**Western - Governors - University**



**National Hockey League**

**Player Evaluator**

An investigation of National Hockey League player

statistics and their impact on the league

Dakota Styck

2023

# Table of Contents

[**Table of Contents 2**](#_y4jzi64kraey)

[**Letter of Transmittal 3**](#_8yah5lchq0ua)

[**A - Project Recommendation 4**](#_t857mww6f9kt)

[A.1 - Problem Summary 4](#_nxufigr7o7lp)

[A.2 - Application Benefits 4](#_lifuz2f7yzkp)

[A.3 - Application Description 5](#_po17j5kkvy5m)

[A.4 - Data Description 5](#_430lr9y4855x)

[A.5 - Objectives and Hypothesis 5](#_2h6e4vtaspu4)

[A.6 - Methodology 6](#_re1zapitvxii)

[A.7 - Funding Requirements 6](#_18drr04b0tpo)

[A.8 - Data Precautions 7](#_razjfpx3nf0p)

[A.9 - Developer’s Expertise 7](#_row657514l4h)

[**B - Project Proposal 8**](#_xvpwkawbmxq5)

[B.1 - Problem Statement 8](#_5z3rzf9ur9f1)

[B.2 - Customer Summary 8](#_9a8xy089rhaz)

[B.3 - Existing System Analysis 8](#_829f3pip51wi)

[B.4 - Data 9](#_fua9fcxh6l43)

[B.5 - Project Methodology 10](#_22kxhjj1wyj)

[B.6 - Project Outcomes 10](#_w0k524chiq1i)

[B.7 - Implementation Plan 11](#_9mqgpb5toocw)

[B.8 - Evaluation Plan 11](#_tjoxu6arbl1s)

[B.9 - Resources and Costs 13](#_m4ec2f9oqqr4)

[B.10 - Timeline and Milestones 13](#_4256i9xzt70q)

[**C - Application 15**](#_b6g7xi12ojgk)

[**D - Post-Implementation Report 16**](#_jzjwv46khmc7)

[D.1 - A Business Vision 16](#_ehn57b9f2tkh)

[D.2 - Datasets 16](#_kn3tqhcx9l3y)

[D.4 - Data Product Code 19](#_s42705z3uv3y)

[D.5 - Objective Verification 21](#_6q8ir5p1ik19)

[D.6 - Effective Visualization and Reporting 21](#_el9tlnifbv7x)

[D.7 - Accuracy Analysis 23](#_tfmp5bm8nhb8)

[D.8 - Application Testing 23](#_n6yyuhx8vcg)

[D.9 - Application Files 24](#_5l72hn6rnp58)

[D.10 - User Guide 24](#_54hrj4zshaf)

[D.11 - Summation of Learning Experience 25](#_4zqz93wvkozd)

# Letter of Transmittal

Dakota Styck

Computer Science Undergraduate

123 Main Street

Somewhere, IL 55555

February 24, 2023

Kyle Davidson

General Manager

Chicago Blackhawks

1901 West Madison Street

Chicago, IL 60612

Dear Mr. Davidson,

As a GM in the NHL, you know the importance of scouting. What if there was a program to make your scouts more efficient? My PC application will allow you to do just that. It will use player statistics from previous seasons to determine their value in the league. Not only that, but it will also predict the player’s value for next season. This app will increase your scouts’ effectiveness by showing them who is worth scouting at a glance.

The estimated cost for implementing this project will be $100,200. The first year’s expense for database management will be included in the implementation cost, but there will be an annual fee of $80,000 to maintain the system. Post-release there are also plans for additional features. Each one of those features will require a one-time fee of $20,000 to design and implement. The remaining $200 will go towards copyright fees.

Sincerely,

Dakota Styck

# A - Project Recommendation

## A.1 - Problem Summary

The proposed project will display player statistics (stats) for previous seasons on its dashboard. It will use those stats to predict each player’s value. It will also make projections for how each player will perform in the upcoming season. The initial release is intended for use during the off-season. However, a future release will be able to predict a player’s rest-of-season performance based on their previous stats and how they are performing for the season in progress.

This project will increase the efficiency of scouts in the National Hockey League (NHL). There are over 600 players in the NHL. The proposed system would narrow down which of those are worth scouting. It will benefit the organization in multiple ways. First, it will reduce the amount of money needed for employee scouts. Second, it will help the general manager (GM) determine the value of players in the league. Using this application, they will see if a player is worth their asking price for the contract, trade, and free agency decisions.

## A.2 - Application Benefits

Player scouting has always been a significant asset to NHL organizations. This program will vastly improve that aspect for any team using it. The first way it helps is by reducing the money needed for scouts. The second way is by allowing GMs to see a player’s value at a quick glance.

Scouts will no longer need to waste time figuring out which players are worth scouting in depth. This application will determine what players are worth scouting and which aren’t. More efficient scouts mean less will be required to get the same results. Fewer scouts equal more money for the organization to spend on other aspects.

This application will also give GMs the ability to determine player values at a quick glance. It will help determine possible gems buried in deep rosters. Knowing a player’s value will also help determine if they are someone the GM should acquire (or release). It will be beneficial when making contract decisions.

## A.3 - Application Description

The application’s dashboard will contain an interactive table that contains player stats for the previous season. An estimated player value (EPV) will also be attached to their stats. A player’s EPV will be calculated using an algorithm I design. It will determine the player’s overall value based on not only their stats but also the stats of everyone else for that season. The table will have the ability to show stat lines for different seasons. Filters allow the user to search for players of a specific position.

The dashboard will also contain two different KPIs for the user. One will be a bar graph of the top 5 skaters for the table’s current season/filter combination. The other will be a line graph that shows a team’s average player value over the past few seasons.

There will also be a second scene the user can access that contains the same interactive table and bar graph from the dashboard. However, this scene will be for future predictions. The table will contain predictions for every stat category used in the application and a projected EPV.

## A.4 - Data Description

The proposed project will contain NHL player statistics from the previous three seasons. The raw data for it will be obtained from Hockey-Reference.com. Each data entry will consist of a skater's name, a unique ID, their stats for a specific season, and an EPV. The player stats included in the application will be goals, assists, plus-minus, penalty minutes, time on ice (in minutes), hits, blocks, and faceoff wins/losses. To reduce the risk of outliers the application will only give players with 20 or more games played in the given season an EPV.

The statistics attached to each player/season combination will be quantitative. The raw data will contain nothing but independent variables. Dependent variables will be added afterward, including points, faceoff percentage, estimated player value, and stat predictions for the upcoming season.

## A.5 - Objectives and Hypothesis

The desired outcome for this project is to give scouts and GMs in the NHL the ability to quickly view player statistics and estimated values in a clean, easy-to-use, graphical user interface (GUI). The desired accuracy for projected statistics is 90%.

## A.6 - Methodology

An agile methodology will be used to develop the proposed project. It’s important to use this type of methodology when working on a Java application. It is hard to predict how the program’s functions will work together ahead of time. Using an agile methodology allows for easier corrections as the project progresses.

There are multiple phases in the agile methodology: plan, design, develop, test, deploy, review, and launch. Since there will be multiple functionalities and scenes, it is best to focus on a small section of the program for each sprint. For example, the first sprint will focus on designing a login scene. Requirements will be gathered for the scene from stakeholders. The project manager will then use those requirements to design it. After the scene has been designed, coders will work on developing a functional version of it. Unit testing will be done during the development process. At the end of the sprint, if the scene works as intended, the next section of the project can start. If it doesn’t work, the sprint will repeat until desired results are achieved.

## A.7 - Funding Requirements

The overall funding requirement for the initial release of this proposed project will be $100,200. It will cost $80,000 each year to hire a database manager to keep the information relevant to the end users. The project will also require a software engineer, costing $5,000 a month. The project will take four months to complete, so that’s $20,000 in total for the software engineer. There are also additional features planned for future releases. Each one will need a software engineer with the same monthly wag. Getting a copyright for the software shouldn’t cost more than $200. The length of a copyright is the author's life plus 70 years. That is not something that will need to be renewed any time soon. None of the tools needed for the initial release cost any money. However, future releases may require a fee if a database server is needed to keep up with the expanding data. Statistics will be acquired for free from Hockey-Database.com. All they require to use the data is a mention of where it comes from.

## A.8 - Data Precautions

The only sensitive information will be user data. A username and password will be required to log in to the program. Passwords will be protected by using an SHA algorithm to hash them. Player statistics are public data and do not need to be protected in any way.

## A.9 - Developer’s Expertise

I am more than qualified to oversee this project. I’m a computer science undergraduate and only have two classes remaining before I can get my bachelor's degree. I’ve worked with Java for over four years and have created multiple projects with a GUI. I have an ITIL4 certification for project management. I’ve also been a die-hard Blackhawks fan for over a decade and have spent a lot of time reviewing player statistics to manage my fantasy hockey teams, which I tend to do well in.

# 

# B - Project Proposal

## B.1 - Problem Statement

There are over 600 players in the National Hockey League (NHL). Being a general manager (GM) of a team in the NHL is an overwhelming task. One major aspect of their difficult job is player evaluation. They are constantly making decisions on the best makeup for their team’s roster. Organizations spend millions of dollars each year on their scouting department to handle this issue. If there was an application available that helped make player evaluations easier it would save the organization money and improve its chances of success.

## B.2 - Customer Summary

The intended customers for the proposed product are scouts and GMs in the NHL. The product will feature a clean and easy-to-use user interface (UI) that makes it usable by anyone. This product aims to make their jobs easier and more efficient. It will be a great asset for any organization in the NHL.

This product will help GMs evaluate their team’s roster. They will be able to compare the value of their players to others in the league at a glance. The product will take into account the amount of ice time players had. This will allow it to find hidden gems on a roster where they might not be getting the playing time they deserve. This will also allow the program to be able to predict which players are getting more ice time than they deserve. It will be a valuable tool when making contract and trade decisions.

This product will also save scouts a lot of time. Thanks to the player values this program will assign, scouts will be able to quickly see who is worth further evaluation. This will save them from unnecessary traveling to scout a player who wouldn’t be a valuable asset for their team.

## B.3 - Existing System Analysis

Currently, there are many different systems available to the public that allow users to review NHL player statistics. However, none of these systems include predictive data. It can be hard to predict how a player will do by staring at a basic spreadsheet. That is why my program is needed in this industry. It will give users an easy way to compare player values.

## B.4 - Data

Raw data will be collected from Hockey-Reference.com. The program will use comma-separated value (CSV) files downloaded from this site. The website allows for adjusting what data is included in the downloaded CSV files. The proposed program will upload the data sets from these CSV files into a list of appropriate objects. The program can then easily use these objects to create tables of data, make predictions, perform calculations, and display other kinds of visualizations for the user.

The program will include a CSV file for each of the last three seasons. Before downloading each of these CSV files a few columns of data will be removed to minimize the size of the files. The columns that will not be downloaded to the CSV files include rank, age, team, points, point shares, even strength goals, power-play goals, short-handed goals, game-winning goals, even strength assists, power-play assists, shooting percentage, average time on ice, and faceoff percentage. Some of these columns are independent variables that can be calculated in the program, while others are just not necessary for the scope of this project. The remaining columns that will be downloaded in the CSV files include player name, position, games played, goals, assists, plus minus, penalty minutes, shots, time on ice, blocks, hits, faceoff wins, faceoff losses, and a unique player ID. Each row of data will have a unique season and player ID combination that the program can use to access it.

The other type of CSV files that will be downloaded from Hockey-Reference.com are team rosters for each of the last three seasons. Every team will have a CSV file for each season included in the program. The data used for this will be from February 1st of the given season. The columns to be removed before downloading these files include player number, flag, position, age, experience, summary, salary, and draft. The following columns will be used: player name, height, weight, shoots (handedness), birth date, and the player’s unique ID.

The columns that were removed from the data should prevent there from being any anomalies in the data. There will be outliers in the raw data, however, that won’t be handled until predictions are made. Players with less than twenty games played in a season will not be eligible for an estimated player value. The program will also use minimum/maximum stats to estimate values for every player in that season. Players without the minimum number of games played will not be included in these maximum/minimum numbers.

## B.5 - Project Methodology

The application will be designed using an agile methodology. The agile methodology goes through the following phases: plan, design, develop, and test. It then repeats these phases through project completion.

The program will consist of many features that will be designed one at a time. For example, the first thing that will be designed for the application is a login screen. In the planning phase, requirements will be gathered. In the design phase, the screen’s layout and functionalities will be created. This design will then be used to develop a functional java application that includes all of the requirements for the screen. After thorough testing, if the screen works as intended, planning for the next step will begin. However, if it doesn’t turn out as intended, the phases will repeat until the intended results are met.

## B.6 - Project Outcomes

This project will deliver a fully functional java application. This application will contain skater statistics for the previous three seasons. The application’s dashboard contains three interactive visualizations: a table of player statistics, a bar graph of the top five skaters, and a line graph showing a team’s average player value compared to the best and worst teams in the league. The application also has another page that can be accessed to view skater predictions for the upcoming season.

Upon launching the application the user will be directed to a login screen. If they don’t have an account they will have the option to do so. Once an account has been created and the user logs in they will be directed to the application’s dashboard. Here they can filter the information displayed to them by season and skater position. They can either use the menu bar or a button on the dashboard to get to a future predictions scene. The future prediction scene is similar to the dashboard and users will also be able to filter that information by skater position.

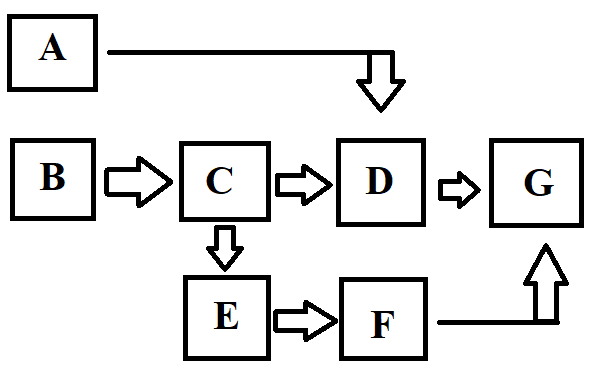
## B.7 - Implementation Plan

The general strategy to implement this project is to complete it in several small phases. Each phase includes rigorous testing throughout the design process.

Project Phases:

1. Create a login scene
2. Complete the database needed to store relevant data
3. Design and implement the EPV algorithm
4. Create the dashboard
5. Design an algorithm that can predict future performances
6. Create the future performance scene
7. Deployment

Dependency Diagram:



## B.8 - Evaluation Plan

* Login Scene
  + Users can create an account
  + Users can log in with their new account
  + User data is saved to a CSV file in the program and can be used upon restart.
  + Passwords are hashed using an SHA algorithm
* Database
  + CSV files are created and 100% accurate
  + The program can load this data into its system and access it via functions.
* EPV Algorithm
  + This algorithm will predict the player’s value for a given season.
  + The skater must have played at least 20 games
  + Stats are adjusted based on the player's time on the ice.
* Dashboard
  + Buttons the user can use to easily navigate between scenes in the program
  + A menu bar
    - Allows the user to edit/remove their account
    - Allows the user to log out
    - Gives the user the ability to navigate through scenes
  + Three visualizations are properly displayed
    - Player statistics table
      * Users can sort the table by any column
      * Displays accurate information
      * Has a functional filter for season/position combinations
    - Top five skaters bar chart
      * Includes the players with the top five EPVs for the given filters
      * Includes the ability to see a breakdown of each player’s stat ratings
      * Includes the ability to filter what data is shown
    - Relative team average EPV line chart
      * Includes the minimum and maximum average team EPV for the last three seasons.
      * The third variable for the chart is the user’s favorite team (selected during account creation)
      * The third variable can be edited by the user
* Future performance algorithm
  + This algorithm predicts the skater’s statistics for the upcoming season.
  + Based on statistics from the skater’s career
  + The skater’s previous season has a higher weight than their career average.
* Future performance scene
  + Includes the same menu bar as the dashboard
  + Two visualizations
    - Table of player’s predicted statistics.
      * Can be sorted by column
      * Can be filtered by position
    - Bar chart of top five skaters' projections
      * Sorted by EPV
      * Includes individual stat ratings
      * Ability to filter how much information is shown
* Deployment
  + 100% bug-free (known bugs must be fixed)
  + The project will be packaged into a zip file for distribution
  + The user must run the included JRE installer first
  + After the JRE installer has run, they can then launch the program
  + Any issues can be addressed by emailing dstyck@wgu.edu

## B.9 - Resources and Costs

Software Engineer - - - - - - - $5,000 / month

Database Manager - - - - - - - $80,000 / year

Copywrite Fee - - - - - - - - - - $200

Personal Computer - - - - - - - $400

## B.10 - Timeline and Milestones

Week 1 - Login Scene

Week 2 - Database

Week 3 - EPV Algorithm

Week 4 - Dashboard

Week 5 - Future Performance Algorithm

Week 6 - Future Performance Scene

Week 7 - Deployment

# 

# C - Application

Everything for this project is contained in the “Capstone v1.1.zip” folder. Inside the folder are resources files needed to run the project: this report, an “INSTALL ME FIRST (JRE).exe” file to set up your system environment so it can run the project, and a “RUN ME SECOND (Application)” file that will launch the program.

# 

# D - Post-Implementation Report

## D.1 - A Business Vision

There are over 600 players in the National Hockey League (NHL). Being a general manager (GM) of a team in the NHL is an overwhelming task. One major aspect of their difficult job is player evaluation. They are constantly making decisions on the best makeup for their team’s roster. Organizations spend millions of dollars each year on their scouting department to handle this issue. If there was an application available that helped make player evaluations easier it would save the organization money and improve its chances of success.

## D.2 - Datasets

All of the raw data used in this program was taken from Hockey-Reference.com. Their website allows the data to be downloaded in multiple ways, but CSV files were used for this project. The website also allows you to remove columns of data that you don’t want before downloading the CSV files. This required a fair amount of manual labor to download all of the required CSV files but was still drastically more efficient than manually inputting all of the data. For each season included in the program, a CSV file of player stats was required. For each season a CSV file for each team was also required. So for the 2021-2022 season, a total of 33 CSV files were needed since there were 32 teams in the league.

Below is an example of what the raw data for season statistics look like.

Before exporting this data as a CSV file I hid partial rows. These partial rows are used when a skater plays on multiple teams in a season. However, the only data needed from this set is the totals for the season. The following columns were also deemed unnecessary and were taken out before exportation: Age, Tm, PS, EV, PP, SH, GW, S%, ATOI, and FO%. A unique player ID is also included in the exported CSV file, but that information is not displayed in the table.

The image below depicts what the raw data looks like for each team’s roster.

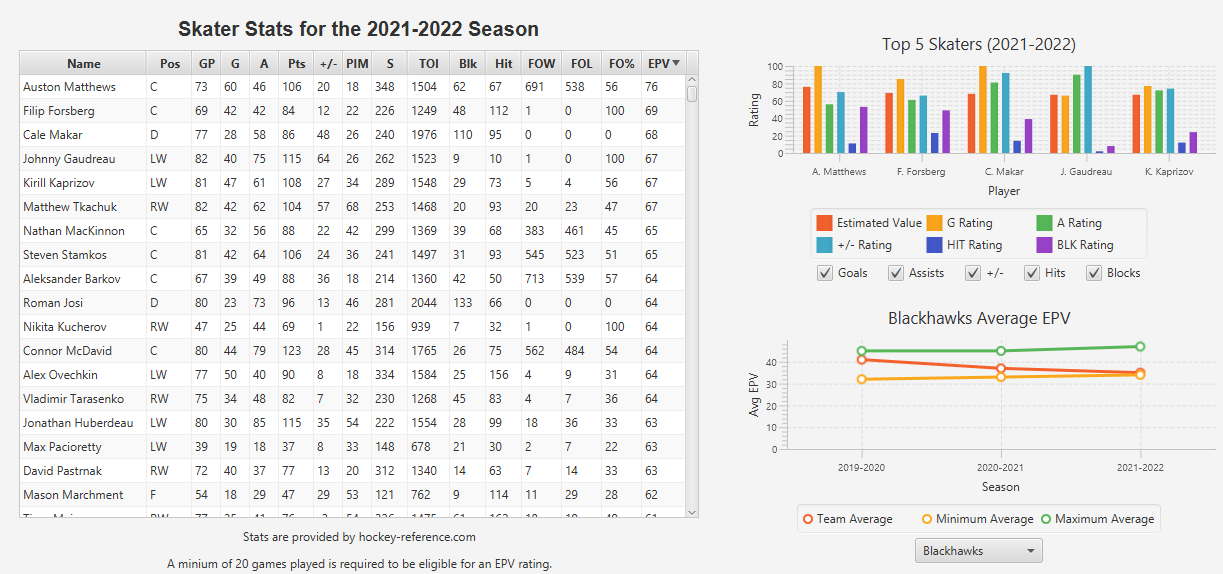
The following columns were deemed unnecessary and were removed before exporting the CSV file: No., Flag, Pos, Age, Exp, Summary, Salary, Draft. A unique player ID is also included for each row in the exported file. Not all of the information exported is used in the current version of the program, however, I left it in the CSV files because I thought they might be useful for future releases.

There’s also a users.csv and a teams.csv file used for the program. The teams.csv was created manually and contains every team currently in the NHL. Each row contains two items: the city and the team name. The users.csv file contains data for all of the users of the program. When a user creates an account their information is saved to this file. Each row contains three values: a username, their password (hashed), and their favorite team.

The exported CSV files can be found in the “src/capstone/csv” directory of the project’s folder.

## D.4 - Data Product Code

When the program is launched the first thing it does is load the CSV files and create objects out of them to be used in the program. The main method makes a call to MyCSV.java and uses a function called loadCSVs() to perform this action. There are 4 different types of CSV files that get loaded into the program: seasons, teams, users, and rosters. Each one has its own load function inside the MyCSV.java file which is then embedded inside the loadCSVs() function.

The following image below depicts both the descriptive and non-descriptive methods displayed on the program’s dashboard. 

The table displays the raw data retrieved from Hockey-Reference.com. Users have the ability to sort this data by any of the given columns. They can use filters to change what season and skater positions are displayed (the filter is not shown in the image).

The table above also contains a non-descriptive method. An estimated player value (EPV) was attached to each row of data. This EPV gives an overall rating to the player based on an algorithm I designed. This algorithm uses the skater's statistics as well as the league’s best value for each category for the given season. The statistics used to determine this value are a skater’s number of goals, assists, plus-minus, penalty minutes, shots, blocks, and hits. The player's time on ice and the number of games played also have an important role in determining the rating for each of the statistical categories. Since not all skaters get the same amount of ice time an adjustment was made to level the playing field and help find players who could potentially break out if given more ice time. Each stat was adjusted to determine what it would have been if the player averaged 20 minutes of ice time per game and played every game in the season. For example, if someone scored 10 goals but only played in 41 games (out of 82) at 10 minutes per night their adjusted goal value would be 40. This is calculated by taking their base 10 goals, multiplying it by 82/41 games, and then multiplying that by 20/10 minutes. After this adjustment is made a rating must then be given. This is calculated by taking the adjusted stat value, dividing it by the max stat value for the given season (which is also an adjusted value), multiplying that by 100 (the max value), and then rounding to the nearest whole number. Using the previous example, if the max goal value for the given season was 60, the skater’s goal rating would be 67/100 (40/60 \* 100). Once the rating for each category has been calculated they are then given a weight based on the importance of the category, added up, and divided by the total number of weights used. The weighting system is also dependent on the skater’s position; a defenseman’s hit category is worth more than that of a forward’s.

In the image of the dashboard, there are also two other non-descriptive visuals. The one on top is a bar chart that contains the top five skaters for the given season (based on their EPV). The stat ratings for multiple categories are also displayed to give the user an in-depth view of why their EPV is so high. Each category can also be removed by unchecking them using the boxes below the chart. This was done in case there is a certain category the user doesn’t care to see. When the table filters are applied it also updates this chart. The second visual is a line graph of a team’s average EPV over time compared to that of the best/worst team averages for each season. By default, the user’s favorite team is displayed, however, the dropdown box under it allows them to pick any other team.

Lastly, there is a whole other page the user can access that makes predictions as to what each skater will do next season. It uses the same table as the one displayed in the dashboard, but all of the data inside it is predictive. To calculate a skater’s predicted value for each category their historical data is used. For example, a player’s historical average goal value is 20 goals, but their goal value for the most recent season was 30. Their predicted value for the next season would be 27 goals. This is determined by taking their career average + 2 \* last year’s value and dividing that total by 3. I wanted to emphasize what the player has done more recently and that’s why the previous season is valued more than the career average. Similar to the dashboard the top 5 skaters are also displayed in a bar chart. The table and chart can be filtered by position.

All of the non-descriptive methods listed above provide the user with insights that are difficult to determine just by looking through a regular table of player statistics. It will greatly help scouts and general managers make business decisions. By giving them these insights they will have a better understanding of the value of the players in the league. GMs can use this information to help them make roster decisions, while scouts can use it to determine who is worth a more in-depth look.

The code for calculating EPV and stat ratings can be found in the Stats.java file located in the src/capstone/objects directory of the project’s folder.

## D.5 - Objective Verification

The objective of this project was to make some of the decision-making processes easier for both general managers and scouts in the National Hockey League. This program successfully does that by giving them easy access to predictive data. The program also displays plenty of raw data they can use to help them with the player personnel decisions. Any team in the league would benefit from using this product.

## D.6 - Effective Visualization and Reporting

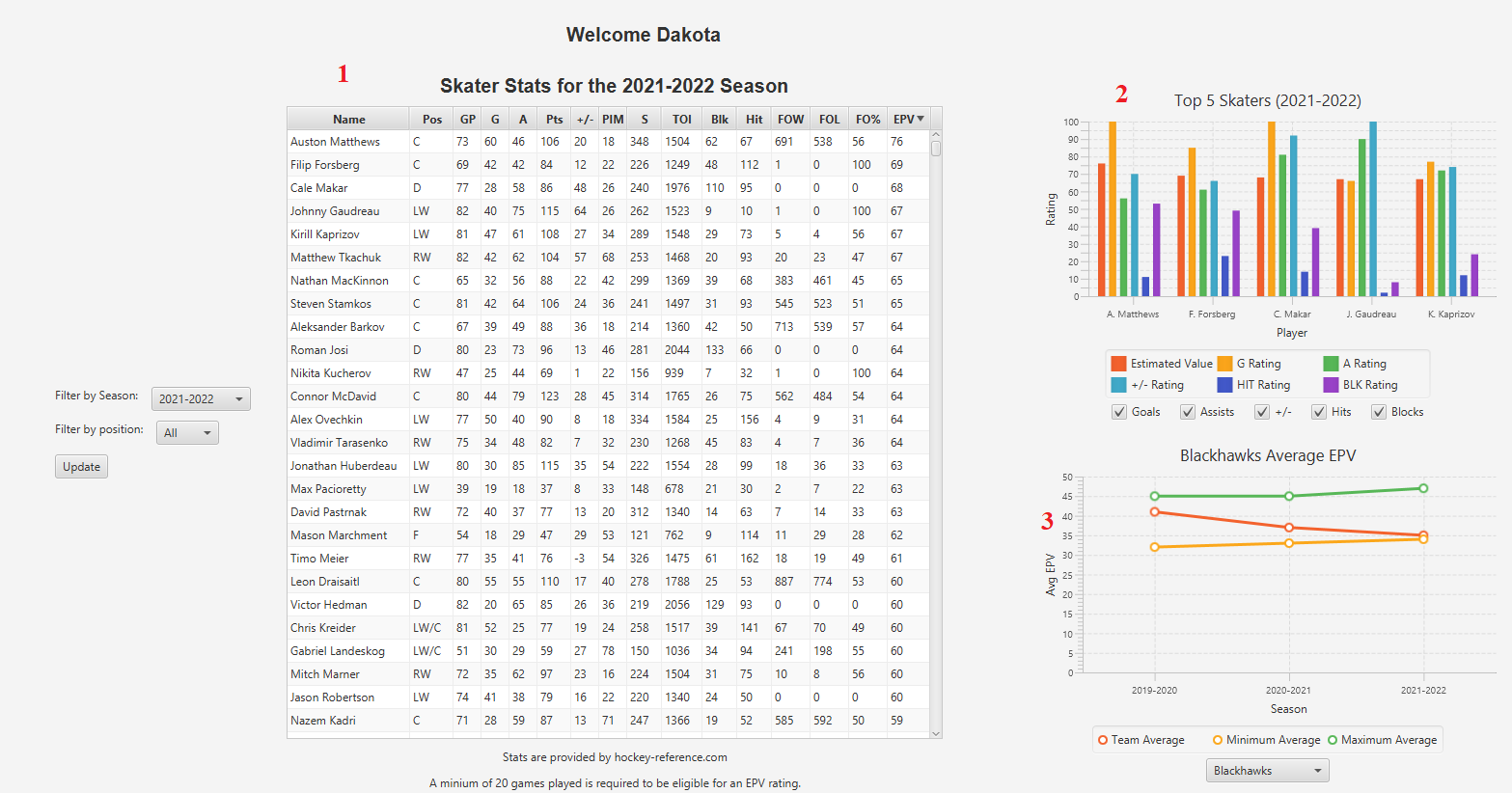
The non-descriptive methods used in this project could not exist without the descriptive methods used; they are completely dependent on the raw data. The non-descriptive methods took into account individual averages as well as league-wide statistics to determine a player's value. There is a table on the program’s dashboard that allows users to explore the raw data. This table allows them to sort the information by any of the given columns. It also has a filter that allows them to limit what type of players are shown as well as the ability to see archived data from previous seasons.

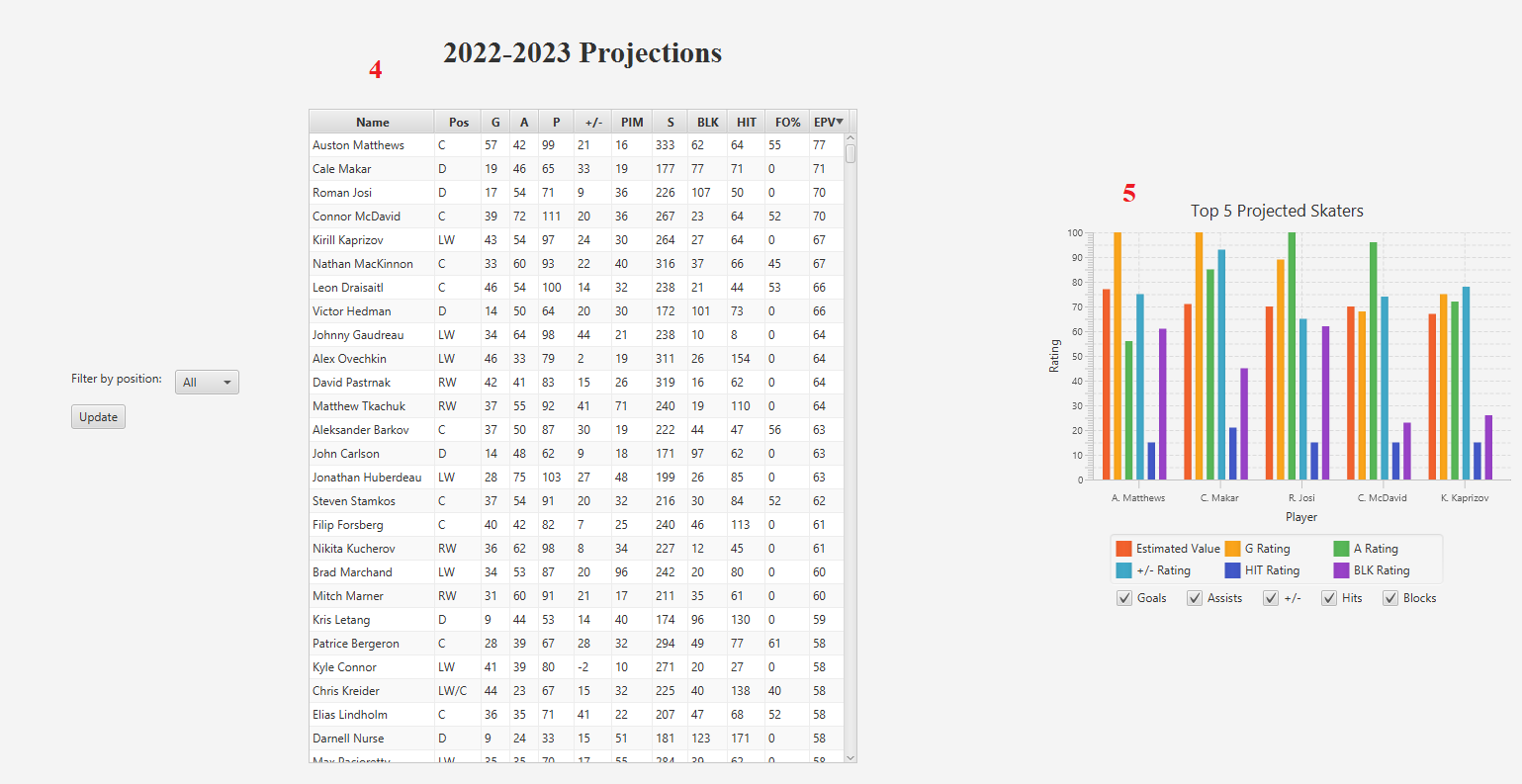
This raw data is then analyzed in multiple ways. Along with all of the raw data in the table, there is also a predictive column attached that is used to display a player’s overall value (their EPV). By default, the table is sorted by EPV so users will be able to quickly see the league’s most valuable players. There is also a bar chart on the dashboard that displays the top 5 EPVs from the table and then breaks that information down even further. It displays multiple ratings given to different statistical categories such as goals, assists, hits, and plus-minus. Furthermore, there is also a section of the program that allows users to see player predictions for the upcoming season. It uses a skater’s historical data to predict what their statistics will look like: goals, assists, hits, etc. This data is displayed in the same type of table as what appears on the dashboard and can also be sorted and filtered. There is also a bar chart with a detailed breakdown of the top 5 skaters (based on EPV) in the table.

While the EPV is a good summary of the statistical ratings there is one more visualization to give the user an overall understanding of how any given team compares to the best and worst team in the league. The program calculates the average EPV of rostered players for each team. It then displays the team's progression or regression over time in a line graph and compares it to the best and worst average team EPV for each season. By default the team selected for comparison is the user’s favorite team, however, they are able to choose any of the available teams for further comparison.

The images below depict the five different visualizations mentioned above. I put a red number next to each one for clarity; those numbers are not actually displayed in the program. The following is a quick summary of each visualization:

1. Skater statistics
2. Top five skaters based on EPV
3. Team average EPV over time
4. Project skater statistics for the upcoming season
5. Projected top five skaters for the upcoming season





## D.7 - Accuracy Analysis

Unfortunately there is no way of testing the accuracy until the upcoming season has finished. However, I’ve been following hockey closely for over a decade now, won multiple fantasy hockey leagues, and feel that the EPV and predictive values are around where they should be. To test the non-descriptive (predictive) method one would need to compare the end-of-season statistics to the predictions made. For example, if a player was predicted to score 40 goals but actually score 30, the data would be considered 75% accurate; 30 / 40 = 0.75.

## D.8 - Application Testing

There was a lot of unit testing throughout the application to make sure everything worked properly. One example of this would be after the first iteration of the login scene was complete I tested all kinds of different combinations to make sure that a user could log in as intended. The first iteration didn’t handle user creation properly; instead of adding to the users.csv file, it would overwrite the file. After doing some research I was able to make the appropriate adjustments. I also did some user acceptance testing (with me being the user). One example of this is after I finished building the dashboard I realized that the UI could use some improvement. One thing I added was the ability to see what table columns refer to on hover. This was made a non-invasive way of informing the user exactly what they are looking at. Most users are expected to be well aware of the terms such as PIM or TOI, but for the EPV column (a stat designed specifically for this program) it seemed necessary.

A large amount of time was spent testing the stat ratings and evaluating the accuracy of skaters’ EPVs. One thing I realized early on was that skaters who played under 20 games (roughly 25% of a normal season) should not be considered for an EPV value. These skaters were clear outliers in the data and needed to be removed. As mentioned previously, an EPV is determined by calculating the rating for each player’s stat, applying a weight to them, and then averaging everything out. The weighting system took a lot of tweaking before I felt the desired results were achieved. After I set the initial weight system up I would run the program and look through the table on the dashboard to see if there was anything blatantly wrong; such as a player I know isn’t that good being ranked over a better one. I would then go back into the code, make a minor adjustment, and run it again. I repeated this until I felt everything looked right. These changes drastically helped to improve the accuracy of the non-descriptive EPV data.

Lastly, there is a “Patch Notes.txt” file that can be found in the capstone folder. After finishing up the project I realized there were still some bugs. There were also some changes to the UI that I felt were needed to make the program more elegant. The patch notes file contains a list of everything updated from v1.0 to v1.1.

## D.9 - Application Files

To execute the program access the following folders/files in the order they are listed. An indent means the file/folder is located inside the preceding folder. There are other files/folders needed to execute the application, but the user does not need to do anything with them.

Capstone v1.1.zip

–> Capstone

–> INSTALL ME FIRST (JRE).exe

–> RUN ME SECOND (Application).jar

## D.10 - User Guide

1. Download the Capstone v1.1.zip folder
2. Open the Capstone folder
3. Double click “INSTALL ME FIRST (JRE).exe” (Only needs to be run if it’s your first time accessing the program)
4. Double click “RUN ME SECOND (Application).jar”
5. You can log in using the username “admin” and password “password” (case sensitive), but it’s recommended you click “New User” for a more immersive experience.
6. Once you’re at the dashboard you will be able to see three interactive visualizations. The table is initially sorted by EPV, but you can click on any column to change that. To see the full extent of this program’s capabilities spend some time adjusting filters. There are filters to the left of the table, under the bar chart, and under the line chart.
7. Feel free to explore the menu bar in the top left of the program to see more of the program’s functionalities such as editing your profile.
8. There are three big buttons on the left side of the dashboard, however only the “Skater Predictions” button is currently available. The “Compare Players” and "Team View" buttons are out of the scope of this project. They display a message that says "Coming Soon..." when you click on them. While these do not currently work they are intended to be implemented in a future release. Click on the "Skater Predictions" button to make your way to the last scene.
9. In the skater predictions scene you will see projected statistics for the upcoming season. The table and bar chart are both interactable in the same way that the dashboard visuals were.
10. Thank you for observing my project. Click on the X in the top right corner to end the program.

## D.11 - Summation of Learning Experience

I’ve worked on multiple java projects in the past that helped prepare me for this project. There were still some things I needed to look up to refresh my memory, but it was like riding a bike. Once I remembered I had no issue implementing whatever it was I needed to look up. There was only one small portion of code that I took from another website, but I referenced the link in the source code as well as in the resources.txt document located in the submitted capstone folder. This portion of the code allowed me to convert users' passwords from strings to hashed values. It was two separate functions that I rolled up into a single one for ease of use.

I love working on projects like this because it gives me hands-on experience with the type of things I would like to be doing post-graduation. It not only helped to reinforce concepts I already knew, but it also taught me new things. Working on a project of this magnitude helped me to understand the necessity of planning everything before jumping straight into coding. Without properly having an understanding of how everything was meant to work together beforehand I probably would have spent a lot of time re-doing my work. I’ve created a program that I’m proud to show off. I also made it easy for myself to go back in and add more functionalities by making everything object-oriented.