screen. It is

received from the _POST variable. The choices are years from 1970 to 1990.

14.1.5 test

The variable test is an instance of a LinearRegressionCriteria object and will hold all of the necessary information for the linear regression including joins and comparisons.

14.1.6 sql

The variable sql is uses to hold the sql statement from the sql criteria derived from the getLinearRegressionSql function.

14.1.7 result

The variable result will be the result of the query that was performed using the sql statement in the sql variable.

14.1.8 dataPoints

The variable dataPoints holds the player data points returned from the function rowToPlayerDataPoints.

15 secureCheck.php

The secureCheck file is a utility file which checks to see if the _SESSION array has been set and if it is not then it redirects the user back to the index page with a message.

16 shopBatting.php

The shopBatting page provides a user interface to choose the statistics and values that they want when utilizing the website. This is the primary user interface.

16.1 Variables

16.1.1 stat

The variable stat holds the stat option selected on the shopBatting screen. It is received from the _POST variable if the page was already submitted. This is so that the user can see the values that they selected when viewing the results. Otherwise it gets its value from the sbStatOptions drop down box. The choices are Hits, Home Runs, and Doubles.

16.1.2 minAB

The variable minAB holds the minimum at bats selected from the shopBatting screen. It is received from the _POST variable if the form was already submitted. This is so that the user can see the values that they selected when viewing the results. Otherwise it gets its value from the sbMinAB drop down box. The choices are 0, 162, and 502.

16.1.3 minBirthYear

The variable minBirthYear holds the minimum birth year selected from the shopBatting screen. It is received from the _POST variable if the form was submitted. This is so that the user can see the values that they selected when viewing the results. Otherwise it gets its value from the sbMinBirthYear drop down box. The choices are years from 1970 to 1990.

16.2 Functions

16.2.1 getSelected()

Summary

This function determines which item is selected based on a comparison variable. It is used to put the previously selected item into the drop down boxes for user reference.

Definition

function getSelected($selected, $comparison)

Parameters

selected – This variable holds the variable taken from the _POST variable. This is value will be compared to what is in the comparison variable.

comparison – This variable holds the value present in the drop down box when passed in and is used to compare to the selected.

Return Value

Returns selected, or the original value if comparison is equal to selected.

17 index.php

The index.php page is the first interface page for the user on the website. It contains a brief overview of the website, and has various options on the left side of the page.

17.1 Variables

17.1.2 message

The variable message holds a message that needs to be presented to the user on the webpage.

18 Test Documentation

Capstone Testing Documentation					
Test Case #	Requirement Tested	Rationale	Input	Expected Output	Passed
1	3.1	Ensure correct data for each table	Run sql statements to verify only 2014 data and each record has general data in the master table	No records for years other than 2014 and no records missing a master record	YES
2	3.2	Ensure Users are able to log in the web application	Attempt to log in a username/password combination that matches a record in the database	The users is logged in and sent to the home page	YES
3	3.2	Ensure Users are not able to log in with incorrect information	Attempt to log in with an incorrect username and/or password	The user is not logged in and is given a message letting them know they had an incorrect username or password	YES
4	3.2	Verify that the security module is working as intended	Ensure that users who are not logged in are only able to view screens that are available on their side navigation	Redirect to the overview screen when a user manually enters a url to a screen not available	YES
5	3.2	Ensure that the password changing works correctly	Attempt to change password using correct old password	Password change is reflected in the database	YES
6	3.2	Ensure that the password changing works correctly	Attempt to change password using incorrect old password	User is notified that the old password is incorrect and the password remains the same	YES
7	3.3.1, 3.3.2	Ensure SQL accuracy in LinearRegressionCriteria Module	Ensure that SQL statements generated from search parameters in LinearRegressionCriteria match the expected sql	SQL statements that generate expected sql statements	YES
8	3.3.1, 3.3.2	Ensure LinearRegression Object calculations are correct	Use a simplified database with a small set of numbers that can be easily verified to generate a Linear Regression	Accurate Linear Regression Data including Standard Deviation, Variance, & Correlation Coefficient	YES
9	3.4	Ensure that Dropdown menus are working correctly	Verify that the expected search criteria is reflected in the LinearRegressionCriteria	Correct search criteria added to LinearRegressionCriteria	YES
10	3.4.3	Make sure that players are being sorted correctly	Test using a smaller database to verify sorting is correctly based on difference in salary and expected salary	Correctly sorted player data points	YES

19 Known Bugs and Issues

At the time of release, no known bugs or issues are reportable.