**TY. B. Tech.**

**CS 303: Software Engineering Laboratory**

**Player Value Analyser**

!!br0ken!!***Version 1.0***

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Python** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **DB, PHP** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **PHP/ Front End** |

**Approved By: Mahesh R. Dube**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Statement of Work | **3** |
| 2 | Feature Set | **11** |
| 3 | System Requirement Specification | **28** |
| 4 | Feasibility Study | **42** |
| 5 | Project Plan | **47** |
| 6 | Product Backlog | **55** |
| 7 | User Story Cards | **64** |
| 8 | System Configuration Management Plan | **84** |
| 9 | System Construction | **114** |
| 10 | Project Review | **122** |

**TY. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 1

**Player Value Analyser**

**Project Statement of Work**

***14-08-2017***

!!br0ken!!***Version 1.0***

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Python** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **DB, PHP** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **PHP/ Front End** |

**Approved By: Mahesh R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Title | **5** |
| 2 | Background | **5** |
| 3 | Objective | **5** |
| 4 | Definitions and Applicable Documents | **7** |
| 5 | Business and/or Technical Environment | **8** |
| 6 | Description and Scope of Work | **9** |
| 7 | Deliverables | **9** |
| 8 | Approach and Methodology | **10** |

# TITLE

**Player Value Analyser** is a tool for football teams to decide the value of players in the open transfer market.

* 1. This is a system that will predict a player’s value depending on his statistics, previous records and other dependable factors. The system’s database will consist of all the top footballers that are playing in the world and will provide a cutting-edge analysis of the player’s condition and value to a team.
  2. Football teams around the world are always in search of tools that can bring technology to the game and help them in decision making. A system such as this will be a very useful tool considering the recent state of bewilderment and unrealistic transfer fees in the global football market which lead to unsustainable situations for clubs.

# BACKGROUND

The competitive football industry is a multi-million-dollar industry. The clubs are worth billions of dollars and players are the clubs most valuable assets. Every season the clubs have a transfer season wherein they decide to buy and sell players from/to other clubs. The Player Value Analyser will prove to be an effective tool to give the club an idea about the player’s worth which will help them spend their money wisely.

* 1. The mandate of the football teams is to entertain people by providing them with the best quality of football while also keeping a good financial structure as to make profits and maintain sustainability.
  2. Football agents, directors and chairmen of football clubs, players and also new agencies can make use of this product. The service will be open to all while there may be a few premium features which can be accessed only upon paying for those services.
  3. The product will use web engineering process to project the output.
  4. The need for this service has been highlighted recently. The money that is getting into football needs to be properly regulated as to ensure that it is wisely spent. The football leagues are even thinking of making certain guidelines related to players values and this tool will be of great use in that case.
  5. The Football leagues in the world have certain clubs with each having an owner. These clubs buy players to improve their quality and they have to pay a price for such players.
  6. The organisation needs a third-party contractor to satisfy the requirement first of all to guarantee a fair and unbiased approach. This is a tool which means that it does not lie in the organisation’s domain i.e. Sports and entertainment.

# OBJECTIVE

The objective of this proposal is to provide a tool to generate fair value of a player. For calculating this fair value, the player’s attributes, his performances in earlier matches, the contract years he has on his current contract, and his overall gameplay will be considered. This will in-turn give a fair idea about how much the player will be worth to the club, and how much a club should spend on him in ideal situations.

* 1. As a part of this proposal, we hope to achieve development of a system that predicts fair player value given his attributes and other data, so as to help make proper decisions for the club.
  2. The system shall be open source and can be used by multiple users wherein there will be certain premium features available to only licensed users.
  3. In the process of building this system, no external contractors will be involved.

# DEFINITIONS AND APPLICABLE DOCUMENTS

**Transfer:**

In professional football, a transfer is the action taken whenever a player under contract moves between clubs. It refers to the transferring of a player's registration from one association football club to another. In general, the players can only be transferred during a transfer window and according to the rules set by a governing body. Usually some sort of compensation is paid for the player's rights, which is known as a transfer fee. When a player moves from one club to another, their old contract is terminated and they negotiate a new one with the club they are moving to, unlike in American, Canadian and Australian sports, where teams essentially trade existing player contracts.

**Transfer market:**

The transfer market is the arena in which football players are available for transfer to clubs. The transfer market consists of a transfer list consisting of players available for transfer, and also the money moving between clubs. For example, a club may be described as having "money to spend on the transfer market" or the market may be described in similar ways to the stock market.

**Transfer Request:**

A player may make a "transfer request," to leave their club before the end of their contract. In this case, the player is publicly stating his desire to move, and encouraging other clubs to make an offer for him. Due to the public nature of transfer requests, they are often being used by players to air their grievances, such as frustration over contract negotiations or a clash of personality with the manager.

**Regression:**

In statistical modelling, regression analysis is a statistical process for estimating the relationships among variables. It includes many techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed. Most commonly, regression analysis estimates the conditional expectation of the dependent variable given the independent variables – that is, the average value of the dependent variable when the independent variables are fixed.

**Documents:**

1. <http://www.remiqz.com/blog/prediction-transfer-values/>
2. <https://www.pro-football-reference.com/blog/index37a8.html>
3. <http://football-data.org/>

# BUSINESS AND TECHNICAL ENVIRONMENT

The System requires the following Business and Technical Environment to successfully commence in the stipulated time and resources.

* 1. The hours of operation will be independent as that of the organisation with weekly feedback given during the reporting time.
     1. The team will work Monday to Friday, 2 hours per day.
* This time will be utilised to work on completing the project documentation which will take up a major role in the initial weeks of the project.
* Later weeks will have more time invested in project planning and implementation with the documents having a lighter format.
* Time will be evenly utilised for Documentation, Planning , Execution, Testing & Debugging.
  + 1. Further work can be completed on weekends depending on the team/member’s convenience.
  1. The system will be delivered in two formats either as a web engineered product or as a downloadable computer system. In each case the system should not utilise a very high specification which will make the system difficult to run on certain systems.
     1. Internet connection will be required for the smooth functioning of the system. (in case of web system)
     2. RAM greater than 8GB (Minimum Requirement)
     3. Libraries with open source licenses will be used.

# 

# 

# 

# DESCRIPTION AND SCOPE OF WORK

The work that is to be done under **‘Player Value Analyser’** involves multiple steps:

1. Acquiring statistical data about players from public data sources.
2. Data Cleaning and formatting according to needs.
3. Data visualization in order to understand the domain and be able to use domain specific knowledge.
4. Feature engineering so as to create useful features from the data.
5. Regression analysis.
6. Developing a web-based front-end so as to ease the usage.

**‘Player Value Analyser’** will be made to only provide fair value of the player considering the player’s performance. It will provide a good estimate of the talent the player will bring to the club in terms of money. It will not consider any personal relations of the player with the clubs and also will not consider other abstract notions about the player and internal club issues. As such, the analyser can’t be expected to predict the exact transfer value for a player, especially for high end transfers that involve other features that aren’t going to be considered here.

The organisation reserves the right to amend the Scope of Work as the situation permits depending on the feasibility and limitations of the scope.

# DELIVERABLES

The system is in the initial stage of development and some of the deliverables may vary as the system continues to develop into a product. Amongst the contract deliverables are the core concepts of the project which will not change in any case. The system will stay true to its vision and the only changes may be seen are the ones in the User Interface.

These are some of the deliverables that team can outlie at this stage of development. Each stage has its own challenges and will be given apt importance by the contractor.

|  |  |
| --- | --- |
| No. | Details |
| 1 | Statement of Work |
| 2 | Feature Set |
| 3 | SRS Document |
| 4 | Feasibility Study and Project Plan using AGILE |
| 5 | Sprint level planning activity |
| 6 | Sprint Plan and Sprint Design |
| 7 | Software Configuration Management Plan (SCMP) |
| 8 | Sprint Execution |
| 9 | Sprint Review and Sign- offs |

# APPROACH AND METHODOLOGY

* 1. Preparing proper documentation and getting the views of the team and organisation by creating proper SOW, Feature Set Document and SRS Document.
  2. A Feasibility Study will be performed depending on the features discussed between the team and organisation and a Project Plan will be drawn up.
  3. The Project will follow the Agile model and all the necessary steps will be taken as per industry standards.
  4. A Sprint Execution will be carried out in phases to finish the project in the stipulated time, this will be done with the help of a Sprint Design and Plan.
  5. A Software Configuration Management Plan (SCMP) will be presented to ensure consistency of the product's performance, functional, and physical attributes with its requirements, design, and operational information throughout its life.
  6. At the end of each sprint, the team will have produced a coded, tested and usable piece of software.
  7. The System will be reviewed by the concerned organisation and all the issues will be presented to the team.
  8. Upon resolution of these issues a final and formal sign-off will be suggested.

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 2

**Player Value Analyser**

**Project Feature Set Description**

***29-08-2017***

!!br0ken!!***Version 1.0***

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By: Mahesh R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Project Vision | **13** |
| 2 | Project Mission | **13** |
| 3 | Project Scope | **14** |
| 4 | Goals | **15** |
| 5 | Feature Set | **26** |
| 6 | Stakeholders | **26** |
| 7 | Acceptance Criteria | **27** |

# 1. PROJECT VISION

To provide the estimate market value of the player to be sold/bought calculated from the players statistics over the last season so that there is no inflation in player prices in the transfer market.

# PROJECT MISSION

The Mission Statement summarises the aim of this project and what it is trying to achieve. This is our Project Mission:

1. The Player Value Analyser is a tool mainly for football (soccer) teams to predict the worth of a player depending on the statistical and other non-statistical data. The software will be presented to the public in the form of a website or a web-distributed application.
2. The Player Value Analyser will provide a figure for each player that a person using the application wishes to know of. The focus will be on pre-existing players in world of football but also to predict a new player’s worth when there is a sense of uncertainty with the valuation.
3. The software wishes to bring a sense of balance in the current football market to preserve the essence of the game. The team has undertaken this project as we are admirers of the football game and are keen to bring our technical viewpoint to the world of sports and entertainment.

# PROJECT SCOPE

**‘Player Value Analyser’** will be made to only provide fair value of the player considering the player’s performance. It will provide a good estimate of the talent the player will bring to the club. It will not consider any personal relations of the player with the clubs and also will not consider other abstract notions about the player and internal club issues. As such, the analyser can’t be expected to predict the exact transfer value for a player, especially for high end transfers that involve other features that aren’t going to be considered here.

These are our project goals as defined by the team:

1. Building Player Profiles.
2. Classifying Players into appropriate categories.
3. Extracting Player Requirements from the input database according to need.
4. Process the appropriate queries according to customer request.
5. Display Player Stats and other data used to analyse the value.
6. Rate the player and display his calculated value.
7. Determine the System’s accuracy and integrate the feedback.

# GOALS

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 1 | 1 | Build Player Profile |
| Target Audience | Customers |
| Driver | To make players profile |
| Description | Populating the whole database with player detail |
| Response | The goal is to extract data from web scraping and other sources. |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 1 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Collecting data of Players | To populate the whole database for future analyzing | The analyzing model will use this data | Software engineer’s machines | Web scaping and simple data entry or using prepared data set |

|  |  |  |
| --- | --- | --- |
| Goal 1 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| It will be a database of enough player details to make good predictions | It can be measured on the basis of how many numbers of player’s details are stored in the database | This is the initial step of database, so good amount of player’s details is the end point of this goal |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 1 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| This is the most major or building block for the whole system | Web Scaping and data Collection can be done in fair amount of time | To some extent, if the system demands 10thousands player’s details then this itself is a challenge | It is motivating because it is a main part of a system |

|  |  |  |
| --- | --- | --- |
| Goal 1 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| This forms the crucial part of the system and hence helps attain a problem that the organization faces | By using the sprint execution method, we can achieve this goal. | The resources we need should be served after the input stage. In terms of hardware requirements, team has enough resources |

|  |  |  |
| --- | --- | --- |
| Goal 1 Description: | | |
| TIME BOUND TEST | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| This is the initial step of database, so good amount of player’s details is the end point of this goal | The focus of this goal is clear: to populate the database. So, steps forward can be taken. | This is going to be base data for all new players. So this is the most important part of the system. |

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 2 | 2 | Classify Players |
| Target Audience | Customers |
| Driver | To distinguish players |
| Description | Make groups in the database according to player’s ability |
| Response | The goal is to divide the whole database according to classifications |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 2 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Classifying players | To make better predictions and better transfer window experience | The analyzing model will use this data | Software engineer’s machines | By using statistics tools will be divide the player’s database |

|  |  |  |
| --- | --- | --- |
| Goal 2 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| Make database distributed in parts according to the demand | It can be measured on the basis of how many classifications we do on the players | This goal will help in better predictions of future players |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 2 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| This goal is the answer of how we will distinguish prices of different classes of player | It’s realistic using some statistics tools we know | It is, as we have to experiment a lot of tools to classify data | It is motivating because  If we don’t implement this then we won’t get the quality of result we want |

|  |  |  |
| --- | --- | --- |
| Goal 2 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| This forms the crucial part of the system and hence helps attain a problem that the organization faces | By using the sprint execution method, we can achieve this goal. | The resources we need should be served after the input stage. In terms of hardware requirements, team has enough resources |

|  |  |  |
| --- | --- | --- |
| Goal 2 Description: | | |
| TIME BOUND TEST | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| If we can classify all the players in some categories then we have completed the goal | The focus of this goal to classify but we need to do lots of experiments | It’s really important because if we don’t do it then we can get garbage results for player’s value |

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 3 | 3 | Process Player Profiles |
| Target Audience | Customers |
| Driver | To get information from player profiles |
| Description | Statistical learning techniques will be applied to process and learn information from dataset of player profiles |
| Response | The goal is to extract usable information from the data. |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 3 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Applying statistical techniques. | To learn from the data to generalize on what attributes from the profile contribute to player value. | The user who wants to know the value of a player will utilize this. | It will be performed beforehand on development machines. | The player profiles collected in the input stage and some computer hardware to do processing. |

|  |  |  |
| --- | --- | --- |
| Goal 3 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| It will be a trained statistical model of players and their values. | It can be measured on the basis of how well it performs on different player profiles. | It would be an incremental process since any new player profiles should also be processed. |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 3 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| This will be one of the main goals to be achieved in this system. | Statistics can be very useful in such scenarios so it is not too far-fetched. | To some extent, but it is not too far-fetched as data about a player should have a relation to his value. | It is motivating because it is a main part of a system that aims to solve a real-world problem. |

|  |  |  |
| --- | --- | --- |
| Goal 3 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| This forms the crucial part of the system and hence helps attain a problem that the organization faces | By using the sprint execution method, we can achieve this goal. | The resources we need should be served after the input stage. In terms of hardware requirements, team has enough resources |

|  |  |  |
| --- | --- | --- |
| Goal 3 Description: | | |
| TIME BOUND Test | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| Although the goal is achieved once we freeze this process and produce a system, it should be used even when the system is in production. | The focus of this goal is clear: to produce a statistical model. So steps forward can be taken. | In the initial stages, it would be less important as input goals need to be cleared first, after that it would have medium to high priority. |

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 4 | 3 | Extract Player Requirements |
| Target Audience | Customers |
| Driver | To provide a way to handle user input. |
| Description | Here, name or some identification of player and/or a player profile will be accepted from user and will be processed |
| Response | The goal is to perform operations on a given profile in order to generate value for it. |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 4 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Handling user input | To predict a player’s value and present it. | The user who wants to know the value of a player will utilize this. | It will be performed on client/user machines. | The player profiles collected in the input stage and some computer hardware to do processing. Also processed statistical model will be required. |

|  |  |  |
| --- | --- | --- |
| Goal 4 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| It will be a prediction of values. | It can be measured on the basis of how well it performs on different player profiles. | This goal must be accomplished before production. |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 4 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| This will be one of the main goals to be achieved in this system. | Once the earlier goal is achieved, it is very realistic. | Its feasibility is dependent on how well the model functions. | It is motivating because it is a main part of a system that aims to solve a valid problem. |

|  |  |  |
| --- | --- | --- |
| Goal 4 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| This forms the crucial part of the system that creates functionality to handle input. | By using the sprint execution method, we can achieve this goal. | The resources we need should be served after the input stage and the previous processing stage. In terms of hardware requirements, team has enough resources |

|  |  |  |
| --- | --- | --- |
| Goal 4 Description: | | |
| TIME BOUND TEST | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| The goal will need to be achieved before the deliverable product is produced. | The focus of this goal is clear: to handle user input programmatically. So, steps forward can be taken. | In the initial stages, it would be less important as input goals need to be cleared first, after that it would have medium to high priority. |

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 5 | 1 | Demonstrate Player Statistics |
| Target Audience | Customers |
| Driver |  |
| Description | The statistics of the player will be displayed so as the teams have an idea of what the player is capable of. |
| Response |  |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 5 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Display Player Statistics | To give an idea about the players capability. | Customer who wants to know about the player. | It will be performed on client/user machines. | The player profiles collected in the input stage. |

|  |  |  |
| --- | --- | --- |
| Goal 5 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| The end result will be a collection of statistics accumulated over a period of time. | The figure of measurement is the accuracy of the stats. | The goal completion will go hand in hand with the project completion. |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 5 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| The goal is achievable because statistics are formed based on the performance of the player. | Yes, because it is based on the performance of the player. | The task has a lot of dependencies and will be tedious. | Yes, as teams need to know about the players. |

|  |  |  |
| --- | --- | --- |
| Goal 5 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| Yes, as the statistics are important to know about the on-field qualities of the player. | The timing will be a little short as the data has to observed over a period of time even after project completion. | Yes, as the statistics only have to be displayed. |

|  |  |  |
| --- | --- | --- |
| Goal 5 Description: | | |
| TIME BOUND Test | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| The goal is achieved when the statistics are displayed to the user. | The focus is clear: to provide customers the player statistics. | Yes. It is not very high on priority because it only has to display the statistics. |

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 6 | 1 | Estimate Player Value |
| Target Audience | Customers |
| Driver |  |
| Description | Depending on the statistics an estimate of the player value will be provided |
| Response |  |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 6 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Display Estimated player value | To give an idea about the player’s market value. | Teams who want to buy the player. | It will be performed on client/user machines. | The player profiles collected in the input stage. |

|  |  |  |
| --- | --- | --- |
| Goal 6 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| The end result will be fixed values and it will directly impact the success of the project. | The player value be estimated taking into consideration all the statistics. | The goal has a clear end point but the process will have to be constantly monitored and updated. |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 6 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| It is very much achievable as the statistics are available. | Yes, because it is based on the performance and statistics of the player. | The task plays an important role in the success of the project aim. | Yes, as teams need to know about the player’s market value. |

|  |  |  |
| --- | --- | --- |
| Goal 6 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| Yes, as the market value is important to know if the team can afford to buy the player. | The timing will be a little short as the data has to observed over a period of time even after project completion. | Yes , as it only has to display a value. |

|  |  |  |
| --- | --- | --- |
| Goal 6 Description: | | |
| TIME BOUND Test | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| The goal is achieved when the market value is displayed to the user. | The focus is clear: to provide customers an estimated value of the player based on the player’s statistics. | Yes. It is high on priority because it is necessary for the teams to know the value. |

|  |  |  |
| --- | --- | --- |
| Goal-ID | Priority | Factors Addressed |
| 7 | 2 | Assess Feedback |
| Target Audience | Customers |
| Driver | To make the model better using feedback |
| Description | Here, the player profile which has predetermined value can be used and we can see how well the model performs on it. |
| Response | The goal is to provide a way for feedback to affect the model for improvement. |
| Open Issues | Discussion and Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Goal 7 Description: | | | | |
| Specific Test | | | | |
| Is ‘What’ identifiable? | Is the ‘Why’ clear? | Can ‘Who’ be identified? | ‘Where’ will it be performed? | ‘Which’ resources are needed? |
| Feedback mechanism | To correct and cross-validate the model | It can be done by the engineers and the user as well, because any new input from user can also generate feedback. | It will be performed on development machines during development of the system and on client/user machines later. | The player profiles collected in the input stage and some computer hardware to do processing. Also processed statistical model will be required. |

|  |  |  |
| --- | --- | --- |
| Goal 7 Description: | | |
| Measurable Test | | |
| Is the end result quantifiable? | ‘Figure’ of Measurement | Has the goal a clear end date/point? |
| It is a qualitative result, as an improvement in predictions. It can be measured by comparing the earlier predictions with new predictions. | It can be measured on the basis of how well it performs on different player profiles after feedback. | This goal must be accomplished before production. |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal 7 Description: | | | |
| ATTAINABLE Test | | | |
| What is your reaction to goal? | Does it feel realistic? | Is it overwhelming? | Do you find it motivating? |
| This will be one of the goals to be achieved in this system to improve from what we have achieved. | Once the earlier goal is achieved, it is feasible. | It is plausible considering that we have a prediction and we can clearly create a way to feedback. | It is motivating because it is a main part of a system that aims to solve a valid problem. |

|  |  |  |
| --- | --- | --- |
| Goal 7 Description: | | |
| RELEVANT Test | | |
| Does it fit into the overall team / organization objective? | Taking overall fit is the timing appropriate? | Do you have sufficient resources / budget to succeed? |
| This forms the part of this system that handles feedback mechanism., which will be a helpful addition. | The timing will be a little short as the data has to observed over a period of time even after project completion. | The resources we need should be served after the input stage and the previous processing stage. In terms of hardware requirements, team has enough resources |

|  |  |  |
| --- | --- | --- |
| Goal 7 Description: | | |
| TIME BOUND Test | | |
| Does it have a clear end date/point? | Is the focus clear so you can create an action plan? | Is its position on an Urgency/Importance grid clear? |
| The goal will need to be achieved before the deliverable product is produced. | The focus of this goal is clear: to create a feedback mechanism. So steps forward can be taken. | In the initial stages, it would be less important as input goals need to be cleared first, after that it would have medium/high importance. |

# FEATURE SET

These are the features that make our product unique.

|  |  |
| --- | --- |
| Feature-ID | Feature Description |
| 1 | **Fair values that point to realistic player talent and worth.** |
| 2 | **Fast predictions of values, since model will be pre-trained** |
| 3 | **Stability because of using tested, peer-reviewed open-source libraries** |
| 4 | **Front-end that enables quick and useful operation** |
| 5 | **Portability since the proposed front end can be accessed on any platform.** |

# STAKEHOLDERS

|  |  |  |  |
| --- | --- | --- | --- |
| Stakeholder | Concerns | Quadrant | Strategy/ Benefits |
| Football Clubs | Ensuring proper handover of project to operations team | Minimal Effort | Communicate project specifications as required |
| Project Guide | Resource and scheduling constraints for production once project is transitioned to operations | Key Player | Solicit stakeholder as member of steering committee and obtain feedback on project planning. Frequent communication and addressing concerns are imperative |
| Concerned Department | Ensuring on time delivery of materials | Minimal Effort | Communicate project schedule and material requirements ahead of time to ensure delivery |
| General Mass User | Product performance must meet or exceed current product | Key Player | Communicate test results and performance specifications and obtain feedback on customer requirements or any changes. Provide frequent status reports and updates. |
| Design and Marketing teams | Concerns regarding resources to assist project team with product design | Keep Satisfied | Communicate applicable resource requirements early and ensure resources are released back to engineering when they’re no longer required |
| Media Companies | Questions regarding design of product | Keep Informed | Allow technical staff to work with stakeholder to answer questions and address concerns and provide test results for validation |

# ACCEPTANCE CRITERIA

1. Since the model will be trained on a large data set of players, it will be able to generalise and predict values properly. The values predicted can be cross validated by querying for a player that has been followed by the club and its scouts and whose worth is known.
2. The Project Manager has set these tasks for achieving successful delivery of the project:
   1. As discussed earlier the objectives of the project have been discussed with the customer and these will be satisfied when delivering the project.
   2. The team’s project manager will review the project before its handing over also an external group will be assigned to check that the team has stayed true to its promises.
   3. Any changes that the customer wishes after the product completion will be addressed in 1-2 weeks of the initial written application by the customer.
   4. The following is a deliverables acceptance document.

|  |  |  |
| --- | --- | --- |
| Item | Concerns | Accepted / Rejected |
| Vision Definition | **Complexity** | **Accepted** |
| Mission Definition | **Relation with Deliverables** | **Accepted** |
| Goals | **Description and structure** | **Accepted** |
| Feature Definitions | **Readability for non-technical stakeholders** | **Accepted** |
| Deliverables definition | **Consistency** | **Accepted** |

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 3

**Player Value Analyser**

**System Requirement Specification**

**12-09-2017**

**Version 1.0**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By: Dr. M. R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **30** |
| 2 | Terms of Reference | **31** |
| 3 | Problem Description | **32** |
| 4 | Functional Hierarchy | **34** |
| 5 | User Interfaces | **36** |
| 6 | Hardware Interfaces | **39** |
| 7 | Software Interfaces | **39** |
| 8 | Logical Databases | **39** |
| 9 | Non-Functional Requirements | **40** |

# 1. INTRODUCTION

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The aim of this document is to gather and analyze and give an in-depth insight of the complete **Player Value Analyzer** software system by defining the problem statement in detail. Nevertheless, it also concentrates on the capabilities required by stakeholders and their needs while defining high-level product features.

|  |  |
| --- | --- |
| Item | Description |
| Purpose | **To jot down the System Requirement Specifications of the Player Value Analyser. The purpose of the SRS to give a complete technical background of our system and its likely implementation.** |
| Audiences | **Developers, Clients and other concerned audiences.** |
| SRS Scope | **This SRS is also aimed at specifying requirements of software to be developed but it can also be applied to assist in the selection of in-house and commercial software products. The standard can be used to create software requirements specifications directly or can be used as a model for defining a organization or project specific standard. It does not identify any specific method, nomenclature or tool for preparing an SRS.** |
| Project Scope | **Primarily, the scope pertains to the football player market for making the Player Value Analyzer project live. It focuses on the company, the stakeholders and applications, football club financers and mass media companies.** |

**References:**

1. Statement of Work
2. Feature Set
3. Standard Football Market Model

# TERMS OF REFERENCE

1. **Background**
   1. The football market is a multibillion-dollar industry and has a lot of business activity. The Football Clubs in the world are constantly trading players for money.
   2. There is also a lot of media interest in these events and this makes it all a very important part of the world’s headlines.
   3. In such cases a sense of authenticity is required as the player values can be very big and no assurances over the performances can be made.
   4. The Clubs ae ready to pay a significant amount of money so as to know how much the real value of a player will be.
   5. The Player Value analyser will consider a player’s rating and his statistics to find out a correct value of the player.
   6. The project will help all interested parties and might even help clubs save money from a risky or unworthy investment.
2. **Objectives**
   1. The Objective of the project is to find the correct approximation of the value of a football player.
   2. Stage-wise objectives.

* Collecting Player Data
* Analysing Player Data
* Finding a pattern and plotting values
* Formulating significant factors contributing to Player’s value
* Displaying Player Value and Stats
  1. Target Audience is
* Football Clubs
* Football Media Companies
* Fans and general interested users
  1. Resources required would be clearer along the course of the implementation but the basic necessities include player data, machine learning techniques and UI design.

1. **Issues**
   1. Efficiency – Efficiency of the algorithm and its outcome to predict values.
   2. Relevance – Proving relevance to stakeholders.
   3. Effectiveness – Exact outputs of the project and realization of benefits.
   4. Impact – the market is ever growing and the product’s use will never cease to exist.
   5. Sustainability – The project if successful will have wide ranging benefits and will become self-sustainable soon after its initiation.
2. **Methodology**
   1. Data Acquisition, Filtering, Data Mining, Data Analysis, Data Sorting & Classification
   2. Stakeholder involvement at initial and final stages will ensure smooth implementation
   3. The planning and designing phase and implementation phase will each take about 1 month.
   4. The information collection tools to be used throughout the project for monitoring purposes are Media Reports and similar online resources
   5. Data analysis rules

1. **Expertise**

The expertise needed for doing a project defines a set of professional requirements for the individuals and teams involved in project implementation. It will be the basis for team building, including training and skill assessment.

The Expertise section of a Project Terms of Reference template should identify the following:

* 1. The type of work involved in the project is Data Analysis and Machine Learning along with UI development.
  2. The type of skills and abilities required to do project work are Machine Learning in Python, Data Processing, DBMS, UI development, etc.
  3. 4 students from T.Y, B. Tech will be involved in the development of the project.
  4. The period of engagement of each team member is about the same, roughly 3 months.
  5. A description of the duties and responsibility per teammate has been provided in earlier documents and will be further described in the succeeding documentation.
  6. The relationship between the team members, including leadership roles are specified in the following table.

|  |  |
| --- | --- |
| Name | Roles |
| Anup Mahindre | Analyst |
| Hiranyey Gajbhiye | Developer |
| Neeraj Ganu | Leader |
| Sanket Ostwal | Designer |

1. **Reporting**

Reports provide valued information about project performance over a certain period. Reporting is a process that starts once a project is launched and continues until the project is completed and its product is handed over. Reporting requirements will define how to write and submit project reports and what information to include. The Reporting Requirements section of a Terms of Reference template should clearly specify the requirements for the reporting process, and might include the details of:

* 1. Table of contents for project reports/ Rules for composing annexes
  2. Report templates/ The language to be used in reports
  3. Computer software programmes to be used/ Submission dates
  4. People responsible for reporting and approving

1. **Work Plan**

A work plan is a kind of strategy that aims to help solve problems throughout a project and boost employee drive and focus. It determines what actions need to be taken to start, implement, and complete the project within a specified time period and under defined budget. It is often used as a general guide for developing a project implementation plan. The Work Plan section of a Project Terms of Reference template should set out the activities and necessary resources required for achieving the project’s results and purpose. It should therefore include a summary of the anticipated work and time schedule, which are based upon the following:

* 1. An analysis of the issues, in terms of the evaluation criteria
  2. The proposed implementation methodology/ The reporting requirements
  3. It will be further covered in the Project Plan document.

# 3. PROBLEM DESCRIPTION

|  |  |
| --- | --- |
| The problem of | Effectively determining the value that a player brings to the club. |
| Affects | Clubs, Clubs financial teams, Players |
| The impact of which is | * The current scenario is that player prices are inflated, so clubs have to borrow money or get funding from sponsors to make big transfers possible * However, this isn’t sustainable, since not all players bring actual value to the club and increased borrowing causes financial instability. * This decreases fair-play in the sense that clubs that secure more money end up on the top. |
| A successful solution would | * A successful solution will give an idea about the real value of a player based on his performance and ratings from recent time . * It will help bring down the overall inflation in the transfer market, helping improve stability and fair-play. |

|  |  |
| --- | --- |
| For | Football Clubs |
| Who | Needs it for determining values of players they want to bring in/ put on transfer list. |
| The ‘Player Value Analyzer’ | is a statistical and analytical tool |
| That | Determines player value based on statistics and performance rather than hype in football world. |
| Unlike | Scouts and advisors |
| Our product | Won’t have biases and will not have prejudices. |

# FUNCTIONAL HIERARCHY

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 1 | Build Player Profile | Description |
| Objective ID | 1 | Acquire Data | |
| Process ID: 1 | Acquire Player Statistics |
| Process ID: 2 | Clean Data Abnormalities |
| Objective ID | 2 | Analyse Data |  |
| Process ID: 1 | Filter Relevant Data |
| Process ID: 2 | Ascertain Data Correctness |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 2 | Classify players | Description |
| Objective ID | 1 | Extract Player Information | |
| Process ID: 1 | Extract Player Positions |
| Process ID: 2 | Extract Player Abilities |
| Objective ID | 2 | Classify Players |  |
| Process ID: 1 | Group Players |
| Process ID: 2 | Verify Player Groups |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 3 | Process Player Profiles | Description |
| Objective ID | 1 | Pre-process Player Data | |
| Process ID: 1 | Associate Player Profiles |
| Process ID: 2 | Generate Player Features |
| Objective ID | 2 | Decide Statistical Model | |
| Process ID: 1 | Choose Statistical Model |
| Process ID: 2 | Optimise Choices |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 4 | Process User Queries | Description |
| Objective ID | 1 | Recognise User Queries | |
| Process ID: 1 | Standardize Query Format |
| Process ID: 2 | Execute Prediction |
| Objective ID | 2 | Handle User Input Data | |
| Process ID: 1 | Pre-Process User Input Data |
| Process ID: 2 | Append Relevant Dataset |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 5 | Demonstrate Player Statistics | Description |
| Objective ID | 1 | Generate Player Statistics | |
| Process ID: 1 | Find Player Statistics |
| Process ID: 2 | Communicate Relevant Statistics |
| Objective ID | 2 | Display Player Statistics | |
| Process ID: 1 | Choose Relevant Statistics |
| Process ID: 2 | Generate Statistical Graphs |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 6 | Estimate Player Value | Description |
| Objective ID | 1 | Validate Player Data | |
| Process ID: 1 | Fetch Player Data |
| Process ID: 2 | Feed Data Model |
| Objective ID | 2 | Generate Predicted Value | |
| Process ID: 1 | Communicate Player Value |
| Process ID: 2 | Display Predicted Value |

|  |  |  |  |
| --- | --- | --- | --- |
| Goal-ID | 7 | Process Feedback | Description |
| Objective ID | 1 | Generate Feedback Mechanism | |
| Process ID: 1 | Assess User Feedback |
| Process ID: 2 | Process User Feedback |
| Objective ID | 2 | Apply Improvement Steps | |
| Process ID: 1 | Determine Feedback Response |
| Process ID: 2 | Apply Necessary Changes |

# USER INTERFACES

5.1 Abbreviated UI, it is the junction between a user and a computer program. An interface is a set of commands or menus through which a user communicates with a program. A command-driven interface is one in which you enter commands. A menu-driven interface is one in which you select command choices from various menus displayed on the screen.

The user interface is one of the most important parts of any program because it determines how easily you can make the program do what you want. A powerful program with a poorly designed user interface has little value. Graphical user interfaces (GUIs) that use windows, icons, and pop-up menus have become standard on personal computers.

GUI is a program interface that takes advantage of the computer's graphics capabilities to make the program easier to use. Well-designed graphical user interfaces can free the user from learning complex command languages. On the other hand, many users find that they work more effectively with a command-driven interface, especially if they already know the command language.

Graphical user interfaces, such as Microsoft Windows and the one used by the Apple Macintosh, feature the following basic components:

* Pointer: A symbol that appears on the display screen and that you move to select objects and commands. Usually, the pointer appears as a small angled arrow. Text -processing applications, however, use an I-beam pointer that is shaped like a capital I.
* Pointing device: A device, such as a mouse or trackball, that enables you to select objects on the display screen.
* Icons: Small pictures that represent commands, files, or windows. By moving the pointer to the icon and pressing a mouse button, you can execute a command or convert the icon into a window. You can also move the icons around the display screen as if they were real objects on your desk.
* Desktop: The area on the display screen where icons are grouped is often referred to as the desktop because the icons are intended to represent real objects on a real desktop.
* Windows: You can divide the screen into different areas. In each window, you can run a different program or display a different file. You can move windows around the display screen, and change their shape and size at will.
* Menus: Most graphical user interfaces let you execute commands by selecting a choice from a menu.

In addition to their visual components, graphical user interfaces also make it easier to move data from one application to another. A true GUI includes standard formats for representing text and graphics. Because the formats are well-defined, different programs that run under a common GUI can share data. This makes it possible, for example, to copy a graph created by a spreadsheet program into a document created by a word processor.

5.2 Characteristics of Successful User Interfaces

* **Clear**: Clarity is the most important element of user interface design. Indeed, the whole purpose of user interface design is to enable people to interact with your system by communicating meaning and function. If people can’t figure out how your application works or where to go on your website they’ll get confused and frustrated.
* **Concise**: Clarity in a user interface is great, however, you should be careful not to fall into the trap of over-clarifying. It is easy to add definitions and explanations, but every time you do that you add mass. Your interface grows. Add too many explanations and your users will have to spend too much time reading through them. Keep things clear but also keep things concise. When you can explain a feature in one sentence instead of three, do it.
* When you can label an item with one word instead of two, do it. Save the valuable time of your users by keeping things concise. Keeping things clear and concise at the same time isn’t easy and takes time and effort to achieve, but the rewards are great.
* **Familiar**: Many designers strive to make their interfaces ‘intuitive’. But what does intuitive really mean? It means something that can be naturally and instinctively understood and comprehended. But how can you make something intuitive? You do it by making it ‘familiar’. Familiar is just that: something which appears like something else you’ve encountered before. When you’re familiar with something, you know how it behaves – you know what to expect. Identify things that are familiar to your users and integrate them into your user interface.
* **Responsive**: Responsive means a couple of things. First of all, responsive means fast. The interface, if not the software behind it, should work fast. Waiting for things to load and using slaggy and slow interfaces is frustrating. Seeing things load quickly, or at the very least, an interface that loads quickly (even if the content is yet to catch up) improves the user experience. Responsive also means the interface provides some form of feedback. The interface should talk back to the user to inform them about what’s happening. Have you pressed that button successfully? How would you know? The button should display a ‘pressed’ state to give that feedback.
* **Consistent**: Consistent interfaces allow users to develop usage patterns – they’ll learn what the different buttons, tabs, icons and other interface elements look like and will recognize them and realize what they do in different contexts. They’ll also learn how certain things work, and will be able to work out how to operate new features quicker, extrapolating from those previous experiences.
* **Attractive**: This one may be a little controversial but I believe a good interface should be attractive. Attractive in a sense that it makes the use of that interface enjoyable. Yes, you can make your UI simple, easy to use, efficient and responsive, and it will do its job well – but if you can go that extra step further and make it attractive, then you will make the experience of using that interface truly satisfying. When your software is pleasant to use, your customers or staff will not simply be using it – they’ll look forward to using it. There are of course many different types of software and websites, all produced for different markets and audiences. What looks ‘good’ for any one particular audience will vary. This means that you should fashion the look and feel of your interface for your audience. Also, aesthetics should be used in moderation and to reinforce function. Adding a level of polish to the interface is different to loading it with superfluous eye-candy.
* **Efficient**: A user interface is the vehicle that takes you places. Those places are the different functions of the software application or website. A good interface should allow you to perform those functions faster and with less effort. Now, ‘efficient’ sounds like a fairly vague attribute – if you combine all of the other things on this list, surely the interface will end up being efficient? Almost, but not quite. What you really need to do to make an interface efficient is to figure out what exactly the user is trying to achieve, and then let them do exactly that without any fuss. You have to identify how your application should ‘work’ – what functions does it need to have, what are the goals you’re trying to achieve? Implement an interface that lets people easily accomplish what they want instead of simply implementing access to a list of features.
* **Forgiving**: Nobody is perfect, and people are bound to make mistakes when using your software or website. How well you can handle those mistakes will be an important indicator of your software’s quality. Don’t punish the user – build a forgiving interface to remedy issues that come up. A forgiving interface is one that can save your users from costly mistakes.
* For example, if someone deletes an important piece of information, can they easily retrieve it or undo this action? When someone navigates to a broken or non-existent page on your website, what do they see? Are they greeted with a cryptic error or do they get a helpful list of alternative destinations?

|  |  |  |  |
| --- | --- | --- | --- |
| UI-ID | UI Name | Type | Scope |
| 1 | Collect Data | Input | Players are added to the database. |
| 2 | Examine Data | Input | Clarify the data, collecting only the useful information. |
| 3 | Acquire statistics | Input | System acquires statistics of the selected player. |
| 4 | Extract player information | Input | User asks for information of player. |
| 5 | Normalize Data | Command | This will normalize the input data. |
| 6 | Display Features | Navigation | The features of the data model will be displayed in a graph. |
| 7 | Statistical Model | Command | Here the appropriate statistical model can be chosen to use in the machine learning module. |
| 8 | Process Queries | Command | Query is processed to acquire data and proceed. |
| 9 | Execute prediction | Command | Data from query is fed to the model and predictions are made. |
| 10 | Pre-process Data | Command | Data that can be added to database is acquired and pre-processed to standardise its format. |
| 11 | Append data | Command | Pre-processed data is verified and appended to the dataset. |
| 12 | Estimate value | Command | The player value will finally be evaluated. Feedback mechanism will be triggered after this. |
| 13 | Search Statistics | Form | View of the database with search functionalities |
| 14 | Transmit Statistics | Form | Selecting a player and choosing file format in which user needs to transmit |
| 15 | Choose Statistics | Form | Sorting, selecting, modifying player’s details on the basis of single or multiple attributes |
| 16 | Generate Statistical Graphs | Graphs | Visualizing player’s attributes for better understanding of condition of Player |
| 17 | Fetch Player Data | Command | Acquire player’s details from Football APIs and displaying, appending data |
| 18 | Feed Data Model | Input | Filling players data through client’s machine |
| 19 | Transmit Value | Command | Calculating player’s value using data acquired by API or User |
| 20 | Display Predicted Value | NL | Showing the result of the prediction model and setting the base price of the player |
| 21 | Value Feedback | NL | The form will gather feedback about the correctness of the value. |
| 22 | Feedback Analysis | Input | The Feedback will be analysed and analysed to improve the results of the system. |
| 23 | Navigation Bar | Navigation | It will appear on each page from where users can switch to any option. |
| 24 | Main Page | Menu | Users can navigate to different parts of the website using the main page. |

# HARDWARE INTERFACES

|  |  |
| --- | --- |
| Profile | Description (minimum requirements) |
| Processor | **Intel 5th Generation** |
| RAM | **8 GB RAM** |
| Server-Side Technology | * **Database storage space: 1 GB** * **Monitor of resolution 1024 x 768** |
| Client-Side Technology | * **Monitor of resolution 1024 x 768** * **Working Internet Connection and Port** |
| External Devices | * **Monitor** * **Mouse** * **Keyboard** |

# SOFTWARE INTERFACES

|  |  |
| --- | --- |
| Profile | Description |
| Front-end Capabilities | **Browser, HTML 5 support** |
| Back-end Capabilities | **PHP** |
| Programming Languages | **Python, PHP** |
| Operating Environment | **Any** |
| Software Platform | **Browser** |
| Database Servers | **MySQL** |
| Framework Resources | **NA** |
| API (If Any) | **Football data APIs** |
| Other Services/Resources | **NA** |
| Communication Interfaces | **Email** |

# LOGICAL DATABASES

|  |  |  |
| --- | --- | --- |
| Database Name | Parameter | Scope |
| FIFA Ratings | Skill Ratings | Input Data |
| Player Details | All the basic details of player | Input Data |
| Player Stats | Player Statistics | Input Data |
| Player Category | Special Characteristics/ Classifications of players | Calculated Data |
| Player Value | Predicted Value of Players | Output Data |
| Window | Recent Transfer Values and Club Sales | Updatable record |

# NON-FUNCTIONAL REQUIREMENTS

* Reliability: Specify the factors required to establish the required reliability of the software system at time of delivery. If you have MTBF requirements, express them here. This doesn’t refer to just having a program that does not crash. This has a specific engineering meaning.
* Availability: Specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart. This is somewhat related to reliability. Some systems run only infrequently on-demand (like MS Word). Some systems have to run 24/7 (like an e-commerce web site). The required availability will greatly impact the design. What are the requirements for system recovery from a failure? “The system shall allow users to restart the application after failure with the loss of at most 12 characters of input”.
* Security: Specify the factors that would protect the software from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to:
  + Utilize certain cryptographic techniques
  + Keep specific log or history data sets
  + Assign certain functions to different modules
  + Restrict communications between some areas of the program
  + Check data integrity for critical variables
* Maintainability: Specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices. If someone else will maintain the system
* Portability: Specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include:
  + Percentage of components with host-dependent code
  + Percentage of code that is host dependent
  + Use of a proven portable language
  + Use of a particular compiler or language subset
  + Use of a particular operating system
* Correctness - extent to which program satisfies specifications, fulfills user’s mission objectives
* Efficiency - amount of computing resources and code required to perform function
* Flexibility - effort needed to modify operational program
* Interoperability - effort needed to couple one system with another
* Reliability - extent to which program performs with required precision
* Reusability - extent to which it can be reused in another application
* Testability - effort needed to test to ensure performs as intended
* Usability - effort required to learn, operate, prepare input, and interpret output

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Characteristic** | **H/M/L** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
| 1 | Correctness | H |  | 2 |  |  |  |  |  |  |  |  |  |  |
| 2 | Efficiency | L |  |  |  |  |  |  |  |  |  | 10 |  |  |
| 3 | Flexibility | L |  |  |  |  |  |  |  |  |  |  | 11 |  |
| 4 | Integrity/Security | H |  |  |  |  |  | 6 |  |  |  |  |  |  |
| 5 | Interoperability | M |  |  |  |  |  |  |  |  | 9 |  |  |  |
| 6 | Maintainability | H |  |  |  |  | 5 |  |  |  |  |  |  |  |
| 7 | Portability | M |  |  |  |  |  |  | 7 |  |  |  |  |  |
| 8 | Reliability | H | 1 |  |  |  |  |  |  |  |  |  |  |  |
| 9 | Reusability | L |  |  |  |  |  |  |  |  |  |  |  | 12 |
| 10 | Testability | M |  |  |  |  |  |  |  | 8 |  |  |  |  |
| 11 | Usability | H |  |  | 3 |  |  |  |  |  |  |  |  |  |
| 12 | Availability | M |  |  |  | 4 |  |  |  |  |  |  |  |  |

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 4

**Player Value Analyser**

**Feasibility Study Report**

**19-09-2017**

**Version 1.0**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By: Dr M. R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **44** |
| 2 | Description of Services | **44** |
| 3 | Technology Considerations | **45** |
| 4 | Feasibility Study Results | **46** |
| 5 | References | **46** |

# INTRODUCTION

|  |  |
| --- | --- |
| Item | Description |
| Scope of Study | 1. Acquire all the required statistics of the player. 2. Provide an estimate market value of the player. 3. Give a good estimate of the talent of the player. |
| Audiences | 1. Football clubs looking to buy/sell the players. 2. Agents of football players. 3. Football Fans 4. News & Media Companies |
| Project Type | Medium Scale |
| Platform Details | Existing Domain Platforms   1. CIES Football Lab 2. transfermarkt   Existing Technology Usage  Front End –   1. HTML 2. CSS 3. PHP   Back End –   1. MySQL 2. MongoDB 3. Python |

# DESCRIPTION OF SERVICES

|  |  |  |  |
| --- | --- | --- | --- |
| Service -ID | Service Name | Audience | Scope |
| S-1 | Collect Player Data | Stakeholder | Detailed information about player. |
| S-2 | Classify Players | Stakeholder | The classification of data according to player’s preferred position. |
| S-3 | Illustrate Statistics | User | Graphical representation of player’s statistics. |
| S-4 | Display Player Value | End User | The User will be able to see the value of every football player. |
| S-5 | Identify Player | User | A player can be searched in the database to find information about him |
| S-6 | Display Player Information | Stakeholder | The Football Clubs can get complete information about the player on the website. |
| S-7 | Examine Feedback | End User | The feedback about the system will be considered and processed. |

# TECHNOLOGY CONSIDERATIONS

|  |  |  |
| --- | --- | --- |
| Current Technology | | |
| Type | **Parameter** | **Description** |
| Hardware | CPU | Intel Xeon E2630 v4 – 10 core processor, 2.2 GHz with Turbo boost up to 3.1 GHz |
| RAM | 16GB |
| GPU | GeForce GTX 1060 (For tensor Flow) |
| Software | IDE’s | Visual Studio 17, Jupyter, PHPmyadmin |
| Local Server | XAMPP (PHP) |
| Browser | Mozilla or Chrome |

|  |  |  |
| --- | --- | --- |
| Deployment Technology | | |
| Type | **Parameter** | **Description** |
| Software | Browser | Mozilla or Chrome |
| Support | HTML5, CSS, BOOTSRAP, JavaScript, jQuery |
| Hardware | Device | Desktop, Laptop, Tablet, Mobile |
| Screen | Screen with minimum 1024×576 resolution |

# FEASIBILITY STUDY RESULTS

|  |  |  |  |
| --- | --- | --- | --- |
| Option | Outcome | Ranking | Discussion |
| Collect Player Statistics | Expected | H | The player statistics are available and are successfully retrieved and sent for processing. |
| Unexpected | L | Player statistics aren’t available. This is unlikely since data about most players is available |
| Check Player Data Correctness | Expected | H | Data is correct. Statistics are available in the right format |
| Unexpected | L | Statistics aren’t in precise format. If few pieces of data is missing/incorrect, it can be replaced with an average value. In case various data fields for a player aren’t available, then value prediction can’t be made. |
| Illustrate Statistical  Graphs | Expected | M | The statistics will be represented with appropriate graphs to make it attractive for the user. |
| Unexpected | L | The statistics will be too inconsistent to be illustrated with the help of graphs. |
| Display Player Value | Expected | H | The Player Value prediction mapped for the player will be displayed correctly. |
| Unexpected | L | Value displayed will be displayed incorrectly because of   1. Database issue 2. Prediction issue |
| Process Queries | Expected | H | Queries are specific. Processing works properly and output is produced |
| Unexpected | L | Query has anomaly in it. In such case, the problem will be pointed out and user will be able to correct it. |
| Display Player Information | Expected | M | Required player information is available and displayed correctly**.** |
| Unexpected | L | Player information unavailable and displaying is not possible. |
| Process feedback | Expected | M | Feedback is positive. The value generated are precise and consistent. |
| Unexpected | M | Feedback is negative. In such case model is retrained with correct parameters that generate positive feedback. |

# 

# REFERENCES

1. Statement of Work
2. Feature Set
3. System Requirement Specifications
4. Standard Football Market Model
5. football-data.org
6. CIES football laboratory
7. whoscored.com
8. squawka.com

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 5

**Player Value Analyser**

**Project Plan Outline**

**25-09-2017**

**Version 1.1**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By: Dr M. R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **49** |
| 2 | Project Milestones | **49** |
| 3 | Work Breakdown Structure | **50** |
| 4 | Project Communication | **51** |
| 5 | Activity Register | **52** |
| 6 | Task Prioritization | **53** |
| 7 | Risk Register | **54** |

# INTRODUCTION

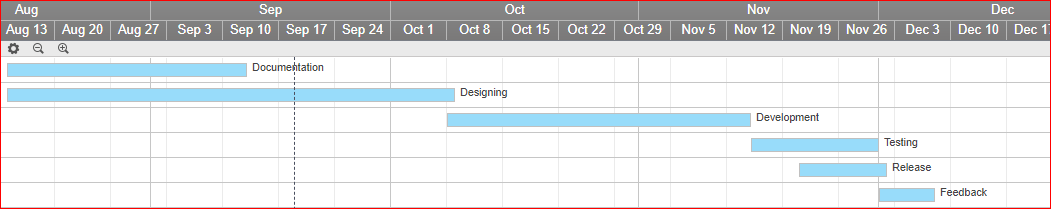
|  |  |
| --- | --- |
| Deliverables | Benefits |
| 1. SOW | Gives an idea of what the system is. |
| 2. Feature Set | Provides the set of features the system will provide. |
| 3. SRS | Specifies the requirements for the system. |
| 4. Feasibility Study | Gives an account of how feasible it is to use the system. |
| 5. Project Plan | Will provide information on how the project will be executed. |
| 6. Sprint Level Planning Activity | Planning will help in easy execution of the system. |
| 7. Sprint Level Design Activity | Preparing the design will make the implementation faster because a blueprint will be available. |
| 8. Software Configuration Management Plan | It will make the execution of the software much easier as there is a plan in place. |
| 9. Sprint Execution | The system will be available to use as early as possible. |
| 10. Sprint Review | Fast review of the system so that so that errors can be removed as early as possible. |

# PROJECT MILESTONES

|  |  |  |
| --- | --- | --- |
| Milestones | Phase | Description |
| 1 | Inception | Delivering Statement of Work document |
| 2 | Inception | Delivering Feature Set document |
| 3 | Elaboration | Feasibility study and Project Plan using AGILE |
| 4 | Elaboration | Sprint level planning activity |
| 5 | Construction | Sprint Plan and Sprint Design |
| 6 | Construction | Software Configuration Management Plan (SCMP) and Sprint Execution |
| 7 | Transition | Sprint Review and Sign- offs |
| 8 | Transition | Feedback |

# WORK BREAKDOWN STRUCTURE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WBS ID | WBS Package | Role | Description | Delivery Date |
| 1 | Documentation | Inception | Creation of SOW, FRS, SRS | 12 Sept 2017 |
| 2 | Designing | Elaboration | Making Prototypes | 08 Oct 2017 |
| 3 | Development | Construction | Development of Real System using appropriate languages | 14 Nov 2017 |
| 4 | Testing | Construction | Testing of System for Defects and checking for correctness | 30 Nov 2017 |
| 5 | Product Release | Transition | Marketing, Managing of the System in live environment | 1 Dec 2017 |
| 6 | Feedback | Transition | Taking user experience as feedback and modifying System | 7 Dec 2017 |

**GANTT CHART**

# PROJECT COMMUNICATION

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Communication Type | Description | Frequency | Format | Participants/ Distribution | Deliverable | Owner |
| Weekly Status Report | Email summary of project status | Weekly | In Person | Project Guide,  Project Team | Status Report | Project Manager |
| Weekly Project Team Meeting | Meeting to review action register and status | Weekly | In Person | Project Team | Updated Action Register | Project Manager |
| Project Monthly Review (PMR) | Present metrics and status to team and sponsor | As Needed | In Person | Project Guide, Team, and Stakeholders | Status and Metric Presentation | Project Manager |
| Project Gate Reviews | Present closeout of project phases and kick-off next phase | As Needed | In Person | Project Sponsor, Team and Stakeholders | Phase completion report and phase kick-off | Project Manager |
| Technical Design Review | Review of any technical designs or work associated with the project | As Needed | In Person | Project Team | Technical Design Package | Project Manager |

# ACTIVITY REGISTER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Activity Number** | **Activity Name** | **Activity description** | **Responsibility** | **Comments** |
| 1 | Prepare  Documentation | * Create Project Initiation Documents | * Neeraj Ganu is responsible for coordinating with the team. | * Meet Deadlines |
| * Documents: SOW, Feature Set and SRS | * WBS Package 1 |
| 2 | Conceptualise Design | * Evaluate Feasibility | * Neeraj Ganu is responsible for execution of project planning phase. | * Quick Execution Required |
| * Develop Project Plan | * WBS Package 2 |
| 3 | Collect Data | * Acquire Data from Sources on the Internet | * Hiranyey Gajbhiye is responsible for acquiring correct data | * WBS Package 2 |
| * Important phase for smoot development |
| 4 | Developing System | * Develop Machine Learning Model | * Anup Mahindre is responsible for delegating everyone with instructions for development. | * Development in Sprints |
| * Implement Model to Predict Values | * WBS Package 3 |
| 5 | Design UI | * Create User Interface | * Sanket Ostwal will oversee the UI creation activity. | * WBS package3 |
| * Design UI to appropriately display the statistics | * The phase execution will have to run parallelly with development stage |
| 6 | Checking for bugs | * Unit and System Testing | * Hiranyey Gajbhiye will be in charge of creating test cases and checking for bugs | * Preparing Test Cases * WBS Package 4 |
| * Debugging |
| 7 | Releasing Product | * Advertising System | * Neeraj Ganu will be responsible for the marketing of the product. * Sanket Ostwal will also share the responsibility. | * Good Marketing Strategies * WBS Package 4 |
| * Finding Clients |
| 8 | Feedback of System | * Taking reviews from customers * Implementing new features | * Anup Mahindre will oversee the feedback and update activities. | * Understanding what changes are needed * WBS Package 6 |

# TASKS PRIORITAZATION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Task is of high importance, with high urgency factor.***  *Must be done today & to high standard.*  *Action ASAP* |  | ***High Importance*** | ***Low Importance*** | ***Task is of low importance, with high urgency factor.***  *These tasks need to be completed on time.*  *ONLY spend sufficient time on them as not important.*  *Don’t be diverted* |
| ***High Urgency*** | 1. **Collect Football Player Statistics** 2. **Initiation Documentation** 3. **Create algorithm to calculate values** 4. **Create Project Repository** | 1. **Build Player Profiles** 2. **Rate Player** 3. **Finding Football APIs** 4. **Study Similar Projects** 5. **Study Legality Issues** |
| ***Task is of high importance, but has low urgency factor.***  *By nature long-term so need to:*   1. *Set target if none exists.* 2. *Break-up into chunks of work* | ***Low Urgency*** | 1. **Create UI** 2. **Determine System’s Accuracy** 3. **Classify Players** 4. **Display Player Values** 5. **Version Control Mechanism** | 1. **Integrate Feedback** 2. **Create Graphs for Acquired Data** | ***Task is both low in importance & urgency.***  *Discard as many of these tasks as possible because they cause great harm to your productivity.*  *Delegate if they develop another’s KSA’s.* |

# RISK REGISTER

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Risk Description** | **Likely Cause of Risk Occurring** | **Effect on Project** | **Phase Affected** | ***Severity Level*** | **Ability to Detect** | **Risk Rank** |
| **1** | **Player statistics not available** | 1. **Not enough data available** 2. **Data not available for open use** | **Failure to make prediction** | **Transition** | **High** | **Moderate** | **Serious** |
| **2** | **Player statistics aren’t correct** | 1. **Data source might have anomalies** | **Statistics need to be fetched again** | **Construction** | ***Med*** | **Moderate** | **Modest** |
| **3** | **Player statistics outdated** | 1. **The player statistics were not updated for the next season** | **predictions oblivious of current scenario** | **Transition** | ***Med*** | **Moderate** | **Serious** |
| **4** | **Process queries are anomalous** | 1. **The user entered wrong player information** 2. **Required information in query is missing** | **Prediction cannot be made** | **Transition** | ***Low*** | **Easy** | **Trivial** |
| **5** | **Wrong features selected/dropped** | 1. **Lack of domain knowledge** 2. **Improper model selection** | **Biased or High variance model** | **Construction** | ***Med*** | **Moderate** | **Serious** |
| **6** | **Insufficient data** | 1. **The data available is insufficient for generalization** | **Overfitting** | **Construction** | ***High*** | **Complex** | **Critical** |

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 6

**Player Value Analyser**

**Project Backlog**

**07-11-2017**

**Version 1.0**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | Gr. No. | Name | Roles |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By: Dr M. R. Dube**

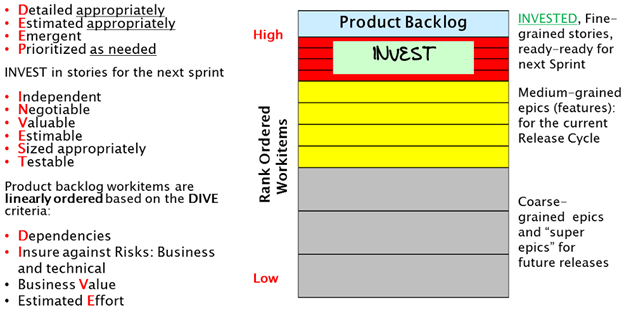
**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **57** |
| 2 | Sprint Planning And Work-Item Granularity | **57** |
| 3 | Release Planning And Work Granularity | **58** |
| 4 | Product Planning And Work-Item Granularity | **59** |
| 5 | Product Backlog: Goals Granularity | **61** |

# INTRODUCTION

*A product backlog stores, organizes and manages all work items that you plan to work on in the future. The key characteristics of a well-organized and managed product backlog are summarized in the image below. DEEP, INVEST and DIVE are meaningful words.*



*Figure 1: Characteristics of a Managed Product Backlog*

*The* ***granularity*** *or size of work items should be determined based on how far into the future you are planning a product, i.e., the planning horizon. It is the observation that the longer or shorter the planning horizon, the larger or smaller the work items. This makes sense as it takes a lot more effort to develop, specify and maintain a large number of small-grain work items compared to developing, specifying and maintaining a small number of large-grain work items. Smaller work items, stories, are typically developed by breaking down larger work items, epics. Stories are the unit of software design, development and value delivery.*

***DEEP product backlog***

*A product backlog may have several hundred or more work items, hence the acronym DEEP. Work items can be comprised of stories, defects and test sets. DEEP is acronym capturing the essence of the logical structure of product backlog.*

* ***Detailed appropriately****: Work-items in the backlog are specified at an appropriate level of detail.*
* ***Estimated appropriately****: Work-items in the product backlog are estimated appropriately.*
* ***Emergent****: Product backlog is not frozen or static; it evolves or emerges on an on-going basis in response to product feedback, and changes in competitive, market and business. New backlog items are added, existing items are groomed (revised, refined, elaborated) or deleted or re-prioritized.*
* ***Prioritized as needed****: Work-items in the backlog are linearly rank-ordered as needed.*

# 2. SPRINT PLANNING AND WORK-ITEM GRANURALITY

*If the planning horizon is the next, i.e., upcoming sprint or iteration (typically 2 to 4 weeks), each Work-items is small enough to fit in a single sprint, and is 100% ready (“ready-ready”) to be worked on, as indicated in Figure 1 – see the top red-color region. A ready-ready story has already been analyzed with clear definition (User Role, Functionality, and Business Value) and associated Acceptance Criteria. Work-items planned for the next sprint are stories, defects and test sets. The Work-items in the next sprint have the highest rank order compared to Work-items in later sprints or later release cycles. I will soon explain how this rank ordering is done.*

*The rank order information is used to decide the order in which the team will undertake work on Work-items in a sprint backlog, and also decide which incomplete Work-items to push out to the release or product backlog at the end of a sprint time-box.*

*Work-items in the next sprint collectively satisfy the well-known INVEST criteria; it is a meaningful English word, as well as an interesting acronym coined by Bill Wake. Its letters represent important characteristics of Work-items in the next sprint backlog. Stories in the next sprint backlog should be:*

* ***Independent of each other****: At the specification level stories are independent; they offer distinctly different functionality and don’t overlap. Moreover, at the implementation level these stories should also be as independent of each other as possible. However, sometimes certain implementation-level dependencies may be unavoidable.*
* ***Negotiable****: Stories in the next sprint are always subject to negotiations and clarifications among product owner (business proxy) and the members of agile development team.*
* ***Valuable****: Each story for the next sprint offers clear value or benefit to either external users or customers (outside the development team), or to the team itself, or to a stakeholder. For most products and projects, most stories offer value to external users or customers.*
* ***Estimable****: From the specification of story itself, an agile team should be able to estimate the effort needed to implement the story; this estimate is in relative size terms (story points), and optionally, it can also be in time units (such as ideal staff-hours or staff-days for the whole team). Thus, stories are estimated in story points, and also often in ideal time units.*
* ***Sized Appropriately****: A simpler interpretation of this criterion is that each story is Small enough to be completed and delivered in a single sprint. The letter “S” can be taken to mean Sized Appropriately; specifically, each story should take no more than N/4 staff-weeks of team effort for an N-week long sprint. Thus, for a 2-week sprint, each story should take no more than 2/4 staff-week = 0.5 staff-week = 20 staff-hours of effort. A story substantially larger than 20 staff-hours of total effort should be treated as an epic and be broken down into smaller stories. For a 4-week sprint, each story should take no more than 4/4 staff-week = 1 staff-week = 40 staff-hours of effort. If a sprint backlog has a mix of stories that are small, medium or large size stories (their average far exceeds N/4 staff-weeks), the average cycle time across all stories will increase dramatically reducing the team velocity.*
* ***Testable****: Each story specification is very clear to be able to develop all test cases from its acceptance criteria (which is part of the specification).*

*Stories may be broken down into implementation tasks, such as Analysis, Design, Code Development, Unit Testing, Test Case Development, On-line Help, etc. These tasks need to be SMART:*

* + *S: Specific*
  + *M: Measurable*
  + *A: Achievable*
  + *R: Relevant*
  + *T: Time-boxed (typically small enough to complete in a single day)*

*If a story needs to take no more than N/4 staff-week of team effort (ex. 20 staff-hours for 2-week sprints), all SMART tasks in a story should add up to no more than N/4 staff-week of team effort. If you have 5 tasks, each task on an average should take 4 hours of ideal time effort or less. Stories and its SMART tasks for the next sprint are worth INVESTing in, as the return on that INVESTment is high because they are scheduled to be worked on and delivered as working software in the next sprint itself.*

# 3. RELEASE PLANNING AND WORK GRANURALITY

*If the planning horizon is an upcoming release cycle (typically 8 to 26 weeks, or 2 to 6 months long – consisting of several sprints), Work-items are “medium-grain” as shown in the middle yellow color region of Figure 1. Typically, many of these Work-items are epics; however, they should be still small enough to fit in a release cycle and can be completed over two or more sprints in a release cycle. These epics are typically called features or feature-epics. These feature-epics should still be specified with User Role, Action, Value and Acceptance Criteria formalism that is often used for specifying stories, but now you are capturing a larger functionality represented by a feature-epic. Feature-epics are divided into stories – small enough to fit in a sprint – before the sprint in which a story will be implemented.*

*Over the time horizon of an entire release cycle, INVESTing in stories for an entire release cycle has poor returns, because it takes a lot of effort to ensure that the INVEST criteria is being satisfied correctly for a large number of stories covering an entire release cycle, and those stories are much more likely to change over the release cycle spanning several sprints; so this kind of INVESTment may not yield expected results as stories will very likely change during an entire release cycle after they have been specified.*

***Feature-epics*** *in a release cycle can and should be estimated in relative size terms, but without expending the effort needed to break down all feature-epics in a release cycle into individual stories. This epic-level estimation can be done by comparing relative sizes of epics.*

*It still makes sense to rank order feature-epics in a release cycle to decide which ones will be scheduled in Sprint 1, 2, 3, and so on. However, this assignment may change as each sprint is completed and more information and learning emerge.*

# 4. PRODUCT PLANNING AND WORK-ITEM GRANURALITY

*If the product planning horizon is over multiple release cycles (typically 6 to 24 months) going beyond the current release cycle, Work-items are “****coarse-grain****” as shown in the bottom gray color region of Figure 1. These large epics or super epics require two or more release cycles to complete. These super epics may be described in plain English (bulleted text) or with screen mock-up or video or prototype or with any form of expression suitable to express the intent and value of super epics. These super epics are divided into feature-epics – small enough to fit in a single release cycle – before the release cycle in which that feature-epic will be implemented.*

*Over the time horizon of multiple release cycles, INVESTing in stories has even poorer returns compared to INVESTing in stories for a single release cycle. This kind of INVESTment will not yield expected results as stories are very likely to change over much longer duration of multiple release cycles.*

*Large epics or super epics that need multiple release cycles to be implemented can and should be estimated in relative size terms, but without expending the effort needed to break down large epics into feature-epics, and breaking those, in turn, into stories.*

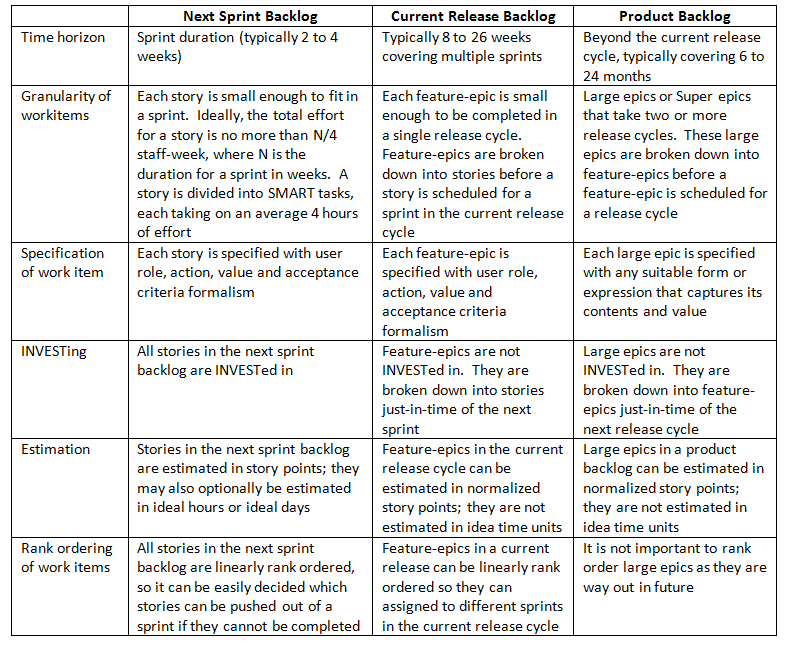
*DIVE the product backlog carefully*

*There is rarely enough time or resources to do everything. Therefore, agile teams must prioritize (rank-order, to be more precise) which stories to focus on and which lowest rank-order stories could be pushed out of scope when close to the end of a sprint. For agile development projects, you should linearly rank-order the backlog, rather than do coarse-grain prioritization where stories and epics are lumped into a small number of priority buckets, such as Low, Medium, High, Critical priorities. Linear rank ordering (i.e., 1, 2, 3, 4 ….n) avoids inflation of priority, keeps everyone honest, and forces decisions on what is really important. It discourages the “kid-in-a-candy-shop” behaviour when the business side clamours that everything is of high-priority or of equal importance.*

*Note that epics and stories are conceptually different, and should not be mixed or aggregated while developing a rank order. An epic rank order is separate from a story rank order.*

*The responsibility of agile rank ordering is shared among all members of a team; however, the rank ordering effort is led by the product owner. Similar to DEEP, INVEST and SMART, DIVE is a meaningful English word, and also an acronym. Product backlog items should be linearly ordered based on the DIVE criteria, which requires careful consideration of all four factors captured in the DIVE acronym:*

* *Dependencies: Even after minimizing the dependencies among stories or epics (which is always a good thing to do), there may still be few unavoidable dependencies and they will have an impact on rank ordering. If Work-item A depends on B, B needs to be rank-ordered higher than A.*
* *Insure against Risks: Business as well as technical risks*
* *Business Value*
* *Estimated Effort*



# 5. PRODUCT BACKLOG: GOALS GRANURALITY

|  |  |
| --- | --- |
| Goal-ID-1 | Build Player Profile |
| Purpose | To populate the whole database for future analysis.  The aim is to collect player data. It will be done by using web-scraping or public datasets. |
| Target Audience | Stakeholders |
| Status | On-going |
| Task Description | 1. Ascertain Data Correctness -S |
|  | 2. Filter Data Sources -S |
|  | 3. Identify appropriate DBMS -S |
|  | 4. Analyse Data -S |
|  | 5. Extract Appropriate Data -S |
|  | 6. Insert extracted data into resource database -R |
|  | 7. Clean Player Data Abnormalities -R |
|  | 8. Find Data Sources -R |
|  | 9. Create Cloud Backup of Database -R |
|  | 10. Segregate Databases -R |

|  |  |
| --- | --- |
| Goal-ID-2 | Classify Players |
| Purpose | Make groups in the database according to player’s attributes to help distinguish them. It will help obtain a better prediction. Also, will make it easier to search players. |
| Target Audience | Stakeholders |
| Status | On-going |
| Task Description | 1. Research Market Terminology S |
|  | 2. Finalise Classification Classes S |
|  | 3. Use classes to improve Value Predictions S |
|  | 4. Use classes to Improve Searchability S |
|  | 5. Remodel Database S |
|  | 6. Analyse Datasets R |
|  | 7. Consider Database Limitations R |
|  | 8. Run Test Classifications R |
|  | 9. Resolve Classification Overlaps R |
|  | 10. Maintain Fluidity in Classification R |

|  |  |
| --- | --- |
| Goal-ID-3 | Process Player Profiles |
| Purpose | To learn from the data to generalize on what attributes from the profile contribute to player value. This will be used to build a statistical model. |
| Target Audience | Stakeholders |
| Status | On-going |
| Task Description | 1. Pre-Process Player Data -S |
|  | 2. Train Statistical Model -S |
|  | 3. Generate Player Features -S |
|  | 4. Test Statistical Model -S |
|  | 5. Save Trained Model -S |
|  | 6. Optimise Choices -R |
|  | 7. Choose Statistical Model -R |
|  | 8. Tune statistical model hyper-parameters -r |
|  | 9. Associate Player Profiles -R |
|  | 10. Probe Statistical Models -R |

|  |  |
| --- | --- |
| Goal-ID-4 | Process User Queries |
| Purpose | To predict Player’s Value using Regression Models and append new Player’s Data given by End User |
| Target Audience | Customers |
| Status | On-going |
| Task Description | 1. Filtering relevant data from database |
|  | 2. Making a data structure to fit all relevant data |
|  | 3. Normalising the data structure according to the model |
|  | 4. Predicting value using the model |
|  | 5. Scale the value accordingly |
|  | 6. Appending value in database |
|  | 7. Form filling by end user |
|  | 8. Validating Player’s Information |
|  | 9. Give warning for anomalies |
|  | 10. Append Player’s Data |

|  |  |
| --- | --- |
| Goal-ID-5 | Demonstrate Player Statistics |
| Purpose | To do Data Analysis and Visualization of available data |
| Target Audience | Customers |
| Status | On-going |
| Task Description | 1. Locate player in database |
|  | 2. Retrieve Player Statistics |
|  | 3. Showcase important information about player |
|  | 4. Demonstrate the performances of the player |
|  | 5. Contract details of player |
|  | 6. Show rankings of player compared to other players |
|  | 7. Detailed reports on player’s performance |
|  | 8. Graphs visualizing player details |
|  | 9. Showing net worth graph of player |
|  | 10. Recommendation of other players |

|  |  |
| --- | --- |
| Goal-ID-6 | Estimate Player Value |
| Purpose | To give an idea about the player’s market value. Helps the teams to analyse their strategy. This the prime sellable feature of the product. |
| Target Audience | Customers |
| Status | On-going |
| Task Description | 1. Check Player Data |
|  | 2. Fetch Player Data |
|  | 3. Feed data to the model |
|  | 4. Generate predictive Value |
|  | 5. Get predictive Value from Model |
|  | 6. Display calculated value |
|  | 7. Process and Transmit to Client |
|  | 8. Update Player Data |
|  | 9. Store Player Data |
|  | 10. Display important data features |

|  |  |
| --- | --- |
| Goal-ID-7 | Process Feedback |
| Purpose | To correct and cross validate the system. It helps in improving the system. Necessary so that the system stays in touch with the real parameters and gives accurate predictions. |
| Target Audience | Stakeholders |
| Status | On-going |
| Task Description | 1. Get feedback from user |
|  | 2. Process Feedback |
|  | 3. Determine Feedback Response |
|  | 4. Analyse Required Changes |
|  | 5. Finalise the necessary changes |
|  | 6. Get it approved from the team |
|  | 7. Apply the necessary changes |
|  | 8. Formulate the software |
|  | 9. Re-Release Software |
|  | 10. Re-Evaluate Regularly |

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 7

**Player Value Analyser**

**User Story Cards**

**13-11-2017**

**Version 1.0**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By:**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **66** |
| 2 | Epics and User Stories | **66** |
| 3 | User Stories-Goal 1 | **68** |
| 4 | User Stories-Goal 2 | **70** |
| 5 | User Stories-Goal 3 | **72** |
| 6 | User Stories-Goal 4 | **74** |
| 7 | User Stories-Goal 5 | **76** |
| 8 | User Stories-Goal 6 | **79** |
| 9 | User Stories-Goal 7 | **81** |

# INTRODUCTION

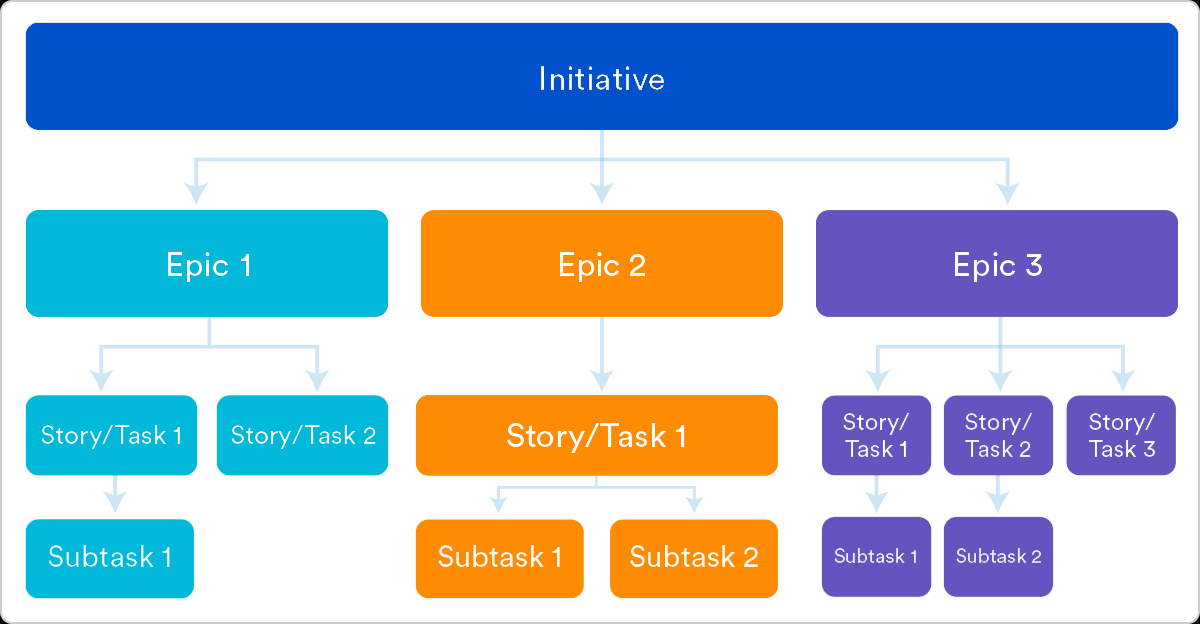
*What does defining customer problems look like in an agile world? The agile manifesto reminds us that we don’t always have to do it the “traditional” way. As product managers, we should be doing whatever is required to tell the story of the customer. Try different things: experiment, explore, then do what works best for you and your team in the context that you might be working in.*

* *If it means you can have several discussions and sketch something on a bit of paper – then do it.*
* *What if you could get everyone (including the customer) in a room and do a user story mapping exercise? If that communicates the problems well, then you don’t need to go much further.*
* *Or what if you can visit the customer and watch them use your product in context? Could you get your engineers and designers to sit next to the customer to listen to and observe their problems?*
* *Instrumenting your product with analytics hooks give you aggregate, concrete data about how customers as a whole are using your product.*
* *Another option would be to grab the product triad (a product manager, engineer and a designer) for a quick stand-up to sketch, discuss and make some quick decisions on the spot.*
* *Need to explore some more? Try running a workshop where you gather key stakeholders and do lots and lots of white-boarding or even paper prototyping to dive deep into understanding the problems you are trying to solve and how you could solve those problems.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Epic** Large body of work, contains stories | **Story** Smallest unit of work, also known as a task | **Version** The release of software to the customer | **Sprint** Iteration where team does the work |

# EPICS AND USER STORIES

*Epics are larger bodies of work that stories roll up into. An epic can span across multiple sprints and versions. Versions are different from epics, because they are a point in time where software is released to the customer. A version might contain multiple epics. Epics help teams create hierarchy and structure. Stories help teams keep track of specific details for the task at hand and can be broken down into sub-tasks.*



* *An* ***epic*** *is a large body of work that can be broken down into a number of smaller stories. For example, performance-related work in a release. An epic can span more than one project, if multiple projects are included in the board to which the epic belongs.*
* *Unlike sprints, epics often change in scope over time as a natural aspect of agile development. Epics are almost always delivered over a set of sprints. As a team learns more about an epic through development and customer feedback, user stories will be added and removed to optimize the team's release time.*
* ***Burndown******charts*** *can also be used to visualize epics, which keep teams motivated and the executive stakeholders informed. A good epic burndown chart shows the agile nature of development. It's clear how the team is progressing as well as where the product owner added and removed user stories. Having these data points clearly visible keeps everyone on the same page and facilitates open conversation about the evolution of the product and completion forecasts. Not to mention that transparency builds trust!*
* *A story or* ***user story*** *is the smallest unit of work in an agile framework. It is a software system requirement that is expressed in a few short sentences, ideally using non-technical language.*
* *The goal of a user story is to deliver a particular value back to the customer. Note that "customers" don't have to be external end users in the traditional sense, they can also be internal customers or colleagues within your organization who depend on your team.*
* ***User stories*** *are a few sentences in simple language that outline the desired outcome. They don't go into detailed requirements.*
* ***Versions*** *are the actual releases of software out to customers. Remember, at the end of each sprint the team should be able to ship the software to customers. Versions are the curated changes the product owner actually ships.*
* ***Versions*** *are often developed over a set of sprints, much like epics. Savvy product owners may choose to deliver an epic over several versions. An epic does not have to be fully contained within a version. By delivering an epic over several versions, the product owner can learn how the market is responding to that epic and make calculated decisions about its future direction rather than doing one giant release.*
* *A* ***sprint*** *is a short period in which the development team implements and delivers a discrete and potentially shippable application increment, e.g. a working milestone version. If you haven't run sprints before, we recommend using a fixed two-week duration for each sprint. It's long enough to get something accomplished, but not so long that the team isn't getting regular feedback.*
* *In* ***scrum****, teams commit to complete a set of user stories during a fixed time period. Generally speaking, sprints are one, two, or four weeks long. It's up to the team to determine the length of a sprint. Once a sprint cadence is determined, the team perpetually operates on that cadence. Fixed length sprints reinforce estimation skills and enable the ability to predict the future* ***velocity*** *for the team once they have the data from several completed sprints.*

*Once a team commits to a set of user stories for the sprint, and the sprint is started, the scrum master is in charge of fending off changes to the user stories. This keeps the team focused and combats "s****cope creep****" (adding work to the sprint after the sprint starts). Adding work mid-sprint compromises the team's ability to forecast and estimate accurately.*

*At the end of each sprint, the team is required to deliver a working piece of software. In scrum, that's called a* ***potentially shippable increment*** *(PSI). The product owner ultimately decides when the PSI gets released to customers, but the work should be complete enough to be suitable for release at the end of the sprint.*

*In agile development,* ***work in progress*** *(WIP) limits set the maximum amount of work that can exist in each status of a workflow. Limiting the amount of work in progress makes it easier to identify inefficiency in a team's workflow. Bottlenecks in a team's delivery pipeline are clearly visible before a situation becomes dire.*

# USER STORIES: GOAL-1: BUILD PLAYER PROFILE

|  |  |  |
| --- | --- | --- |
| Objective-1 | Acquire Player Data | |
| Purpose | The purpose is to develop Player Profiles to enable the system to analyse and visualise the data. | |
| Target Audience | Developer | |
| Status | Completed | |
| Role: | **As a**developer | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Request creation of Player Profile | use it to do further analytics. |
|  | 2. Find Player Data Sources | build player profiles. |
|  | 3. Create a preliminary database | can store the acquired player data. |
|  | 4. Create a Player Data Spyder | get updated player data. |
|  | 5. Formulate database structure | start creating profiles. |
|  | 6. Populate Player Database | meet the preliminary objective. |
|  | 7. Generate a backup | retrieve data in case of loss of files. |
|  | 8. Share backup with Project Team | expect team to perform assigned tasks. |
|  | 9. Assign database privileges | monitor the changes made to the database. |
|  | 10. Launch Player Profile Page | fulfil project deliverables. |

|  |  |  |
| --- | --- | --- |
| Process-1 | Acquire Player Statistics | |
| Purpose | Collect Player Statistics for creating player ranking index used to find transfer values. | |
| Target Audience | Internal Stakeholders | |
| Status | Completed | |
| Role: | **As a**developer | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Set up a mandatory field set | maintain consistency of database. |
|  | 2. Research Player statistics | make system as accurate as possible. |
|  | 3. Set statistics’ fields in database | add player statistics. |
|  | 4. Accept Player Profile inputs | add new players. |
|  | 5. Resister Player Profiles into database | store new players. |
|  | 6. Set Statistic limits and bounds | keep the data relative and realistic. |
|  | 7. Populate Player statistics | compare players. |
|  | 8. Decide appropriate sorting for database | create a relative rating index. |
|  | 9. Update Database structure | accommodate statistics in player profiles. |
|  | 10. Create log file | keep track of changes made. |

|  |  |  |
| --- | --- | --- |
| Process-2 | Clean Data Abnormalities | |
| Purpose | To keep the data relative and precise. | |
| Target Audience | Developer | |
| Status | On-going | |
| Role: | **As a**developer | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Validate Player Profile Format | verify that database is consistent. |
|  | 2. Decide attribute ranges | compare abnormalities of data |
|  | 3. Plot graph of all values | sieve abnormalities. |
|  | 4. Search for spikes in graph | identify data abnormalities. |
|  | 5. Investigate searched abnormality | verify the legitimacy. |
|  | 6. Produce the correct, improvised data | resolve the abnormality issue. |
|  | 7. Correct the found abnormality | refine the data. |
|  | 8. Commit changes on database | render the changes to the team. |
|  | 9. Check if data has been already used | rollback the incorrect data operations. |
|  | 10. Notify team about changes | ensure consistency in the system. |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Analyse Player Data | |
| Purpose | To decide the Player analysis process of the system and the methodology to follow. | |
| Target Audience | Internal Stakeholders | |
| Status | Completed | |
| Role: | **As a**developer | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Organise database attributes | easily analyse the data. |
|  | 2. Design patterns for attributes | study the data. |
|  | 3. Conceptualise output parameters | process of analysing is directed. |
|  | 4. Prioritise the important parameters | produce accurate results. |
|  | 5. Draw a map for the shortlisted attributes | formalize the observations. |
|  | 6. Organise the parameters | simplify analysis process. |
|  | 7. Record formulated observations | discuss with the analysis team. |
|  | 8. Correspond with Analysis team | refine the observations |
|  | 9. Consolidate outline of analysis process | systemize procedure. |
|  | 10. Construct final analysis methodology | begin development process. |

|  |  |  |
| --- | --- | --- |
| Process-1 | Filter Relevant Data | |
| Purpose | The purpose is to get detailed, relevant data about player which is filtered and curated. | |
| Target Audience | Customers | |
| Status | On-going | |
| Role: | **As an**end user | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Find a certain player | view player relevant data |
|  | 2. Attain transfer value of player | report the value. |
|  | 3. Get a curated list of players | compare the players according to my needs. |
|  | 4. Find similar players playing in same position | compare player quality. |
|  | 5. Prioritize players by rating | get best deal for the club. |
|  | 6. Consider all players with comparable price | compare player price. |
|  | 7. Filter stats indicating on field behaviour | decide attitude of player. |
|  | 8. Filter stats indicating off field characteristics | popularity of player. |
|  | 9. Display player’s current team | know the chance of buying. |
|  | 10. View player’s current team’s squad | decide importance of player. |

|  |  |  |
| --- | --- | --- |
| Process-2 | Ascertain Data Correctness | |
| Purpose | This will ensure the reliability and correctness of system. | |
| Target Audience | Customers | |
| Status | Completed | |
| Role: | **As an**end user, developer | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Get correct player data | Use it in news reports |
|  | 2. Receive exact information | Cite the website as source. |
|  | 3. Ensure appropriate info of the player | Protect the player that I represent. |
|  | 4. Use Player Data | Ask Club to scout player. |
|  | 5. Cite transfer value | Get an estimate of the player’s quality. |
|  | 6. Get correct player value prediction | To negotiate with club. |
|  | 7. Make a precise database | Use it to display on third party portals. |
|  | 8. Track validation process | Verify data. |
|  | 9. Run background checks | Ascertain data correctness. |
|  | 10. Know the data sources | Trust the system. |

# USER STORIES: GOAL-2: CLASSIFY PLAYERS

|  |  |  |
| --- | --- | --- |
| Objective-1 | Extract Player Information | |
| Purpose | Make groups in the database according to player’s attributes to help distinguish them. | |
| Target Audience | External Stakeholders | |
| Status | On-going | |
| Role: | **As a**football club director | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. find and add large number of players | Make player recruitment easier. |
|  | 2. maintain all my players in database | Easily find the squad. |
|  | 3. extract player information | Use it for the signing of new players. |
|  | 4. find information of club squad | Keep track of player progression. |
|  | 5. diagnose another team’s squad | To prepare my club for facing other teams. |
|  | 6. analyse player form | Make decision on contract of player. |
|  | 7. survey market values | Negotiate new contracts and transfer fees. |
|  | 8. investigate new players | Ascertain my transfer target’s price. |
|  | 9. establish player background | Expect transparency. |
|  | 10. administer information extraction | Put system to good use. |

|  |  |  |
| --- | --- | --- |
| Process-1 | Extract Player Position | |
| Purpose | It will make it easier to search players. | |
| Target Audience | External Stakeholders | |
| Status | On-going | |
| Role: | **As a**Football Club Coach | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. play a team with certain formation | Help the team perform better. |
|  | 2. find players according to position | Play my preferred team formation. |
|  | 3. search players that can play in certain position | Sign them in my team. |
|  | 4. check similar players | Decide best solution. |
|  | 5. survey player past experience | Compare player quality. |
|  | 6. diagnose player playing style | Check compatibility with team. |
|  | 7. extract player rating | Compare with existing players. |
|  | 8. investigate player value | Decide transfer budget. |
|  | 9. analyse bid | Notify club about player. |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Process-2 | Extract Player Ability | |
| Purpose | It will help obtain a better prediction. | |
| Target Audience | Customers | |
| Status | Completed | |
| Role: | **As a**user | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. know the skill attributes of a player | Assess the quality of a player |
|  | 2. find club squad | Follow the football club’s activities |
|  | 3. determine club results | Compile a report on a club’s progress |
|  | 4. survey player form | Give feedback of player. |
|  | 5. compare transfer prices | Compare players from different countries. |
|  | 6. Extract player skill values | Find out player strengths. |
|  | 7. find player weaknesses | Be cautious when initiating player transfer. |
|  | 8. find player injury record | Be aware of physical condition of player. |
|  | 9. compare player abilities | See head to head matchup. |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Classify Players | |
| Purpose | To make classes of players as per categories. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a**Developer | |
|  | **I want to** *<perform some task>* | **I want to** *<perform some task>* |
| Task Description | 1.Acquire player attributes and data | Identify player data |
|  | 2.Examine data distribution per attribute | View relationships between the data |
|  | 3.Fix inconsistencies in selected attributes | Ensure correct data is used in future processes. |
|  | 4.Prototype groups according to attributes | Select viable grouping attributes |
|  | 5.Establish player groups | Enable grouped data usage in further processes |
|  | 6.Access player groups | View grouped data |
|  | 7.Form a basis/gist for the group | The usage of the group is enabled |
|  | 8.Choose encoding method for the basis | Encode data appropriately |
|  | 9.Establish results of encoding analysis | Forward it to developers |
|  | 10.Integrate results | They can be used in modelling. |

|  |  |  |
| --- | --- | --- |
| Process-1 | Group Players | |
| Purpose | Decide attributes that can decide classes and groups. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a**Developer | |
|  | **I want to** *<perform some task>* | **I want to** *<perform some task>* |
| Task Description | 1.Acquire Player Data | Identify Player Information |
|  | 2.Identify Player Attributes | View Player Attributes from player database. |
|  | 3.Examine attribute-wise data | Extract attribute-wise player groups. |
|  | 4.Extract attributes for grouping | Extract attribute-wise player groups. |
|  | 5.Detect inconsistencies in selected attributes | Eliminate inconsistencies. |
|  | 6.Normalise and repair inconsistencies | Ensure proper data |
|  | 7.Verify correctness of normalisation | Validate normalisation |
|  | 8.Create player groups according to attribute | Use the grouped data for better statistical model |
|  | 9.Demonstrate player groups | Usage of groups is ensured |
|  | 10.Deliver player groups | Forward groups for further analysis |

|  |  |  |
| --- | --- | --- |
| Process-2 | Verify Player Groups | |
| Purpose | Verify player groups formed. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a**developer | |
|  | **I want to** *<perform some task>* | **I want to** *<perform some task>* |
| Task Description | 1.Access player groups | Use player groups for analysis |
|  | 2.Analyse player groups | Gain insights into the data |
|  | 3.Conceptualise basis of the groups | Verify the insights gained |
|  | 4.Derive basis for grouping | Validate the basis derived |
|  | 5.Decide encoding technique for derived basis | Represent the basis and insights in the database |
|  | 6.Validate encoding technique | Ensure proper representation via encoding |
|  | 7.Analyse encoded data | Verify usability of the encoded data |
|  | 8.Modify encoding technique | Ensure best encoding technique is used |
|  | 9.Establish results of encoding analysis | Represent them and use in further analysis |
|  | 10.Integrate results | Enable usage of results in further analysis and modelling |

# USER STORIES: GOAL-3: Process Player Profiles

|  |  |  |
| --- | --- | --- |
| Objective-1 | Pre-process Player Data | |
| Purpose | Perform pre-processing on data such as: Cleaning, Normalisation, etc. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *<type of user>* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Verify raw player data | ensure data correctness. |
|  | 2.Evaluate player data | identify outliers. |
|  | 3.Dispose irrelevant attributes | eliminate unnecessary information. |
|  | 4.Group and compare attributes | examine data distribution. |
|  | 5.Associate domain knowledge gained | improve performance of statistical methods. |
|  | 6.Represent the domain knowledge features | ensure applicability of statistical methods. |
|  | 7.Standardize player features | ensure equal importance to each feature |
|  | 8.Evaluate data dimensionality | detect problems with high dimensionality |
|  | 9.Use different feature selection strategies | fix dimensionality problems |
|  | 10.Consolidate final data with features | use it to train statistical models |

|  |  |  |
| --- | --- | --- |
| Process-1 | Associate Player Profiles | |
| Purpose | Integrate data player-wise. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Fetch formatted player data | use it for further training process. |
|  | 2.List available raw player attributes | choose attributes to be considered. |
|  | 3.Keep only relevant attributes | eliminate unnecessary information. |
|  | 4.Verify consistency of selected attributes | prevent problems affecting further process. |
|  | 5.Transform attributes | formulate attributes that are more relevant. |
|  | 6.Compare attributes between players | decide what attributes are important. |
|  | 7.Inspect attributes by player positions | analyse importance of player position. |
|  | 8.Compare attributes by age | analyse importance of age. |
|  | 9.Inspect attributes by nationality | decide importance of nationality. |
|  | 10.Associate insights gained with data | decide better statistical models. |

|  |  |  |
| --- | --- | --- |
| Process-2 | Generate Player Features | |
| Purpose | Use the knowledge from earlier analysis to create new features. | |
| Target Audience | Internal Stakeholders | |
| Status | On-going | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Tabulate player data | prototype features to generate. |
|  | 2.Generate features that are valuable | improve the statistical model. |
|  | 3.Apply transformations on the features | get more information from the features. |
|  | 4.Visualize transformed features | verify transformations |
|  | 5.Scale features | ensure valid scaling for statistical methods. |
|  | 6.Compare the scales of the features | verify similar scale in features. |
|  | 7.Examine the features in dataset as a whole | judge the dimensionality of data. |
|  | 8.Evaluate different feature selection strategies | choose appropriate strategy. |
|  | 9.Apply feature selection strategy to the data | avoid curse of dimensionality of data. |
|  | 10.Integrate generated features and methods | apply them to new data in the future. |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Decide Statistical Model | |
| Purpose | Analyse different statistical models. Choose a model appropriate to player data. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *<type of user>* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Access data in proper format | Use it to choose a model. |
|  | 2.Access different statistical models | Utilise them for making predictions. |
|  | 3.Apply a model and store its results | Access the results for further analysis. |
|  | 4.Cross-validate and analyse models | Select them based on their quality. |
|  | 5.Verify and overview analysis | Select optimal model. |
|  | 6.Access selected optimal model parameters | Establish their use. |
|  | 7.Modify the parameters | Analyse the changes caused by them. |
|  | 8.Cross-validate changed parameter results | Choose the optimal parameters. |
|  | 9.Use different scoring methods | Analyse the model and establish optimality. |
|  | 10.Finalize on the model and parameters | Use the model in further processes. |

|  |  |  |
| --- | --- | --- |
| Process-1 | Choose Statistical Model | |
| Purpose | Indicate purpose of the process here in 3/4/ statements. | |
| Target Audience | Internal Stakeholders | |
| Status | On-going | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Access different statistical models | Use them for making predictions. |
|  | 2.Access data in proper format | Feed it to the models. |
|  | 3.Split the data into training and testing sets | Test model properly. |
|  | 4.Use different statistical models | Compare their usefulness. |
|  | 5.Apply a model and store its results | Access them in further assessment. |
|  | 6.Analyse results of different models | Keep only relevant models. |
|  | 7.Perform cross-validation on models | Test their ability to generalize. |
|  | 8.Analyse cross-validation results for models | Select them based on their quality. |
|  | 9.Validate the output of selected models | Ensure their correctness. |
|  | 10.Overview the analysis | Converge on an optimal model for the data. |

|  |  |  |
| --- | --- | --- |
| Process-2 | Optimise Choices | |
| Purpose | Select optimal model for player data. Selection is done by comparing results of models. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Access the optimal model | Study it’s working and improve it. |
|  | 2.Analyse the parameters of the model | Establish their use. |
|  | 3.Change the parameters and see the results | Check if there can be improvement |
|  | 4.Maintain the results with changed parameters | Compare results of various parameters |
|  | 5.Cross-validate changed parameter results | Ensure their ability to generalize |
|  | 6.Analyse the cross-validation result | Select the optimal set of parameters. |
|  | 7.Train the model with optimal parameters | Achieve best performing model |
|  | 8.Test the model | Ensure accuracy. |
|  | 9.Use different scoring methods | Analyse the model outputs. |
|  | 10.Finalize on the model and parameters | Use the model in further processes. |

# USER STORIES: GOAL-4: Process User Queries

|  |  |  |
| --- | --- | --- |
| Objective-1 | Recognise User Queries | |
| Purpose | To recognise user input. | |
| Target Audience | Customers | |
| Status | On-going | |
| Role: | **As a**user | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. find and add large number of players | Make player recruitment easier. |
|  | 2. maintain all my players in database | Easily find the squad. |
|  | 3. extract player information | Use it for the signing of new players. |
|  | 4. find information of club squad | Keep track of player progression. |
|  | 5. diagnose another team’s squad | To prepare my club for facing other teams. |
|  | 6. analyse player form | Make decision on contract of player. |
|  | 7. survey market values | Negotiate new contracts and transfer fees. |
|  | 8.Compare attributes by age | analyse importance of age. |
|  | 9.Inspect attributes by nationality | decide importance of nationality. |
|  | 10.Associate insights gained with data | decide better statistical models. |

|  |  |  |
| --- | --- | --- |
| Process-1 | Standardize Query Format | |
| Purpose | This will standardize user input. | |
| Target Audience | Internal Stakeholders | |
| Status | On-going | |
| Role: | **As a**database manager | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. decide a certain query format | Process queries |
|  | 2. get query parameters | Know which data to retrieve |
|  | 3. know the skill attributes of a player | Assess the quality of a player |
|  | 4. find club squad | Follow the football club’s activities |
|  | 5. determine club results | Compile a report on a club’s progress |
|  | 6. survey player form | Give feedback of player. |
|  | 7. check if all attributes are non-null | Display them |
|  | 8. match data types to output | Use proper format |
|  | 9. pass arguments between interfaces | Access database |

|  |  |  |
| --- | --- | --- |
| Process-2 | Execute Prediction | |
| Purpose | This will enable the output of the system. | |
| Target Audience | Internal Stakeholders | |
| Status | On-going | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Access formatted user queries | Process and get related player data |
|  | 2.Extract player data from query | Use it for prediction |
|  | 3.Validate extracted player data | Ensure it fits model requirements |
|  | 4.Detect missing values | Display input error. |
|  | 5.Normalise input data | Search database |
|  | 6.Feed data to the model | Begin prediction |
|  | 7.Extract output from the model | Process query |
|  | 8.Verify proper output | Validate system |
|  | 9.Communicate the prediction to the user | Meet the objective |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Handle User Input Data | |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. | |
| Target Audience | Customers | |
| Status | On-going | |
| Role: | **As a** *end user* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Access all player profiles | Use all player data |
|  | 2. Acquire all player data | Remove anomalies from the data |
|  | 3. Validate data | Erase all the errors from data |
|  | 4. Correct the errors in the data | Generate value with ease |
|  | 5. View all attributes of the player | Give points to the attributes |
|  | 6. Convert attributes into points | Provide data to the system |
|  | 7. Feed data to the model | Use the data in the formula |
|  | 8. Create a formula to generate predicted value | Generate an estimate value |
|  | 9. Generate predicted value of the player | Display it to the user |
|  | 10.Display generated value | Provide the users a correct estimate value |

|  |  |  |
| --- | --- | --- |
| Process-1 | Pre-Process User Input Data | |
| Purpose | This will pre-process user input data. | |
| Target Audience | Internal Stakeholders | |
| Status | On-going | |
| Role: | As adeveloper | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Verify raw player data | ensure data correctness. |
|  | 2.Evaluate player data | identify outliers. |
|  | 3.Dispose irrelevant attributes | eliminate unnecessary information. |
|  | 4.Group and compare attributes | examine data distribution. |
|  | 5.Associate domain knowledge gained | improve performance of statistical methods. |
|  | 6.Represent the domain knowledge features | ensure applicability of statistical methods. |
|  | 7.Standardize player features | ensure equal importance to each feature |
|  | 8.Evaluate data dimensionality | detect problems with high dimensionality |
|  | 9.Use different feature selection strategies | fix dimensionality problems |
|  | 10.Consolidate final data with features | use it to train statistical models |

|  |  |  |
| --- | --- | --- |
| Process-2 | Append Relevant Dataset | |
| Purpose | Indicate purpose of the Process here in 3/4/ statements. | |
| Target Audience | Customers | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Acquire player profiles | View profiles |
|  | 2. Access player profiles | View all player data |
|  | 3. Acquire Player Data | Work on removing anomalies |
|  | 4. Access the player Data | Validate it |
|  | 5. Feed Data to the model | So that anomalies are removed |
|  | 6. Compare Player Data | Know the errors |
|  | 7. View all the attributes | View all data |
|  | 8. Segregate Important attributes | Access important data |
|  | 9. Validate Player Data | Calculate estimate value |
|  | 10. Edit Player profiles | Display correct data |

# USER STORIES: GOAL-5: Demonstrate Player Statistics

|  |  |  |
| --- | --- | --- |
| Objective-1 | Generate Player Statistics | |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Generate Performance Score | Get player’s Quality |
|  | 2.Generate Chemistry Score with other players | Make a better team |
|  | 3.Find Goals Scored | judge his forward performance |
|  | 4.Check skills | tell how stylish he is |
|  | 5.Find popularity | See how well team will get famous |
|  | 6.Tell statistics | Show Quantitative Scores |
|  | 7.Use Past Achievements | Show Highlights |
|  | 8.Check Player’s previous rating | See Present performance |
|  | 9.Check Net Worth | Adjust Base price |
|  | 10.Collect other Misc. data | Show intricate data |

|  |  |  |
| --- | --- | --- |
| Process-1 | Find Player Statistics | |
| Purpose | Indicate purpose of the process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *Developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Collect data using scraping | Get general data for each player |
|  | 2.Find Performance Score | Get player’s Quality |
|  | 3.Find best friends | So that manager can look at them also |
|  | 4.Check Skill he has | Show his style shots |
|  | 5.Check All non-basic details | Show more details about him |
|  | 6.Write more details about him | See his news |
|  | 7.Show best details | Attract Managers |
|  | 8.Use social network | Show his social life |
|  | 9.Download lots of picture and video | Show it to managers |
|  | 10.Find his fan following | See how good he is |

|  |  |  |
| --- | --- | --- |
| Process-2 | Communicate Relevant Statistics | |
| Purpose | Indicate purpose of the Process here in 3/4/ statements. | |
| Target Audience | Customers | |
| Status | On-going | |
| Role: | **As a** *User* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Generate Player Statistics Use it for modelling player profiles | 1. Generate Player Statistics Use it for modelling player profiles |
|  | 2. Find player Statistics Sort and categorize data | 2. Find player Statistics Sort and categorize data |
|  | 3. Communicate Relevant statistics Remove unnecessary statistics | 3. Communicate Relevant statistics Remove unnecessary statistics |
|  | 4. Sort basic player data Categorize Accordingly | 4. Sort basic player data Categorize Accordingly |
|  | 5. Make different categories of players Search Player statistics easily | 5. Make different categories of players Search Player statistics easily |
|  | 6. Display relevant Player statistics Display in brief the Player statistics | 6. Display relevant Player statistics Display in brief the Player statistics |
|  | 7. Display various statistical graphs Understand player statistics more easily | 7. Display various statistical graphs Understand player statistics more easily |
|  | 8. Include Player photos and videos Watch and analyse player’s performance | 8. Include Player photos and videos Watch and analyse player’s performance |
|  | 9. Recommend other similar players View different but similar players | 9. Recommend other similar players View different but similar players |
|  | 10. Display player profiles View all statistics | 10. Display player profiles View all statistics |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Display Player Statistics | |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. | |
| Target Audience | Customers | |
| Status | On-going | |
| Role: | **As a** *Developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.make Bar graph | Show his basic stats |
|  | 2.make Pie graph | Show his shares in a team |
|  | 3.make scatter graph | Show his different goal Streaks |
|  | 4.make deviation graph | Show his variation |
|  | 5.make growth chart | Show his future Goal |
|  | 6.show his downfalls | Tell his bad decision |
|  | 7.show ranking | Tell how good he is |
|  | 8.show his milestones | Tell manager what he has done |
|  | 9.show his achievements | Make him more attractive |
|  | 10.show his Misc. data | Show his nonprofessional things |

|  |  |  |
| --- | --- | --- |
| Process-1 | Choose Relevant Statistics | |
| Purpose | Indicate purpose of the process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *Developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Generate Player Statistics | Use it for modelling player profiles |
|  | 2. Find player Statistics | Sort and categorize data |
|  | 3. Communicate Relevant statistics | Remove unnecessary statistics |
|  | 4. Sort relevant Player data | Categorize Accordingly |
|  | 5. Remove data abnormalities | Display only correct data |
|  | 6. Make different categories of data | Search Player statistics easily |
|  | 7. Consider only relevant player statistics | Remove all unnecessary statistics |
|  | 8. Add statistics to Player profile | Display Player Profile |
|  | 9. Update player statistics | Include all the latest statistics |
|  | 10.Display statistics within profiles | Update Player profiles to display |

|  |  |  |
| --- | --- | --- |
| Process-2 | Generate Statistical Graphs | |
| Purpose | Indicate purpose of the Process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *Developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Generate Player Statistics | Use it for modelling player profiles |
|  | 2. Find player Statistics | Sort and categorize data |
|  | 3. Communicate Relevant statistics | Remove unnecessary statistics |
|  | 4. Sort relevant Player data | Categorize Accordingly |
|  | 5. Categorize player statistics | Search Player statistics easily |
|  | 6. Choose relevant Player Statistics | Remove all unnecessary statistics |
|  | 7. Arrange data suitable for creating graphs | Create statistical graphs easily |
|  | 8. Generate statistical graphs | Add them to player profiles |
|  | 9. Add graphs to player profiles | Display them within the profiles |
|  | 10. Display player profiles | View all statistics |

# USER STORIES: GOAL-6: Estimate Player Value

|  |  |  |
| --- | --- | --- |
| Objective-1 | Validate Player Data | |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Check his basic details | Append in database if its rights |
|  | 2.Scape player’s data from respected source | Append in database for more info |
|  | 3.Check details filled by End user | Be sure about the data |
|  | 4.Delete all anomalies | Have clean database |
|  | 5.Call database admin for big mistakes | Have solution for those entries |
|  | 6.Append each player category in their own table | Have faster searching |
|  | 7.Archive all player who are not playing | Clear Clutter |
|  | 8.Make a good structure | Parse data easily |
|  | 9.Make database normalised | Have less redundant data |
|  | 10.Not let unauthorised person access database | Have data Safety |

|  |  |  |
| --- | --- | --- |
| Process-1 | Fetch Player Data | |
| Purpose | Indicate purpose of the process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Take data from End User | Have one more source for player’s data |
|  | 2.Take data from web scraping | Have reliable data source |
|  | 3.Append data by admin | Have initial database |
|  | 4.Take data from dataset repositories | Have huge database for head start |
|  | 5.Parse data from different API | Have faster data gathering |
|  | 6.Ask FIFA officials for rating | Use those ratings in database |
|  | 7.Use Social Network for misc. data | Have Informal data about players |
|  | 8.Use news details for new discoveries | Use them for highlights |
|  | 9.USE FIFA RECORDS | Show Milestones of player |
|  | 10.Have a forum | Get data from untrusted sources |

|  |  |  |
| --- | --- | --- |
| Process-2 | Feed Data Model | |
| Purpose | Indicate purpose of the Process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *Developers* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Insert basic details in main table | Use it frequently |
|  | 2.Insert Players in different category | Sort them out easily |
|  | 3.Insert rankings in another table | Show manager player’s skill |
|  | 4.Insert links about player | User can see them for more info |
|  | 5.Insert picture, videos in database | Show user who is the player |
|  | 6.Update Current details in database | Use the newest numbers |
|  | 7.Use only stats which will affect prediction | Have better prediction |
|  | 8.Have highest R square value for the data model | Have Highest accuracy |
|  | 9.Make confusion matrix for the predicted values | See how close they are to the real value |
|  | 10.Use Dimensional reductionist | Delete irrelevant data from prediction |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Generate Predicted Value | |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *Developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Use relevant data only for prediction | Have better prediction |
|  | 2.Calculate R square for each data model | Choose highest R square model |
|  | 3.Scale the data | Have Normalised result |
|  | 4.Take care of categorised data | Have them in the prediction model |
|  | 5.Have only good data in the set | Get good predictions |
|  | 6.Train the model | See how values are generated |
|  | 7.Use different regression models | Choose the best result giving model |
|  | 8.Predict the test values using model | Check for deviation |
|  | 9.Calculate Confusion matrix | See accuracy of model |
|  | 10.Graph the predictions | Visualise real values and predicted values |

|  |  |  |
| --- | --- | --- |
| Process-1 | Communicate Player Value | |
| Purpose | Indicate purpose of the process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *User* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Make a new table only for values | Use them for main purpose of the system |
|  | 2.Show all details of players on page | Make manager take decision |
|  | 3.Display contracts of players | Show all request of player |
|  | 4.Show all terms asked by the player | Tell user about the demands of player |
|  | 5.Tell contract duration | Make user think about this player |
|  | 6.Show net worth of player | Make managers discuss about this |
|  | 7.Show base price of player | Show initial bid of the player |
|  | 8.Show current price of player | Tell manager about current value of player |
|  | 9.Show predicted price of player till window | Tell manager about price at which to buy |
|  | 10.Show other similar players and their values | Make manager think about other options also |

|  |  |  |
| --- | --- | --- |
| Process-2 | Display Predicted Value | |
| Purpose | Indicate purpose of the Process here in 3/4/ statements. | |
| Target Audience | Stakeholders | |
| Status | On-going | |
| Role: | **As a** *User* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.See impressive GUI | See everything clearly |
|  | 2.See all stats of player | Compare him with others |
|  | 3.See basic details of player | Know all basic things about him |
|  | 4.See videos of him | See how he plays |
|  | 5.See all the values of him | Take a decision of buying him |
|  | 6.check all contract details | Tell my club about him |
|  | 7.See similar players like him | Think of choosing other better players |
|  | 8.See bidding market on site | Bid for player for a better price |
|  | 9.See transfer’s window date | Know when to bid for the best price |
|  | 10.See confirmation of Players transfer | Do final deals according to the site |

# USER STORIES: GOAL-7: Process Feedback

|  |  |  |
| --- | --- | --- |
| Objective-1 | Generate feedback mechanism | |
| Purpose | It is for taking a feedback from the user regarding the system. Helps in determining if there are bugs or if any improvements can be made. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Create form for feedback | Get user feedback for the system |
|  | 2. Acquire user feedback | Access the feedback |
|  | 3.Access user feedback | Store the feedback for further use |
|  | 4. Store user feedback | Process the queries |
|  | 5. Process feedback | Determine the next steps |
|  | 6. Determine feedback response | apply changes in system |
|  | 7. Apply necessary changes | Fix bugs or make necessary improvements |
|  | 8.Update the system | Make it read for release |
|  | 9.Re-release software | Provide updates system for use |
|  | 10. Generate a feedback mechanism | Receive feedback from users |

|  |  |  |
| --- | --- | --- |
| Process-1 | Assess User feedback | |
| Purpose | To know if the feedback given by the user is relevant and if the changes to be made are necessary to the system. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Generate a feedback mechanism | Take action on the feedback given by the user |
|  | 2.Create a from for feedback | Get user feedback regarding the system |
|  | 3.Acquire User feedback from the form | Store it and have access to it |
|  | 4.Store user feedback | Check it and take necessary steps |
|  | 5. Check if feedback is valid | Know what further steps are to be taken |
|  | 6.Validate feedback | Know if changes suggested are important |
|  | 7.Check if the changes in the given feedback are necessary | Understand if changes are to be made or not |
|  | 8. Assess user feedback | Know what changes are to made |
|  | 9. Make changes if necessary | Keep system updates |
|  | 10. Update the system | Re-release the software with necessary changes |

|  |  |  |
| --- | --- | --- |
| Process-2 | Process User Feedback | |
| Purpose | Helps to determine whether the feedback is genuine and if the changes are necessary to be made. It helps in keeping the system updated. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Generate a feedback mechanism | Take action on the user feedback |
|  | 2.Create a form for feedback | Get user feedback regarding the system |
|  | 3.Acquire User feedback | Store it and have access to it |
|  | 4.Have access to the feedback | Check it and take necessary steps |
|  | 5.Check the feedback | Know what further steps are to be taken |
|  | 6.Validate user feedback | Know if changes suggested are important |
|  | 7.Check if changes are necessary | Understand if changes are to be made. |
|  | 8.Finalise the changes | Get it approved by the rest of the team |
|  | 9.Get it approved by the team | Make necessary changes to the system |
|  | 10.Process the feedback | Keep the system updated |

|  |  |  |
| --- | --- | --- |
| Objective-2 | Apply improvement steps | |
| Purpose | This will help to update the software with the right changes needed. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Create feedback form | Receive feedback from the user |
|  | 2.Get feedback from the user | Access, it and apply relevant changes |
|  | 3.Access the user feedback | Process the feedback |
|  | 4.Process the user feedback | Know what response is to be given to it |
|  | 5. Determine feedback response | Understand what changes are to be made |
|  | 6. Analyse changes required | Check if they are necessary |
|  | 7. Finalise the changes to be made | Get it approved from the rest of the team |
|  | 8. Get it approved from the team | Make necessary changes with everybody’s consent |
|  | 9. Make changes in the system | Apply it to the system and update it |
|  | 10. Apply improvement changes | Keep the software up-to-date |

|  |  |  |
| --- | --- | --- |
| Process-1 | Determine Feedback Response | |
| Purpose | Know what is to be done with the feedback. If feedback is genuine changes are made and if not, a different response is made. | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1.Create a feedback mechanism | Take action on the feedback given by the user |
|  | 2.Create a form for generating feedback | Receive feedback from the user |
|  | 3.Acquire the feedback from the user | Store it and apply relevant changes |
|  | 4.Storethe feedback | Access the feedback when required |
|  | 5.Access the feedback | check the user feedback |
|  | 6. Check the feedback | So that it can be validated |
|  | 7. Validate the feedback | Process it for further changes |
|  | 8. Process the feedback | Determine what response is to made |
|  | 9. Determine Feedback response. | Understand what changes are to be made to the system |
|  | 10 Know if changes are to be made | Keep the system updated |

|  |  |  |
| --- | --- | --- |
| Process-2 | Apply necessary changes | |
| Purpose | To modify the system. Keep it updated and free from bugs | |
| Target Audience | Customers/ Stakeholders | |
| Status | On-going/ Completed | |
| Role: | **As a** *developer* | |
|  | **I want to** *<perform some task>* | **so that I can** *<achieve some goal>* |
| Task Description | 1. Generate feedback mechanism | Take action on the user feedback |
|  | 2. Acquire user feedback | Store it and apply relevant changes |
|  | 3. Validate the feedback | Process it for further changes |
|  | 4. Process the feedback | Determine what response is to made |
|  | 5. Determine feedback response | Understand what changes are to be made to the system |
|  | 6. Analyse changes to be made | Finalise them |
|  | 7. Finalise the changes | Get it approved by the rest of the team |
|  | 8. Get it approved from the team | Make the necessary changes in the system |
|  | 9.Apply the necessary changes in the software | Keep the software ready for re-release |
|  | 10.Re-release software | Keep the software updated for use |

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 8

**Player Value Analyser**

**Software Configuration Management**

***21-07-2017***

***Version 1.0***

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Analyst** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **Developer** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **Designer** |

**Approved By: Dr M. R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **86** |
| 2 | Roles and Responsibilities | **86** |
| 3 | Configuration Identification | **88** |
| 4 | Component Specification: Goal-1 | **90** |
| 5 | Component Specification: Goal-2 | **95** |
| 6 | Component Specification: Goal-3 | **99** |
| 7 | Component Specification: Goal-4 | **102** |
| 8 | Component Specification: Goal-5 | **105** |
| 9 | Component Specification: Goal-6 | **108** |
| 10 | Component Specification: Goal-7 | **111** |

# INTRODUCTION

*As identified in the Software Configuration Management (SCM) Plan Standard, the implementation of a formal and structured SCM environment ensures that all Software Development product artefacts are baselined and maintained in a stable environment.*

*This SCM Procedures identifies the procedures that conform to the requirements identified in the SCM Plan Standard. This document is intended to provide a uniform approach to SCM for the software product being developed or modified by projects regardless of location or staffing model. It describes the procedures for managing and controlling the development, delivery, and maintenance of the specific Software Product <Product name>.*

*The SCM Procedures applies to <Product Name> under development or maintenance. It also applies to all documentation products and other project or program initiative documentation that management communicates now or in the future as required to be controlled by SCM procedures. Each project associated with the product will develop work instructions for the implementation of these procedures.*

*The primary audience for this document consists of staff assigned to projects where <Product Name> is within scope are required to implement and apply SCM procedures.*

# ROLES AND RESPONSIBILITIES

*<This section identifies the specific roles and responsibilities as they relate to SCM, each Project will identify the role that will be responsible for the Product. The SCM Manager will create work instruction documents to assist the project team members with the responsibilities within their assigned role. Each Project will identify who is assigned to each role by having one roles and responsibilities table below per project by coping table for each project and pasting directly below the previous table provided.>*

*The table below is a specific list of the personnel who may be members of Project teams and SCM teams along with their assigned roles and responsibilities as they relate to SCM. The Roles defined herein can sometimes be overlapped with other roles and responsibilities depending on the environment. In addition, one person allocated for a specific role as listed below may often have the responsibility of other roles.*

| *<Named Project(s)>* ***Role*** | *<Named Project(s)>* ***Responsibility*** |
| --- | --- |
| *Program Manager/*  *Project Management*  Neeraj Ganu | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions for each Product being worked as part of the Program/Project.* * *Ensures proper execution of the SCM Plan Standard.* * *Oversees the SCM process.* * *Assesses and evaluates all other change requests.* * *Establish appropriate Change Control Board (CCB).* * *Submit CCB baseline information.* * *Identify dependent projects.* * *Establish/revise required artifacts.* * *Creation of SCM Procedures and work instructions for each VA product they are assigned.* |
| *Software Configuration Manager*  Anup Mahindre | * *Educates project team members in SCM “best practices.”* * *Develops and maintains SCM Procedures and work instructions for each VA product they are assigned.* * *Establishes, promotes, and releases baselines.* * *Performs or validates interim and final builds.* * *Prepares release package, release archives and Version Description Documents (VDD).* * *Accountable for instituting the established processes and reporting progress statistics based on change requests.* * *Identifies product baselines as necessary of all products within their assigned Projects.* * *Responsible for SCM audits and necessary status accounting related to the product.* * *Conducts audits at scheduled milestones.* |
| *Development Manager/Leads*  Hiranyey Gajbhiye | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions.* * *Submits build/release requests.* * *Coordinates development activities and assigns tasks.* * *Ensures all SCM Procedures and work instructions are implemented and followed for all software, documentation, and/or any other components for which they are responsible.* * *Ensures all developers’ work within the specified SCM process and related guidelines as specified in the SCM Procedures and work instructions.* * *Attends the CCB meetings and provide technical details, as required.* |
| *Developers/System Administration/Functional/ Technical Analysts/DBAs/System Administration*  Sanket Ostwal | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions.* * *Maintain accurate, detailed information for all assigned change requests (CRs), in the CR database, related to the applicable development detail of the CRs lifecycle.* * *Provide impact analysis reporting for the CCB approved problems or changes, including documentation of suggested solutions to facilitate CCB disposition activities.* * *Documentation of build, release, and installation instructions.* |
| *Software Change Manager*  Anup Mahindre | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions.* * *Governing body for reviewing and approving change requests under the SCM Procedures and work instructions.* |
| *Technical Writer*  Neeraj Ganu | * *Develops technical deliverable documentation to support the software deliverables.* * *Provides editing, formatting, and graphics support for documentation.* * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions.* |
| *Software Quality Assurance Manager*  Anup Mahindre | * *Develops and maintains artifacts following proper version control procedures defined in the SCM Procedures and work instructions.* * *Ensures all SQA Analysts work within the SCM Procedures and work instructions.* * *Verifies that only SCM-approved deliverables are installed into the test environment(s).* * *Ensures that SQA Analysts are always testing from official SCM deliverables.* * *Attends CCB meetings and provides testing details, as required.* * *Reviews status accounting related to the project.* * *Reviews deliverable artifacts.* |
| *Software Quality Assurance Analysts/ Testing Analyst/*  Hiranyey Gajbhiye | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions.* * *Responsible for testing installed releases, as SCM provides releases from development.* * *Update CRs assigned to them according to test activity results.* * *Determines Pass/Fail for each CR scheduled for a release.* * *Opens CRs (defect and or enhancements) for any newly discovered problems during testing.* |
| *Release Manager/ Implementation Team/ EVS/Operations Team/*  Neeraj Ganu | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions document.* * *Coordinates the release and deployment of software to the existing sites and the newly activated sites following SCM Procedures and work instructions.* * *Assures products meet all exit criteria prior to release* * *Assures change control and SCM processes have been followed as defined in the SCM Procedures and work instructions.* |
| *Process Engineer*  Sanket Ostwal | * *Develops and maintains artifacts following proper version control procedures using the SCM Procedures and work instructions.* * *Guides the Team members in following the EPG published process maps.* |

# CONFIGURATION IDENTIFICATION

*<This section describes the Configuration Identification of the Software Product and providing a unique identity to the product, it’s components, and associated documentation, including the definition of appropriate level of identification. In order to identify the configuration item(s)(CI) s that are to be placed under SCM control, the SCM Manager must understand that Configuration Identification is the process of selecting the CIs and the development items subject to Change Control for a product, assigning unique identifiers to them, and recording their functional and physical characteristics in technical documentation.*

*The following items are subject to configuration identification for software products as per the SCM Plan Standard and are to be placed under SCM control:*

* *Products that are delivered to the customer*
* *Designated internal work products, including source code used to generate the deliverable*
* *Commercial off the Shelf (COTS) products*
* *Non-Developmental Items (NDI) products*
* *Tools*
* *Other items that are used in creating and describing these work products, including documentation describing the function and physical requirements and characteristics of the product*

*These items consist of the set of currently approved or conditionally approved technical documentation, source code, executable images, and object files that identify and describe the functional and physical characteristics of the application.*

***Commercial off the Shelf (COTS) products***

*A COTS item is defined as a commercial item that is of a type customarily used by the general public or by non-governmental entities for purposes other than governmental purposes, and:*

* *Has been sold, leased, or licensed to the general public; or has been offered for sale, lease, or license to the general public*
* *Has been sold or offered for sale in substantial quantities in the commercial marketplace*
* *Has been offered to the Government, under a contract or subcontract at any tier, without modification, in the same form in which it is sold in the commercial marketplace*

*COTS items shall be identified within the system configuration by the manufacturers name, item identification, and version in sufficient detail to allow re-acquisition of the identical item. If a COTS item is changed in such a manner that it no longer meets the definition of COTS, the item must be reclassified by its new classification.*

***Non-Developmental Items (NDI) products***

*An NDI is defined as any COTS item that requires only minor tailoring of a type customarily available in the commercial marketplace, and is within the normal function of the COTS item. This tailoring does not include modification or customization beyond what is normally provided in the commercial marketplace and is outside of the provider’s normal pricing structure.*

*NDI items shall be identified within the system configuration by the manufacturers name, item identification, version, and tailoring in sufficient detail to allow re-acquisition of the identical item. If a NDI item is changed in such a manner that it no longer meets the definition of NDI the item must be reclassified by its new classification*

***Modified Item***

*A modified item is defined as a COTS or GOTS item which is customized for a specific purpose and to meet specific requirements beyond the normal function of the COTS or GOTS item is defined as a Modified item*

***Third Party Item***

*A Third-Party Item is defined as a new item or modified item developed by a subcontractor for a specific purpose and to meet specific requirements.*

***Developmental Item***

*A Developmental Item is defined as a new item or modified item developed for a specific purpose and to meet specific requirements.*

|  |  |
| --- | --- |
| COTS USED |  |
| NDI USED | Jupyter Notebook, NumPy, SciPy, Pandas, Scikit-Learn, Net Beans, Bootstrap, AngularJS |
| MODIFIED ITEMS | None |
| THIRD PARTY ITEMS | None |
| DEVELOPMENTAL PRODUCTS | None |

# COMPONENT SPECIFICATION: GOAL-1

|  |  |
| --- | --- |
| ***Component Name*** | **Player Profile Builder** |
| ***Audience*** | Internal Stakeholders |
| ***Responsibilities*** | Building Player Profiles, Analyzing Player Data, Validating Data |
| ***Processing*** | 1. Ascertain Data Correctness 2. Filter Data Sources 3. Identify appropriate DBMS 4. Analyze Data 5. Extract Appropriate Data 6. Insert extracted data into resource database 7. Clean Player Data 8. Find Data Sources 9. Create Cloud Backup of Database 10. Segregate Databases |
| ***Reference*** | Profile Builder |
| ***Constraints*** | Free Player Data Availability Restrictions |
| ***Composition*** | Sub - System 1, Module 1 |
| ***Resources*** | Database 1, 4 tables |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Cleans Data, Checks for Abnormality |

## Procedure Definition Language (Pseudo-code):

INTERFACE: PlayerProfileBuilder interfaces

DO:

mysql\_connect(DATABASE);

connect\_to(NET)

INITIALIZE PYTHON SCRIPT FOR scraping

BEGIN

Find Data Sources

FIND GOOD SITES

RUN scapy.py

data=FETCH DATA()

Parse to XML OR JSON

SEND pydata to our System

GET XML OR JSON in our System

if(data==null):

restart scraping

else:

unparse\_Data()

if(data is in range):

Group all data into category

Identify appropriate DBMS

Insert extracted data into resource database

else:

Group all data into category

Identify appropriate DBMS

Insert extracted data into resource database

Clean Player Data Abnormalities

Make backup of newly added data

Segregate Database

IF( DATA := CALL DATA\_AQUIRER();) THEN

CALL DATA\_ANALYSER(DATA);

ELSE

OUTPUT(DATA SOURCE UNAVAILABLE)

RETURN -1;

END

# COMPONENT SPECIFICATION: GOAL-1 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **Data Acquirer** |
| ***Audience*** | Internal Stakeholders |
| ***Responsibilities*** | State computational abilities here. |
| ***Processing*** | 1. Filter Data Sources 2. Collect Player Data 3. Analyze Player Data 4. Ascertain Data Correctness 5. Determine Data Abnormalities 6. Remove Data Abnormalities 7. Clean Data 8. Use Compatible DBMS 9. Insert Player Data into DBMS 10. Create Cloud Backup of Database |
| ***Reference*** | Data Acquirer |
| ***Constraints*** | RDBMS |
| ***Composition*** | Sub - System 1, Module 1 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 |
| ***Interface/Tasks*** | Acquire Data, Remove Data Abnormalities |

## Procedure Definition Language (Pseudo-code):

INTERFACE: DataAquirer interface

DO

Find Data Sources

foreach(TABLE table in database):

if table is desired:

fetch tht table

BEGIN

GET DATA SCOURCES;

import database libraries

construct prepared Statement

insert query in tht statement

run that query

if(query\_runned)

Insert extracted data into resource database

print done

else

print do this process again

IF(DATA SOURCES AVAILABLE) THEN

CHECK DATA DIMENSIONS;

FILTER DATA

FETCH DATA INTO TEMPORARY DATASET;

CREATE DBMS INSTANCE;

INSERT DATA FROM TEMPORARY DATASET INTO DBMS INSTANCE;

RETURN DATA;

ELSE

RETURN NULL;

END

# COMPONENT SPECIFICATION: GOAL-1 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | Data Analyzer |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | State computational abilities here. |
| ***Processing*** | 1. Determine Data Sources  2. Filter Data Sources  3. Collect Player Data  4. Analyze player Data  5. Filter Appropriate and important Data  6. Segregate Player Data  7. Determine Data Correctness  8. Remove Incorrect Data  9. Clean Data Abnormality  10.Insert Extracted Data into Database |
| ***Reference*** | Data Analyzer |
| ***Constraints*** | Processing Speed |
| ***Composition*** | Sub - System 1, Module 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 |
| ***Interface/Tasks*** | Filter Relevant Data, Ascertain Correctness of Data |

## Procedure Definition Language (Pseudo-code):

INTERFACE RAW\_DATA\_ANALYSER

DO

Ascertain data correctness

Clean Player Data Abnormalities

BEGIN

FOR EACH COLUMN IN DATA LOOP

IF(VALUES OF COLUMN NOT IN EXPECTED RANGE) THEN

REPORT DATA ABNORMALITY;

CLEAN ABNORMAL DATA;

import database libraries

construct prepared Statement

insert query in tht statement

run that query

if(query\_runned)

Insert extracted data into resource database

print done

else

print do this process again

END LOOP;

END

# COMPONENT SPECIFICATION: GOAL-2

|  |  |
| --- | --- |
| ***Component Name*** | **Player Classifier** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Build Player Classes |
| ***Processing*** | 1. Market Terminology 2. Finalize Classification Classes 3. Use Classes to Improve Value Prediction 4. Use classes to Improve Searching 5. Remodel Database 6. Analyze Datasets 7. Consider Database Limitations 8. Run Test Classifications 9. Resolve Classification Overlaps 10. Maintain Fluidity in Classification |
| ***Reference*** | Player Classifier |
| ***Constraints*** | MySQL Database |
| ***Composition*** | Sub - System 1, Module 3 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 5 to 13 |
| ***Interface/Tasks*** | Update Database, Perform Classification of Players |

## Procedure Definition Language (Pseudo-code):

INTERFACE: Classify Players

DO

Build Player Classes

BEGIN

import database libraries

connect to db

If(connencted)

foreach(TABLE table in database):

if table is desired:

fetch tht table

WHILE (All data is not read)

{

PlayerPosition = ExtractPlayerPositionFromFile()

PlayerAbilities= ExtractPlayerAbilitiesFromFile()

if(new Category found)

MAKE new table for new categories()

else

Keep same Database

Verify(PlayerPosition)

Verify(PlayerAbilites)

Group\_dataset(PlayerPosition)

Group\_dataset(PlayerAbilities)

Update\_Database\_Table()

}

Else

Print Connection Error

if(user ask to connect again)

connect again

END IF

END

# COMPONENT SPECIFICATION: GOAL-2 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **Player Information Extractor** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | State computational abilities here. |
| ***Processing*** | 1. Acquire player Data  2. Analyze Player Data  3. Verify Data  4. Store Player information into Database  5. Extract Player data  6. Extract player position  7. Classify players according to position  8. Extract Important Player Attributes  9. Classify Players According to those attributes  10. Display Player information |
| ***Reference*** | ExtractPlayerPositionFromFile |
| ***Constraints*** | MySQL Database |
| ***Composition*** | Sub - System 1, Module 3 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Update Database, Perform Classification of Players |

## Procedure Definition Language (Pseudo-code):

INTERFACE: ExtractPlayerPositionFromFile(Player\_Name)

DO

Extract Player Information

BEGIN

import database libraries

connect to db

If(connencted)

foreach(TABLE table in database):

if table is desired:

fetch tht table

WHILE (Database.PName == Player\_Name) LOOP

Name=ExtractName()

Position=ExtractPosition()

Ranking=ExtractRanking()

LINKS=ExtractLINKS()

Do all data validation

encrypt in XML or JSON

Return Player

END LOOP

OUTPUT (“PLAYER NOT FOUND”)

END

# COMPONENT SPECIFICATION: GOAL-2 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | **Player Groups Creator** |
| ***Audience*** | Internal Stakeholders |
| ***Responsibilities*** | State computational abilities here. |
| ***Processing*** | 1. Acquire player Data  2. Analyze Player Data  3. Remove Abnormalities from Data  4. Store Player information into Database  5. Extract Player data  6. Filter Player Data according to Attributes  7. Group Players according to attributes  8. Check for incorrect Data within the groups  9. Verify the Player groups  10. Display the groups when necessary |
| ***Reference*** | ExtractPlayerAbilitiesFromFile |
| ***Constraints*** | MySQL Database |
| ***Composition*** | Sub - System 1, Module 3 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 10 to 18 |
| ***Interface/Tasks*** | Update Database, Perform Classification of Players |

## Procedure Definition Language (Pseudo-code):

INTERFACE: ExtractPlayerAbilitiesFromFile(Player\_Name)

DO

Extract Player Information

BEGIN

Function

Open dataset

Connect to Database

WHILE (Database.PName == Player\_Name) LOOP

Name=ExtractName()

BasicDetails=ExtractDetails()

Skills=ExtractSkill()

Abilities=ExtractAbilities()

ADD THIS PLAYER IN GROUP

PRINT NEW GROUP DETAILS

Return Player

END LOOP

END

# COMPONENT SPECIFICATION: GOAL-3

|  |  |
| --- | --- |
| ***Component Name*** | **Player Profile Processor** |
| ***Audience*** | External Stakeholders |
| ***Responsibilities*** | Process Player Profile Requests, Obtain and Populate Profiles |
| ***Processing*** | 1. Pre-Process Player Data 2. Train Statistical Model 3. Generate Player Features 4. Test Statistical Model 5. Save Trained Model 6. Optimize Choices 7. Choose Statistical Model 8. Tune statistical model hyper-parameters 9. Associate Player Profiles 10. Probe Statistical Models |
| ***Reference*** | **Player\_Profile\_Processor** |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system 2, Module 1 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 10 |
| ***Interface/Tasks*** | Check incorrect scaling, scale player profiles |

## Procedure Definition Language (Pseudo-code):

INTERFACE: PLAYER\_PROFILE\_PROCESSOR

DO

if(Player\_Details==NOT SCALED)

SCALE PLAYER PROFILES

Normalise data

else

Normalise data

BEGIN

for(Statistical Model in Library):

Check if its acceptable

Choose Statistical Model

if(Model is good)

Optimise Choices

After few epoch

Tune statistical model hyper-parameters

Associate Player Profiles and model

PLAYER\_DATA\_PREPROCESSOR();

TRAIN\_DATA,TEST\_DATA := TRAIN\_TEST\_SPLIT(PLAYER DATA);

MODEL1=MODEL\_SELECTOR();

END

# COMPONENT SPECIFICATION: GOAL-3 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **Player Data Pre-Processor** |
| ***Audience*** | External Stakeholders |
| ***Responsibilities*** | Process Player Profile Requests, Obtain and Populate Profiles |
| ***Processing*** | 1. Access Player Data 2. Remove Abnormalities from Data 3. Generate Player Attributes 4. Group Player Attributes 5. Create Player Profiles 6. Insert Data into Profiles 7. Associate Player profiles 8. Add Player Features 9. Demonstrate Important Player information 10. Generate Player Features |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system 2, Module 1 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Choose Statistical Model  , Optimise Choices |

## Procedure Definition Language (Pseudo-code):

INTERFACE: PLAYER\_PROFILE\_PREPROCESSOR

DO

CHECK INCORRECT SCALING

SCALE PLAYER PROFILES

BEGIN:

import database libraries

connect to db

If(connencted)

foreach(TABLE table in database):

if table is desired:

fetch tht table

IF(DATA IS NOT SCALED) THEN

BRING DATA TO SAME SCALE

FEED SCALED DATA TO DATABASE INSTANCE

RETURN;

END

# COMPONENT SPECIFICATION: GOAL-3 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | **Model Selector** |
| ***Audience*** | External Stakeholders |
| ***Responsibilities*** | Process Player Profile Requests, Obtain and Populate Profiles |
| ***Processing*** | 1. Pre-process Player Data 2. Associate Player Profiles 3. Generate Player Features 4. Probe Statistical Models 5. Choose Statistical model 6. Train Statistical Model 7. Test Statistical Model 8. Save Trained and Tested Model 9. Decide Statistical Model Parameters 10. Optimize Choices |
| ***Reference*** | MODEL\_SELECTOR |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system 2, Module 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Tune statistical model hyper-parameters, Associate Player Profiles and model |

## Procedure Definition Language (Pseudo-code):

INTERFACE: MODEL\_SELECTOR  
DO  
  Probe Statistical Models  
  Choose Statistical Model  
  Optimise Choices  
  Tune statistical model hyper-parameters  
  Associate Player Profiles and model  
BEGIN  
  LOAD SELECTED MODELS;  
  FOR I IN SELECTED MODELS LOOP:  
    TRAIN I  
    RESULTS[I] := I.MAKE\_PREDICTIONS(TEST\_DATA);  
  END LOOP;  
  OPTIMAL\_MODEL := DECIDE\_OPTIMAL\_MODEL(RESULTS);  
  TUNED\_OPTIMAL\_MODEL = TUNE(OPTIMAL\_MODEL);  
  RETURN TUNED\_OPTIMAL\_MODEL;  
END

# COMPONENT SPECIFICATION: GOAL-4

|  |  |
| --- | --- |
| ***Component Name*** | **USER QUERY PROCESSOR** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Process user queries |
| ***Processing*** | 1. Filter Relevant Data from Database 2. Make Data Structure to fit Data 3. Normalize Data Structure According to Model 4. Predict Value using Model 5. Scale Value Accordingly 6. Append Value in Database 7. Form Filling by End User 8. Validate player Information 9. Determine Data Abnormality 10. Append Player Data |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 2 , Module |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Get User Queries, Get User Input |

## Procedure Definition Language (Pseudo-code):

INTERFACE:USER\_QUERY\_PROCESSOR

DO

GET USER QUERIES

GET USER INPUT

BEGIN

BEGIN

CHOOSE MODEL

SET HYPERPARAMETER

FIND HIGHEST R SQUARE VALUE

DO DIMENSIONAL REDUCTIONALITY

KEEP PCA HIGHEST

IF(QUERY) THEN

USER\_QUERY\_RECOGNIZER(USER\_QUERY);

ELSE

USER\_INPUT\_HANDLER(USER\_INPUT);

END LOOP;

END

# COMPONENT SPECIFICATION: GOAL-4 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **USER QUERY RECOGNIZER** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Get user queries |
| ***Processing*** | 1. Access Database  2. Retrieve User Input  3. Analyze User Input  4. Convert Input to Query  5. Recognize User Query  6. Enter into Database  7. Execute Query  8. Retrieve Data After Execution  9. Display Data Requested  10. Update Database if necessary |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 2 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Get User Queries, Process User Queries And Generate Predictions |

## Procedure Definition Language (Pseudo-code):

INTERFACE USER\_QUERIES\_RECOGNIZER

DO

GET USER QUERIES

GENERATE STATS

BEGIN

KEEP NEW DATA

FIT THE DATA IN MODEL

PREDICT VALUE

PLAYER\_DATA = FETCH\_PLAYER\_DATA(USER\_QUERY);

LOAD MODEL;

PREDICTION = MODEL.PREDICT(PLAYER\_DATA);

OUTPUT(PLAYER\_DATA);

OUTPUT(‘PREDICTED VALUE:’);

OUTPUT(PREDICTION);

END

# COMPONENT SPECIFICATION: GOAL-4 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | **USER INPUT HANDLER** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Get user input |
| ***Processing*** | 1. Retrieve User Input Data 2. Create form for user 3. Retrieve Data from the Form 4. Analyze User Input Data 5. Take Certain Action on User Input 6. Access Database 7. Update Dataset 8. Notify User 9. Fix irregular output 10. Append Input |
| ***Reference*** | USER\_INPUT\_HANDLER |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 2 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | Get User Input, Process User Input And Append To Database |

## Procedure Definition Language (Pseudo-code):

INTERFACE USER\_INPUT\_HANDLER  
DO  
GET USER INPUT  
PROCESS USER INPUT AND APPEND TO DATABASE  
BEGIN  
FOR USER\_INPUT IN USER\_INPUTS LOOP  
    IF USER\_INPUT HAS MISSING VALUES THEN  
      OUTPUT(‘MISSING DATA WARNING’);  
      RETURN;  
    END IF;  
  IF USER\_INPUT HAS ABNORMALITIES THEN  
      OUTPUT(‘WARNING’);  
      FIX ABNORMALITIES;  
  END IF;  
END LOOP;  
APPEND USER\_INPUTS TO USER\_INPUTS DATABASE;  
END

# COMPONENT SPECIFICATION: GOAL-5

|  |  |
| --- | --- |
| ***Component Name*** | **PLAYER STATISTICS DEMONSTRATOR** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Fetch input player id/player name |
| ***Processing*** | 1. Locate Player in Database 2. Retrieve Player Statistics 3. Showcase Important Information On Player 4. Demonstrate Player performance 5. Contract Player Details 6. Display Player Rankings 7. Compare player Rankings 8. Demonstrate Graphs Visualizing Player Details 9. Show Player net worth graph 10. Go Back to Other Players |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 2 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE PLAYER\_STATISTICS\_DEMONSTRATOR:  
DO    
BEGIN  
  FETCH INPUT PLAYER\_ID/PLAYER\_NAME;  
  CONNECT TO DATABASE;  
  IF(PLAYER EXISTS IN DATABASE) THEN  
    STATISTICS=PLAYER\_STATISTICS\_GENERATOR();  
    DISPLAY\_PLAYER\_STATISTICS(STATISTICS);  
    RETURN;  
  ELSE  
    RETURN("PLAYER NOT FOUND");  
  END IF;  
END

# COMPONENT SPECIFICATION: GOAL-5 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **Player Statistics Generator** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Fetch required statistics from database for player |
| ***Processing*** | 1. Access Database  2. Locate Player in Database  3. Search Player Data  4. Retrieve Player Statistics  5. Show Important Information On Player  6. Demonstrate Player performance |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 2 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE PLAYER\_STATISTICS\_GENERATOR  
DO  
  IMPORT DATABASE LIBRARIES  
BEGIN  
  CONNECT TO DATABASE;  
  FETCH PLAYER DATA;  
  FETCH SELECTED STATISTICS;  
  FETCH REQUIRED STATISTICS FROM DATABASE FOR PLAYER;  
  RETURN STATISTICS;  
END

# COMPONENT SPECIFICATION: GOAL-5 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | **Player Statistics Displayer** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Generate HTML/JS code for graphs |
| ***Processing*** | 1. Access Database  2. Locate Player in Database  3. Retrieve Player Statistics  4. Show Important Information On Player  5. Demonstrate Player performance  6. Display Player Rankings  7. Display Estimate Value  8. Demonstrate Value Graph  9. Show Graphs Demonstrating player Details  10. Recommend Other players |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 2 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE DISPLAYER\_PLAYER\_STATISTICS  
DO  
  
BEGIN  
  CONNECT TO DATABASE;  
  GET STATISTICS;  
  DETERMINE GRAPHS TO BE DISPLAYED;  
  GENERATE HTML/JS CODE FOR GRAPHS;  
  DISPLAY GRAPHS;  
  RETURN;  
END

# COMPONENT SPECIFICATION: GOAL-6

|  |  |
| --- | --- |
| ***Component Name*** | **Player Value Estimator** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Estimate Player Values |
| ***Processing*** | 1. Check Player Data  2. Fetch Player Data  3. Feed data to the model  4. Generate predictive Value  5. Get predictive Value from Model  6. Display calculated value  7. Process and Transmit to Client  8. Update Player Data  9. Store Player Data  10. Display important data features |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 3 , Module: 1 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE: - Player Value Estimator

DO: Estimate Player Values

BEGIN

if(CONNECT(DATABASE))

foreach(Entry : database)

Player=Player\_Validator(Entry)

Data=Unwrap\_XML(Player)

Html\_Code=Player\_Showcaser(Data)

GenerateHTML(Html\_Code)

Save this Html

else

Show ERROR

END

# COMPONENT SPECIFICATION: GOAL-6 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **Player Data Validator** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Fetch data from all tables |
| ***Processing*** | 1. Check Player Data  2. Fetch Player Data  3. Feed data to the model  4. Generate predictive Value  5. Get predictive Value from Model  6. Display calculated value  7. Process and Transmit to Client  8. Update Player Data  9. Store Player Data  10. Display important data features |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 3 , Module: 1 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE: - Player Validator

DO

Go to given index in database

Fetch data from all tables

BEGIN

Check all data acquired is right

ADD Details in an Player Object

Make Object Immutable

Parse the Object using XML

END

# COMPONENT SPECIFICATION: GOAL-6 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | **Predicted Value Generator** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Make Graphs |
| ***Processing*** | 1. Check Player Data  2. Fetch Player Data  3. Feed data to the model  4. Generate predictive Value  5. Get predictive Value from Model  6. Display calculated value  7. Process and Transmit to Client  8. Update Player Data  9. Store Player Data  10. Display important data features |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 3 , Module: 1 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE: - Player Showcaser

DO

Format the table

Show Basic Details Tab

Show model of player

Make Graphs

Show Contract Details

Show Predicted Price

Show similar players

BEGIN

If(table\_formated())

If(data\_available())

Show output();

Else

Show Error;

END

# COMPONENT SPECIFICATION: GOAL-7

|  |  |
| --- | --- |
| ***Component Name*** | **Feedback Processor** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Process Feedback |
| ***Processing*** | 1. Get feedback from user  2. Process Feedback  3. Determine Feedback Response  4. Analyze Required Changes  5. Finalize the necessary changes  6. Get it approved from the team  7. Apply the necessary changes  8. Formulate the software  9. Re-Release Software  10. Re-Evaluate Regularly |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 3 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE:- Feedback Processor

DO: Process Feedback

BEGIN

if(user==closing site):

AskForFeedback()

if(userOpinion==Yes):

result=Load(Feedback\_Form\_Filler)

Feedback\_Answer\_Checker(result)

if(filledCorrectly):

Print(THANK YOU)

else:

reload(FeebackPage)

END

# COMPONENT SPECIFICATION: GOAL-7 OBJECTIVE-1

|  |  |
| --- | --- |
| ***Component Name*** | **Feedback Mechanism Generator** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Generate Feedback Mechanism |
| ***Processing*** | 1. Get feedback from user  2. Process Feedback  3. Determine Feedback Response  4. Analyse Required Changes  5. Finalise the necessary changes  6. Get it approved from the team  7. Apply the necessary changes  8. Formulate the software  9. Re-Release Software  10. Re-Evaluate Regularly |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 3 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE:-Feedback Form Filler

DO: Generate Feedback Mechanism

BEGIN

Load(User\_Feedback.html)

Guide User in Filling form

Wait till form submission

Result=FormDetails()

return Result

END

# COMPONENT SPECIFICATION: GOAL-7 OBJECTIVE-2

|  |  |
| --- | --- |
| ***Component Name*** | **Improvement Steps Applier** |
| ***Audience*** | Stakeholders |
| ***Responsibilities*** | Improvisation Steps |
| ***Processing*** | 1. Get feedback from user  2. Process Feedback  3. Determine Feedback Response  4. Analyze Required Changes  5. Finalize the necessary changes  6. Get it approved from the team  7. Apply the necessary changes  8. Formulate the software  9. Re-Release Software  10. Re-Evaluate Regularly |
| ***Reference*** | Procedure Name |
| ***Constraints*** | Environment Issues |
| ***Composition*** | Sub-system: 3 , Module: 2 |
| ***Resources*** | Player Database |
| ***Interactions*** | Components: 1 to 18 |
| ***Interface/Tasks*** | None |

## Procedure Definition Language (Pseudo-code):

INTERFACE:-Feedback Answer Checker

DO Improvisation Steps

BEGIN

foreach(Feedback : File)

assess=Compute(Feedback);

if(assess==GOOD)

SendEmail("ThankYou")

if(assess==BAD)

SendEmail("Sorry")

if(assess==Constructive)

Send Flag to Developer()

if(Developer==agree)

Implement the user feedback

END

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 9

**Player Value Analyser**

**System Construction**

***28-10-2017***

***Version 1.0***

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Python** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **DB, PHP** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **PHP/ Front End** |

**Approved By: Dr M. R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **116** |
| 2 | Architecture Objectives | **117** |
| 3 | System Design Specification | **118** |

# INTRODUCTION

# *The software engineering community realized that software architecture is not only about structures (components and interfaces), but also about system behavior (interaction between components, protocols). Furthermore, this community introduced an architectural design phase in the system life cycle, in which requirements should be satisfied and which should serve as a basis for detailed design activities. Researchers and engineers in software engineering have adopted the term 'architecture' as well. Nevertheless, there is no consensus about the subject; no universally-accepted definition of the term 'architecture' is agreed upon.*

# *Perry and Wolf (1992) consider a software architecture as a set of architectural elements that have a particular form. Similar to Zachman and Van Waes, they distinguish three different classes of architectural elements: processing, data, and connecting elements. Perry and Wolf consider an architecture as a necessary framework in which requirements are satisfied and which serves as a basis for the design.*

# *Garlan et al. (1995) stated that a system's architectural design is concerned with describing its decomposition into computational elements and their interactions. Design tasks at this level include organizing the system as a composition of components; developing global control structures; selecting protocols for communication, synchronization, and data access; assigning functionality to design elements; physically distributing the components; scaling the system and estimating performance; defining the expected evolutionary paths; and selecting among design alternatives.*

# *Soni et al. (1995) stated that software architecture is concerned with capturing the structures of a system and the relationships among the elements both within and between structures. Software architectures describe how a system is decomposed into components, how these components are interconnected, and how they communicate and interact with each other. Based on a survey on the role of architecture in the design and development of large systems within Siemens, Soni et al. notice that different structures are used at different stages of the development process. Each structure describes the system from a different perspective.*

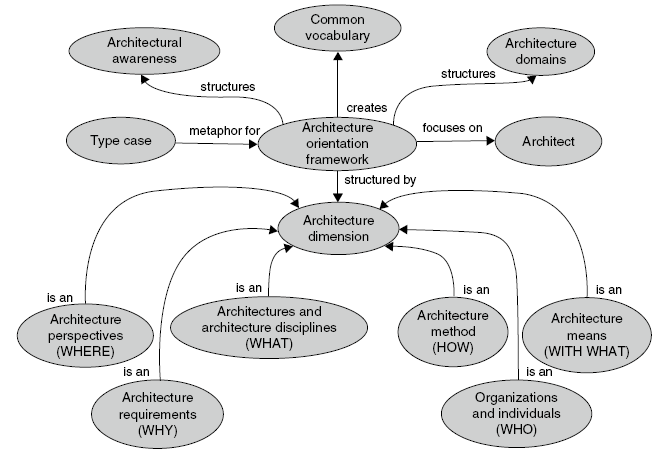
# *Soni et al. argue that the four different architectures they distinguished are needed because of the growing complexity of software throughout history (see Figure 1.3). Initially, only the code architecture was required. The module and execution architecture became necessary when systems became larger and distributed. Now, software engineers would like to use communicating objects and assemblies of reused components. Therefore, a high-level structure is described in the form of a conceptual architecture. On the other hand, Zachman and especially Van Waes reason that their various architectures are wanted as representation for each of the involved actors.*

# *Garlan and Perry (1995) found that the term 'architecture' is used in a number of ways in software engineering. Among the various uses are a) the architecture of a particular system, as in 'the architecture of this system consists of the following three components,' b) an architectural style, as in 'this system adopts a client-server architecture,' and c) the general study of architecture, as in 'the papers in that issue are about architecture.'*

# *A discussion group at Carnegie Mellon University's Software Engineering Institute developed a typical definition: the structure of the components of a program/system, their interrelationships, and principles and guidelines governing their design and evolution over time. They represent a spectrum in the software architecture community about the emphasis that should be placed on architecture - its constituent parts, the whole entity, the way it behaves once built, or the process of building it. Taken together, they reflect the various aspects of software architecture.*

# *Software architecture is concerned with the design and implementation of IT systems. From the viewpoint of architectural activity, software architecture covers the steps necessary to design and implement architecture. With regard to the structural aspect of architecture, software architecture describes the structures of IT systems. From this point on, the terms “IT system” and “system” are used synonymously provided no explicit differentiation is necessary. A system is a unit that consists of integrated software and hardware building blocks and exists for the purpose of fulfilling a functional objective. To achieve this objective, it communicates with its environment and must take account of the conditions defined by the environment.*

# *http://www.home.zonnet.nl/azwegers/thesis/figures/2_2.gif*



# ARCHITECTURE OBJECTIVES

* ***To manage complexity****: An architectural model allows one to present the essence of a complex system in a (simple) model. An architectural model supports the ability to comprehend complex systems; it presents them at a level of abstraction at which a system's high-level design can be understood. It supports the analysis of relationships as an aid to understand complexities in a design environment. In particular, an architecture is needed in complex, dynamic environments (Van Waes, 1991). Zachman states that the increased scope of design and levels of complexity of system implementations are forcing the use of architectural models for defining and controlling the interfaces and the integration of the system components (Zachman, 1987). Architectural models abstract away from details instead of from the essential complexity. Brooks claims that 'the complexity of software is an essential property, not an accidental one' (Brooks, 1995; p. 183). Descriptions of a software entity that abstract away its complexity often abstract away its essence.*
* ***To serve as a set of specifications****: An architecture may be seen as a result of the design process. It is laid down in specifications, which are derived from the requirements, and from which the desired system can be built. Specifying an architecture is concerned with the specification of components, their interactions, and the constraints on these entities and their interactions. These unambiguous specifications define the scope of future development activities, and serve as a basis for further design and implementation activities.*
* ***Means of communication****: Furthermore, an architectural model may play the role of a means of communication during a system (re-)design process. The architect can use it to visualise various aspects of the system to be designed, thus providing the various parties concerned with a basis for discussion and decision-making. By producing order in chaos, architectural models help each party to clarify its perception of the problem. Visualisation and explanation of the relevant aspects of the problem area, and the possible relationships between them, supports the various actors to focus their attention on the essential elements, thus providing a basis for discussion of the problems.*
* ***To indicate the most vital system elements****: Furthermore, the architecture determines the nature and quality of a system. As such, an architectural model indicates the invariant or most vital system elements, which must be treated carefully during system re-design. Systems evolve and are adapted to new uses, just as buildings change over time and are adapted to new uses. One frequently accompanying property of evolution is an increasing brittleness of the system, caused by violations of the architecture. Violations of the architecture frequently lead to an increase in problems in the system and contribute to an increasing resistance to change, or at least to changing gracefully.*
* ***Means to reduce the impact of changes****: Another role of an architecture involves its contribution to the effective re-design of a system. The architecture should reduce the impact of changes to the lower component levels, and to as few components as possible. Both for shop floor control systems and for products, it is advantageous to use as many parts of the existing system or product design as possible. In a re-engineering trajectory, an architectural model of the system allows one to pinpoint and discuss the areas requiring major change, and to integrate the new specifications into the existing model. Furthermore, architectural change is not so much determined by the system components, as well by the interfaces between these components; the ease with which components can be modified, replaced, or with which the system can be extended by new components is dependent on the extent to which the interfaces of the new components match those of the old ones.*
* ***Means to gain strategic benefits****: Finally,(product) architecture may have certain strategic importance for a company. The development of a new product brings together a wide range of technologies. Only a few of these technologies contribute to ultimate competitive advantage. Successful companies do not compete on (and even give away) the enabling technologies on which their core utility is based. By the architectural design of functions that can be filled in by cheap, standard components, companies profit from the strong competition in the markets for these components, and are free to focus on their true sources of competitive value. In addition, a company might extend the value of its product by publishing the product's interfaces to the outside world. Other enterprises might use this product as an indispensable part for their own products*

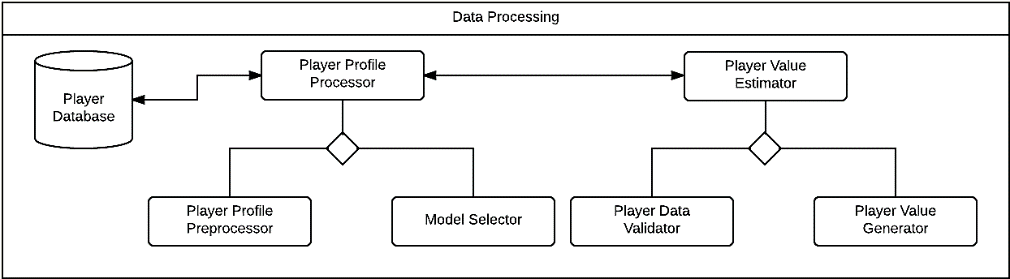
# SYSTEM DESIGN SPECIFICATION

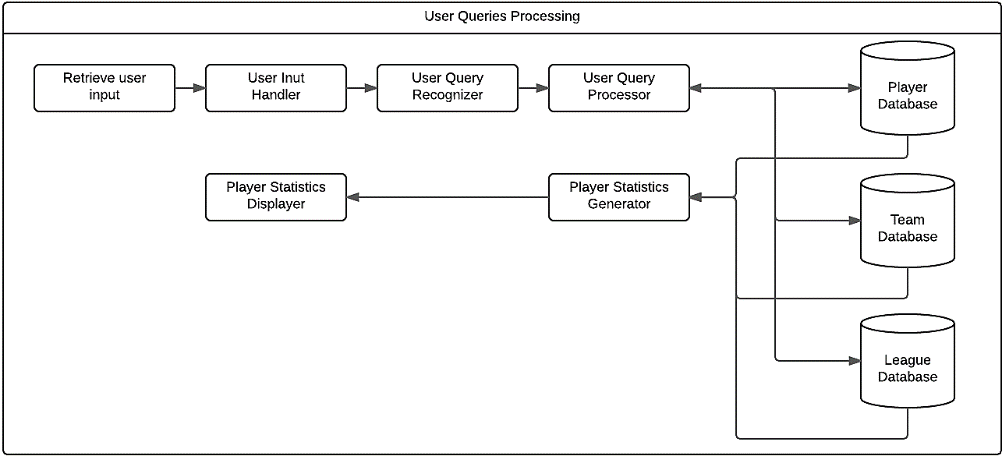
*A modular architecture may naturally result in a layered architecture; modules are assigned to specific layers. Layers reflect design decisions based on allowable relations and interfacing constraints. The layers in an architecture represent allowable interfaces among modules. Modules within a layer can communicate with each other. Modules in different layers can communicate with each other only if their respective layers are adjacent (Soni et al., 1995). A layer builds on its underlying layer, which at its turn builds on its underlying layer as well. Consequently, a layer explicitly uses the functionality of its underlying layer, and implicitly uses the functionality of all layers underneath its underlying layer.*

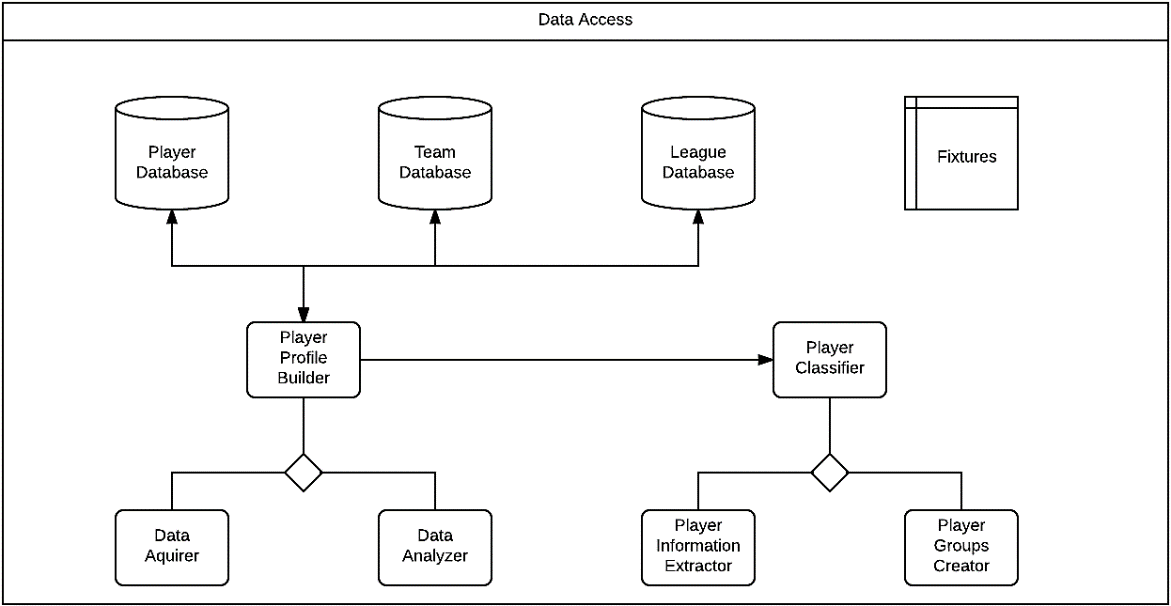
*Layers are used mainly to solve mapping problems. The mapping task is decomposed in layers: each layer performs a specific part of the mapping. In this sense, the division in layers is part of an architecture. The advantage of layers is the flexibility: changes can be made inside a layer without affecting other layers. A disadvantage of a layered architecture is its rigidity: new layers are hard to be shoved in between existing layers, since this requires a (major) change of interfaces. Examples of the application of layers in mappings are:*

* *the targets of an enterprise must be mapped on its physical processes; therefore, a strategical, tactical, and operational layer are distinguished;*

**

**

**

**

|  |  |
| --- | --- |
| Layer-1 | User Interfaces |
| Purpose | This the layer that the users will use to interact with the system. |
| Related Components | User Interfaces. |
| Software Interfaces | Layer 3 and Layer 4 Interfaces |
| Composition Style | **Generalization** |
| Communication Pattern | **Vertical** |
| Implementation Steps | 1. Create Website  2. Host Server  3. Access database to present statistics to user  4. Present player on-the-field information  5. Present player off-the-field information  6. Generate required statistical graphs  7. Generate forms for feedback  8. Generate navigation elements |

|  |  |
| --- | --- |
| Layer-2 | Data Processing |
| Purpose | This Layer processes the data of the system and makes predictions. |
| Related Components | Goal 3 and Goal 6 components |
| Software Interfaces | Layer 1 and Layer 4 Interfaces |
| Composition Style | **Aggregation** |
| Communication Pattern | **Horizontal** |
| Implementation Steps | 1. Access Data  2. Remove data abnormalities  3. Remove unnecessary database  4. Encode data in a format suitable for statistical models  5. Statistical Modelling  6. Check and cross-validate model  7. Tune model  8. Generate predictions and store in the database |

|  |  |
| --- | --- |
| Layer-3 | User Queries Processing |
| Purpose | The User Query Processing is done in this layer of components. |
| Related Components | Goal 4 and Goal 5 components |
| Software Interfaces | Layer 1 and Layer 4 Interfaces |
| Composition Style | **Composition** |
| Communication Pattern | **Horizontal** |
| Implementation Steps | 1. Accept Query  2. Extract the attributes required by the query  3. Gain Database access  4. Verify Database access  5. Query the required values from database  6. Validate values from the response  7. Send appropriate graphs/values to the interface |

|  |  |
| --- | --- |
| Layer-4 | Data Access |
| Purpose | The Data Access and Acquiring is done in this layer. |
| Related Components | Goal 1 and Goal 2 components |
| Software Interfaces | Layer 3 and Layer 4 Interfaces |
| Composition Style | **Aggregation** |
| Communication Pattern | **Vertical** |
| Implementation Steps | 1. Fetch data from data sources  2. Validate received data  3. Populate Player database  4. Populate League database  5. Populate Clubs database  6. Build Player Profiles  7. Group players on various attributes  8. Analyse groups and represent group insights in the database |

**T.Y. B. Tech.**

**CS 303: Software Engineering Laboratory**

Assignment No: 10

**Player Value Analyser**

**System Review and Acceptance**

***28-11-2017***

!!br0ken!! ***Version 1.0***

|  |  |  |  |
| --- | --- | --- | --- |
| Project Group Information | | | |
| Roll. No. | **Gr. No.** | **Name** | **Roles** |
| 18 | **151488** | **Anup Mahindre** | **Python** |
| 31 | **151600** | **Hiranyey Gajbhiye** | **DB, PHP** |
| 33 | **151482** | **Neeraj Ganu** | **Leader** |
| 34 | **151724** | **Sanket Ostwal** | **PHP/ Front End** |

**Approved By: Dr M. R. Dube**

**Academic Year: 2017-18 Semester: I**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page** |
| 1 | Introduction | **124** |
| 2 | Review Types | **124** |
| 3 | Verification Summary | **126** |
| 4 | Verification Matrix | **146** |

# INTRODUCTION

*At the time of the scheduled peer review, ensure proper representation and preparation by the reviewers. Provide clarifications on the work products. Present comments and listen to the comments of the other reviewers. Comments can be presented either by page or by reviewer. Keep the comment discussions short with a focus on detection, not correction. Editorial comments are provided separately and are not discussed at the scheduled review.*

*Participate in categorizing comments. The comments will be categorized and documented as errors, defects, and action items. Refer to the definitions for the categorization rules, which are summarized as follows:*

* *Errors (i.e., problems in the material currently under peer review).*

*Optionally, errors are subcategorized as major (affects functionality and/or performance) and minor (does not affect functional- ity and/or performance).*

* *Defects (i.e., problems in materials previously peer reviewed).*

*Optionally, defects are also subcategorized as major and minor.*

*Note: Defects will further be categorized as delivered or undelivered in the program’s change request system.*

* *Action items (i.e., unresolved comments requiring further investigation)*
* *A comment can remain categorized as a comment if the reviewers and presenters agree that there is no error, defect, or action item required.*

*To complete the peer review you must identify errors, defects, and action items to be resolved and documented. If needed, follow the program’s or project’s defined decision-making processes to elevate and reconcile any issues encountered in resolving peer review errors, defects, or action items with appropriate stakeholders. To ensure completion, per- form the following:*

* *Correct all errors and update the peer review information to indicate that the error is resolved.*
* *Submit change request paperwork for all defects. The status and tracking of the defect corrections are then handled through the change request system. The defects associated with the peer review should indicate this transfer and are categorized as resolved, allowing the peer review to be closed.*
* *Resolve and complete all action items. If any action items cannot be completed within the two-week period, these action items should be moved to the program- or project-level action item tracking system. The action items associated with the peer review should indicate this transfer and are categorized as resolved, allowing the peer review to be closed.*

# REVIEW TYPES

*Design and code reviews promise to improve software quality, ensure compliance with standards, and serve as a valuable teaching tool for developers. As with most practices, there are subtle nuances surrounding how they're performed that can dramatically affect their value. In some organizations, reviews are a valuable aspect of the software lifecycle. In others, they are a necessary evil tainted with political bureaucracy and big egos. Suboptimal reviews conducted late in the lifecycle are often misguided due to few objective guidelines that help guide the review process. When used throughout the development lifecycle, code and design quality metrics are valuable inputs to the review process.*

* 1. *Reviews Increase Agility Continuous Integration.*

*Agile practices are abundant, and for many teams interested in increasing their agility, valuable energy and resources have been devoted to improving these practices. Because of this, many teams have abandoned reviews while emphasizing other aspects of agility. But, reviews are an important tool in the agile toolkit.*

*A driving principle of the Agile Manifesto is continuous attention to technical excellence. Another is embracing and harnessing change as an opportunity to increase customer advantage. For developers, change often begins and ends with modifications to the source code. A poorly designed application with smelly code is a breeding ground for risk that makes change incredibly difficult, and is the greatest technical inhibitor to increased agility. Effective reviews that emphasize design quality and code cleanliness are an important aspect of increased agility. Reviews done right help ensure continuous attention to technical excellence. Unfortunately, not all reviews are done right.*

*1.2 Review Worst Practices*

*Some development teams find reviews a healthy and valuable asset to developers and the project team. Other teams realize little value from their review process. There are numerous causes for painful and ineffective reviews. Some symptoms of ineffective reviews include:*

* *Witch hunt reviews - Many reviews degrade quickly into attack and defend mode. This often occurs because the developer who wrote the code feels attacked and threatened when reviewers make direct and opinionated statements about the code. Nothing could be less productive.*
* *Curly brace reviews - Some reviews emphasize formatting and comments instead of more serious problems. Is placement of curly braces and misspelled comments really that important? Curly brace reviews are feeding ground for the anal retentive, and provide no real value.*
* *Blind reviews - Often times, reviewers walk into the review meeting having never laid eyes on the code they are about to review. Most of the review time is spent trying to figure out what the code does. Spending time in the review meeting attempting to understand the code instead of reviewing it for more serious ailments is a waste of time.*
* *Exclusionary reviews - Many times, the code provided for the review is only a sampling of the code written. For example, unit tests might be excluded from the review. In an unhealthy review environment, providing impartial and incomplete code listings will leave the reviewers wondering how the code actually works.*
* *Tree killer review - If you can't baffle them by providing half of what they need to understand the code, then maybe overwhelming them by providing thousands of lines of code might work. Waiting until codebase is incredibly large to host the first review is entirely ineffective. Not only is it to difficult to provide effective feedback on a large codebase, these reviews are often held late in the lifecycle and do not allow the developer to improve her code based on the feedback received.*
* *Token review - It's not uncommon for management to dictate that reviews be held. Token reviews are typically held for political reasons. Management wants to ensure that all code is reviewed for auditing purposes. Unfortunately, developers realize very little value surrounding these reviews. Any problems found are not fixed unless they are absolutely critical. Since the primary motivation is an audit trail for management, the team has little motivation to improve the code.*
* *World review- The reviews conducted with great number of people in attendance. This can be incredibly intimidating for the developers whose code is being reviewed, and it is not sure what value it provides to invite so many people. A few developers, up to five, should serve all the needs required of the review process. If more people want to provide input, there are better ways.*

*The Design checklist is as follows:*

* *Deficiencies and conflicts in requirements, architecture, or program/project plans will be reported.*
* *Design decisions and the decision rationales will be recorded according to plans and defined processes.*
* *Top-level software components of the software end item will be identified and described.*
* *Static relationships between top-level software components will be defined.*
* *Dynamic relationships between top-level software components will be defined.*
* *The concepts of execution of the software end item and its components will be defined.*
* *External interfaces of the software end item and its components will be identified and described.*
* *Top-level software components will be decomposed into lower-level software units.*
* *Internal interfaces between software units will be identified and described according to the standards identified by the project.*
* *Design traceability data will be documented according to plans, processes, and product standards.*
* *Design definitions will be documented according to plans, defined processes, and standards.*
* *Measurement and estimated data will be collected.*
* *Applicable work products will be submitted for peer reviews in accordance with project plans.*
* *Applicable work products will be submitted for control in accordance with program or project plans.*

# VERIFICATION SUMMARY

*Note: The verification summary is required to be written for all the objectives and processes as they were detailed as User Stories. Replicate the standard template for objectives and process for the goals.*

# VERIFICATION STEPS: GOAL-1

|  |  |
| --- | --- |
| Objective-1 | Acquire Player Data |
| Purpose | This will ensure the reliability and correctness of system. |
| Target Audience | Customers |
| Status | Completed |
| Role: | **As an**end user, developer |
| Verification Steps | 1. Verify that player profiles creation request is registered. |
|  | 2. Verify that player profiles are built. |
|  | 3. Verify created database and schema. |
|  | 4. Verify that player data has been fetched. |
|  | 5. Verify Database structure. |
|  | 6. Verify that player database has been populated. |
|  | 7. Verify that player database backup is available. |
|  | 8. Verify that the backup accessible. |
|  | 9. Verify proper database privileges and security. |
|  | 10. Verify Player page content. |

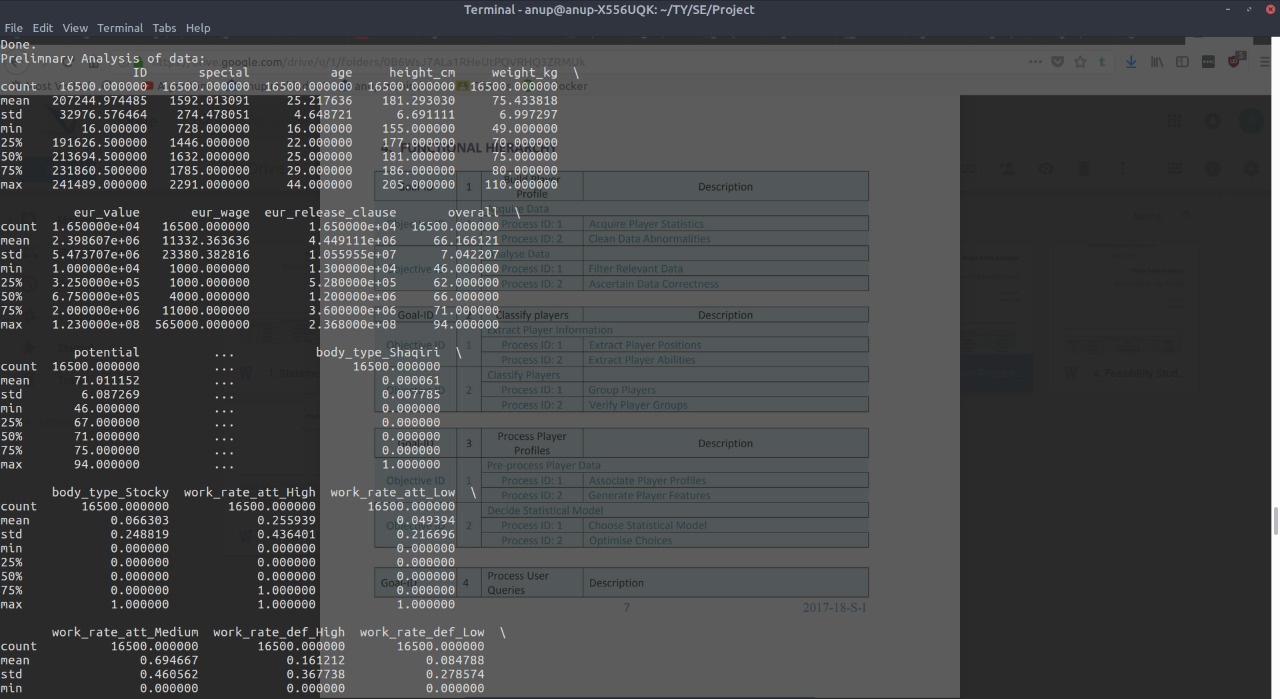
|  |  |
| --- | --- |
| Process-1 | Acquire Player Statistics |
| Purpose | Collect Player Statistics for creating player ranking index used to find transfer values. |
| Target Audience | Internal Stakeholders |
| Status | Completed |
| Role: | **As a**developer |
| Verification Steps | 1. Verify that required fields are correctly decided. |
|  | 2. Validate the player statistics |
|  | 3. Verify that the fields in database are created |
|  | 4. Validate player profile inputs |
|  | 5. Verify that the player statistics are added |
|  | 6. Validate the data limits and bounds |
|  | 7. Verify population of the player database |
|  | 8.Verify proper indexing of the database |
|  | 9.Verify database structure |
|  | 10. Verify that changes to original database are kept track of. |

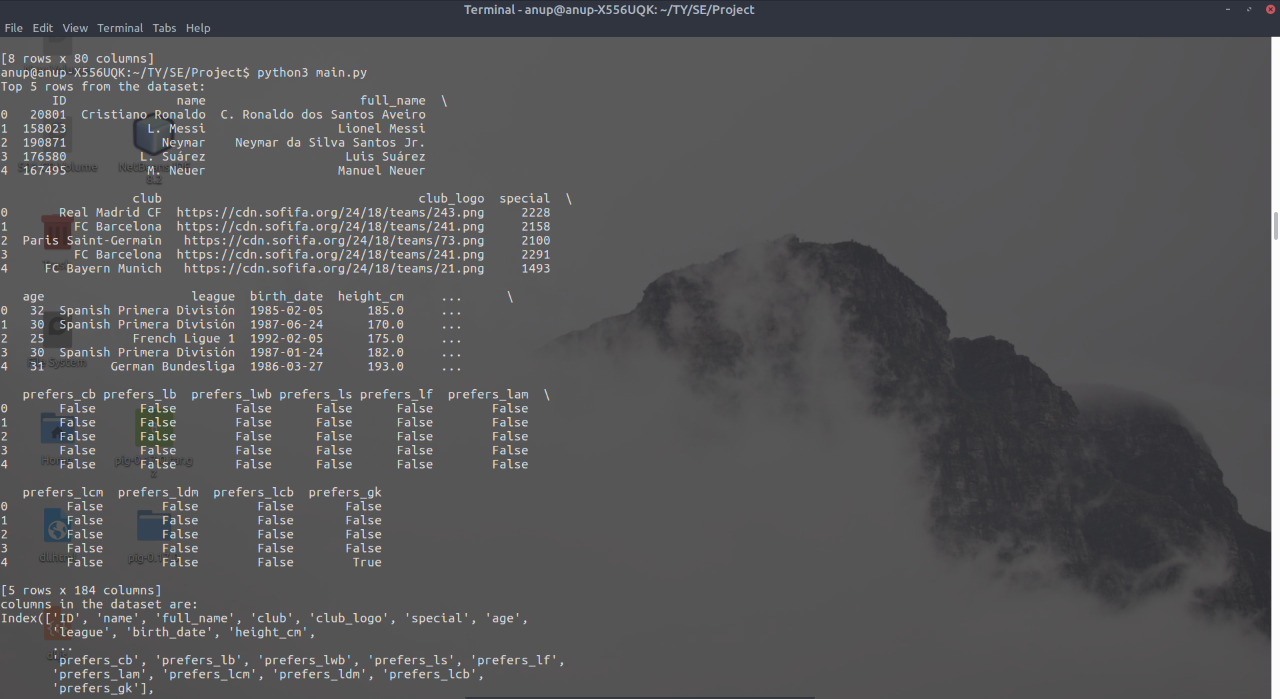
|  |  |
| --- | --- |
| Process-2 | Clean Data Abnormalities |
| Purpose | To keep the data relative and precise. |
| Target Audience | Developer |
| Status | On-going |
| Role: | **As a**developer |
| Verification Steps | 1.Validate Player profile format |
|  | 2.Validate player attribute ranges |
|  | 3.Verify graph plotting of values is complete |
|  | 4.Verify that abnormalities, if any are detected |
|  | 5.Verify that the abnormality can be found and accessed |
|  | 6.Verify the method of generating improvised data |
|  | 7.Validate corrected abnormality |
|  | 8.Verify that the changes have been committed on the database |
|  | 9.Verify that the data that was corrected wasn’t in use. |
|  | 10.Verify that the changes are logged |

|  |  |
| --- | --- |
| Objective-2 | Analyse Player Data |
| Purpose | To decide the Player analysis process of the system and the methodology to follow. |
| Target Audience | Internal Stakeholders |
| Status | Completed |
| Role: | **As a**developer |
| Verification Steps | 1. Verify Organisation of database attributes |
|  | 2. Verify the Design patterns for attributes |
|  | 3. Verify the output parameters |
|  | 4. Verify the important parameters priority |
|  | 5. Verify the short-listed attributes |
|  | 6. Verify Organisation of the parameters |
|  | 7. Verify the formulated observations |
|  | 8. Verify correspondence with Analysis team |
|  | 9. Verify the consolidation of analysis process |
|  | 10. Verify the final analysis methodology |

|  |  |
| --- | --- |
| Process-1 | Filter Relevant Data |
| Purpose | The purpose is to get detailed, relevant data about player which is filtered and curated. |
| Target Audience | Customers |
| Status | On-going |
| Role: | **As an**end user |
| Verification Steps | 1. Verify that a certain player can be found |
|  | 2. Verify that the transfer value of player can be accessed |
|  | 3. Verify that a curated list of playerscan be generated |
|  | 4. Verify that similar players playing in same position can be found |
|  | 5. Verify that players can be ordered by rating |
|  | 6. Verify that all players with comparable price can be seen |
|  | 7. Verify that statistics indicating on field behaviour can be accessed |
|  | 8. Verify that the statistics indicating off field characteristics can be accessed |
|  | 9. Verify that player’s current team can be seen |
|  | 10. Verify that player’s current team’s squad can be seen |

|  |  |
| --- | --- |
| Process-2 | Ascertain Data Correctness |
| Purpose | This will ensure the reliability and correctness of system. |
| Target Audience | Customers |
| Status | Completed |
| Role: | **As an**end user, developer |
| Verification Steps | 1. Validate correct player data |
|  | 2. Validate exact information reception |
|  | 3. Verify appropriate info of the player |
|  | 4. Validate Player Data accessibility |
|  | 5. Verify that transfer value is accessible |
|  | 6. Verify player value prediction feasiblity |
|  | 7. Verify precise database |
|  | 8. Verify data validation process |
|  | 9. Verify background checks |
|  | 10. Verify the data sources |





# VERIFICATION STEPS: GOAL-2

|  |  |
| --- | --- |
| Objective-1 | Extract Player Information |
| Purpose | Make groups in the database according to player’s attributes to help distinguish them. |
| Target Audience | External Stakeholders |
| Status | On-going |
| Role: | **As a**football club director |
| Verification Steps | 1. Verify that more than 1 player can be added. |
|  | 2. Validate saved data. |
|  | 3. Validate Player Data Extraction. |
|  | 4. Validate Club Squad |
|  | 5. Verify that other club squads can be accessed. |
|  | 6. Verify a player’s form can be assessed. |
|  | 7. Verify that market values can be found. |
|  | 8. Verify that other players can be searched. |
|  | 9. Verify that player background information is available. |
|  | 10. Verify that all parts of system are accessible to user. |

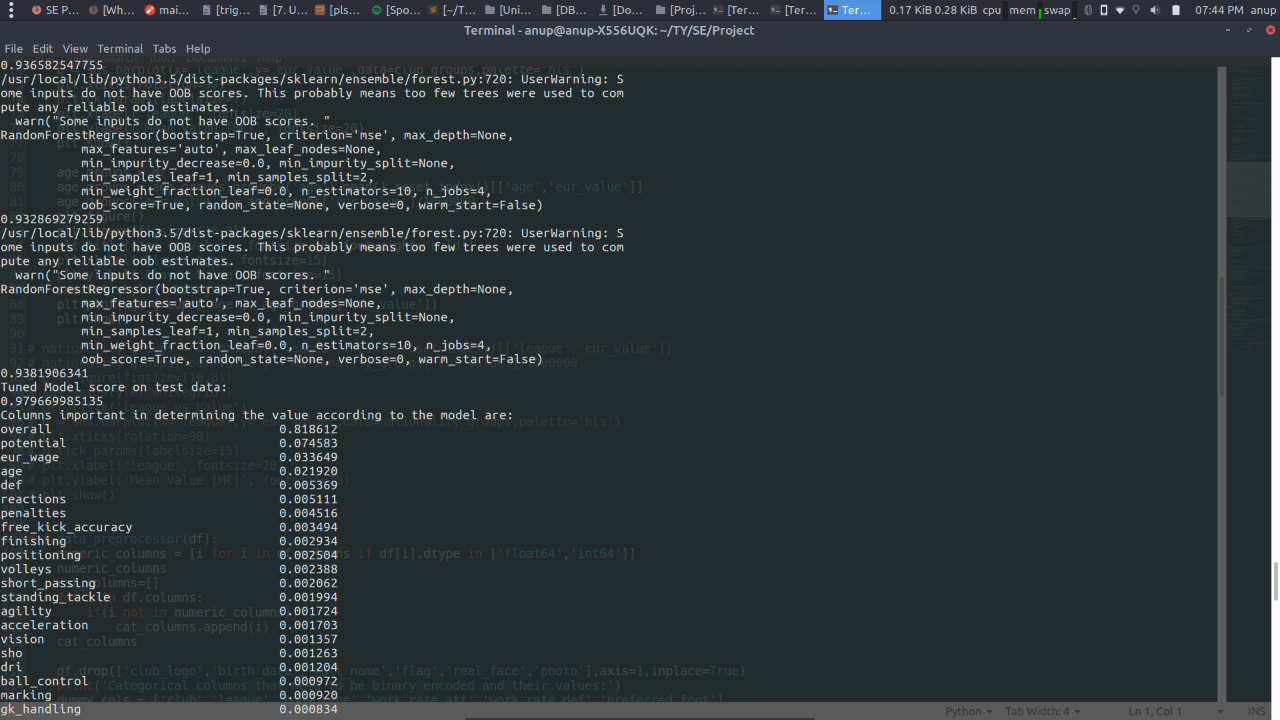
|  |  |
| --- | --- |
| Process-1 | Extract Player Position |
| Purpose | It will make it easier to search players. |
| Target Audience | External Stakeholders |
| Status | On-going |
| Role: | **As a**Football Club Coach |
| Verification Steps | 1. Verify that player positions are listed. |
|  | 2. Verify that the player position can be searched. |
|  | 3. Verify players can be filtered according to positions. |
|  | 4. Verify that list contains players with same position. |
|  | 5. Verify player’s other positions are listed to find out playing style. |
|  | 6. Verify player’s other positions are listed to find out player’s adaptability. |
|  | 7. Verify that player rating is listed. |
|  | 8. Verify that player value is listed. |
|  | 9. Validate value based on other similar players. |
|  | 10. Verify that player’s potential is listed. |

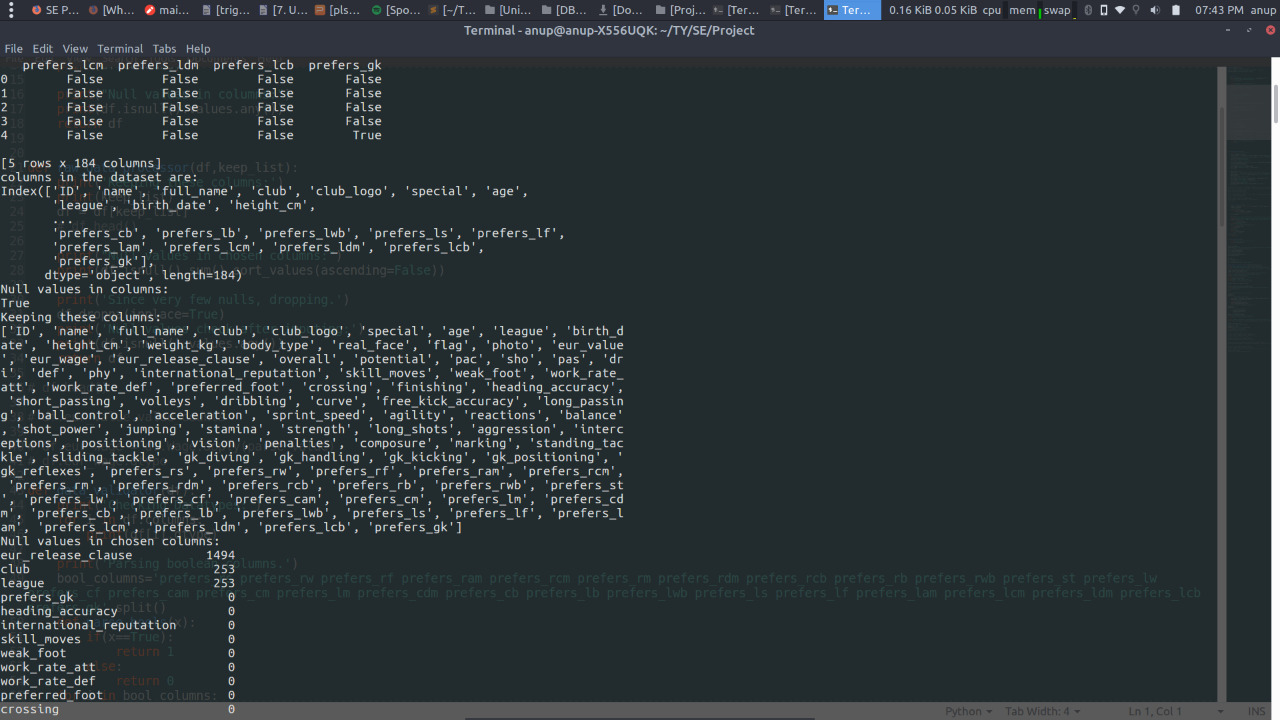
|  |  |
| --- | --- |
| Process-2 | Extract Player Ability |
| Purpose | It will help obtain a better prediction. |
| Target Audience | Customers |
| Status | Completed |
| Role: | **As a**user |
| Verification Steps | 1. Verify that player attributes are displayed. |
|  | 2. Verify that club squad is displayed. |
|  | 3. Verify that club fixtures are shown. |
|  | 4. Verify Club form from the news. |
|  | 5. Verify that transfer prices can be compared. |
|  | 6. Verify that player skills are displayed. |
|  | 7. Verify that player weaknesses can be checked. |
|  | 8. Verify the player’s chances of getting injured. |
|  | 9. Validate the player statistics comparison. |
|  | 10. Validate the player profile pictures. |

|  |  |
| --- | --- |
| Objective-2 | Classify Players |
| Purpose | To make classes of players as per categories. |
| Target Audience | Internal Stakeholders |
| Status | Completed |
| Role: | **As a**Developer |
| Verification Steps | 1. Verify correct data is acquired. |
|  | 2. Verify data distribution is performed. |
|  | 3. Verify inconsistencies are fixed. |
|  | 4. Verify that groups are prototyped. |
|  | 5. Verify player groups are established. |
|  | 6. Verify the player groups are accessible. |
|  | 7. Validate the basis of grouping. |
|  | 8. Validate encoding method is chosen. |
|  | 9. Validate encoding analysis is established. |
|  | 10. Verify results are integrated. |

|  |  |
| --- | --- |
| Process-1 | Group Players |
| Purpose | Decide attributes that can decide classes and groups. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a**Developer |
| Verification Steps | 1. Verify player data is acquired. |
|  | 2. Validate correct attributes are identified. |
|  | 3. Validate attribute wise data is examined. |
|  | 4. Verify attributes are extracted for grouping. |
|  | 5. Verify inconsistencies are detected. |
|  | 6. Validate inconsistencies are repaired and normalised. |
|  | 7. Validate correctness of data is verified. |
|  | 8. Validate player groups creation. |
|  | 9. Verify the groups are demonstrated. |
|  | 10. Validate player groups are delivered. |

|  |  |
| --- | --- |
| Process-2 | Verify Player Groups |
| Purpose | Verify player groups formed. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a**developer |
| Verification Steps | 1. Verify player groups are accessible. |
|  | 2. Verify player groups are analysed. |
|  | 3. Validate conceptualisation of player groups is done. |
|  | 4. Verify the basis derived is validated. |
|  | 5. Verify encoding technique is selected. |
|  | 6. Verify technique is valid. |
|  | 7. Validate encoding analysis. |
|  | 8. Verify modified encoding technique. |
|  | 9. Validate results of analysis. |
|  | 10. Verify integrated results. |





# VERIFICATION STEPS: GOAL-3

|  |  |
| --- | --- |
| Objective-1 | Pre-process Player Data |
| Purpose | Get the database  Standardize it’s format and data type  Make it useful for statistical modelling |
| Target Audience | Customers/ Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify raw player data |
|  | 2.Validate player data |
|  | 3.Verify that irrelevant attributes are disposed |
|  | 4.Verify the grouping and comparison attributes |
|  | 5.Verify the domain knowledge gained |
|  | 6.Verify that the domain knowledge gets represented in features |
|  | 7.Verify player features standardization |
|  | 8.Verify the data dimensionality |
|  | 9.Verify different feature selection has been accomplished |
|  | 10.Verify final data with selected features |

|  |  |
| --- | --- |
| Process-1 | Associate Player Profiles |
| Purpose | Capture the relation between the player and attributes  Discard irrelevant information |
| Target Audience | Customers/ Stakeholders |
| Status | Completed |
| Role: | **As a** *developer* |
| Verification Steps | 1.Validate formatted player data |
|  | 2.Verify available raw player attributes |
|  | 3.Verify only relevant attributes are present |
|  | 4.Verify consistency of selected attributes |
|  | 5.Validate attributes transformation |
|  | 6.Verify the comparison attributes between players |
|  | 7.Validate attributes by player positions |
|  | 8.Validate attributes by age |
|  | 9.Validate attributes by nationality |
|  | 10.Verify insights gained with data |

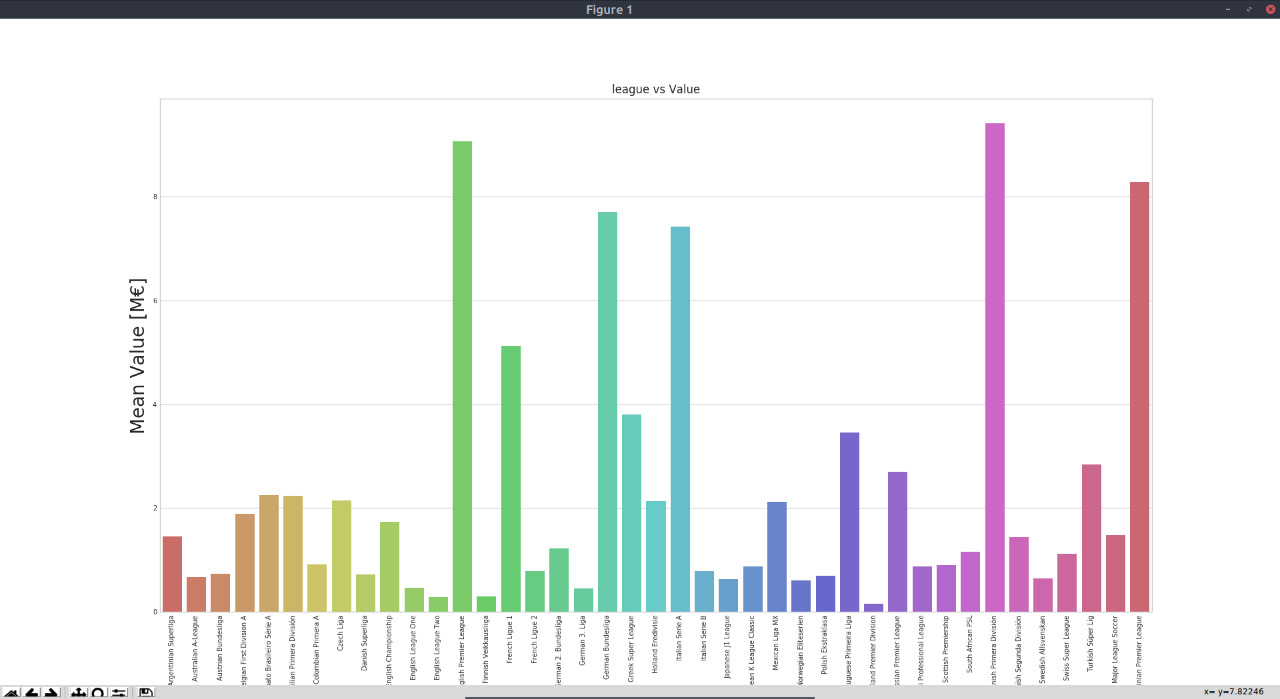
|  |  |
| --- | --- |
| Process-2 | Generate Player Features |
| Purpose | Generate the actual features that will be used in the modelling  Verify that no values are categorical  If categorical values exist, encode them in numerical format |
| Target Audience | Customers/ Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify tabulation of player data |
|  | 2.Verify the list of features that are valuable |
|  | 3.Validate transformations on the features |
|  | 4.Validate transformed features by visualzation |
|  | 5.Verify scaled features |
|  | 6.Verify the scales of the features |
|  | 7.Validate the features in dataset as a whole |
|  | 8.Verify different feature selection strategies |
|  | 9.Validate the application of feature selection strategy to the data |
|  | 10.Verify integration of generated features and methods |

|  |  |
| --- | --- |
| Objective-2 | Decide Statistical Model |
| Purpose | Choose a statistical model  Verify its results  Optimise the parameters and store the model for further usage |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify that accessed data is in proper format |
|  | 2.Verify the different statistical models |
|  | 3.Verify the application of a model and its results |
|  | 4.Verify the cross-validation and analysis of models |
|  | 5.Verify and overview analysis |
|  | 6.Verify selected optimal model parameters |
|  | 7.Verify the modified parameters |
|  | 8.Verify the cross-validation of changed parameter results |
|  | 9.Verify the use different scoring methods |
|  | 10.Verify the finalization on the model and parameters |

|  |  |
| --- | --- |
| Process-1 | Choose Statistical Model |
| Purpose | Try different statistical models  Use and cross-validate them  Store the model that performed the best with the default parameters |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify the statistical models |
|  | 2.Verify the input data to models is in proper format |
|  | 3.Verify the train test split |
|  | 4.Verify different statistical models |
|  | 5.Verify the results |
|  | 6.Verify results of different models |
|  | 7.Verify cross-validation on models |
|  | 8.Verify the analysis of cross-validation results for models |
|  | 9.Validate the output of selected models |
|  | 10.Verify the analysis |

|  |  |
| --- | --- |
| Process-2 | Optimise Choices |
| Purpose | Get the model stored from earlier process  Tune its parameters and cross-validate the changes  Store the tuned model for use in making predictions |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify the optimal model access |
|  | 2.Validate the parameters of the model |
|  | 3.Verify the changed the parameters and the results |
|  | 4.Verify the storage of the results with changed parameters |
|  | 5.Verify the cross-validation of changed parameter results |
|  | 6.Verify the Analysis of the cross-validation result |
|  | 7.Verify the training of the model with optimal parameters |
|  | 8.Verify the testing process of the model |
|  | 9.Verify the different scoring methods used in testing |
|  | 10.Verify the final model and parameters |

# 



# VERIFICATION STEPS: GOAL-4

|  |  |
| --- | --- |
| Objective-1 | Recognise User Queries |
| Purpose | To recognise user input. |
| Target Audience | Customers |
| Status | Completed |
| Role: | **As a** *<type of user>* |
| Verification Steps | 1. Verify large numbers of players can be added. |
|  | 2. Verify squads can be found easily. |
|  | 3. Validate player data in profile page. |
|  | 4. Verify other teams are visible to user. |
|  | 5. Validate squad player data. |
|  | 6. Verify news related to player is displayed. |
|  | 7. Validate market values from reliable sources. |
|  | 8. Verify player age is displayed |
|  | 9. Verify that nationality is listed. |
|  | 10. Validate the insights were used. |

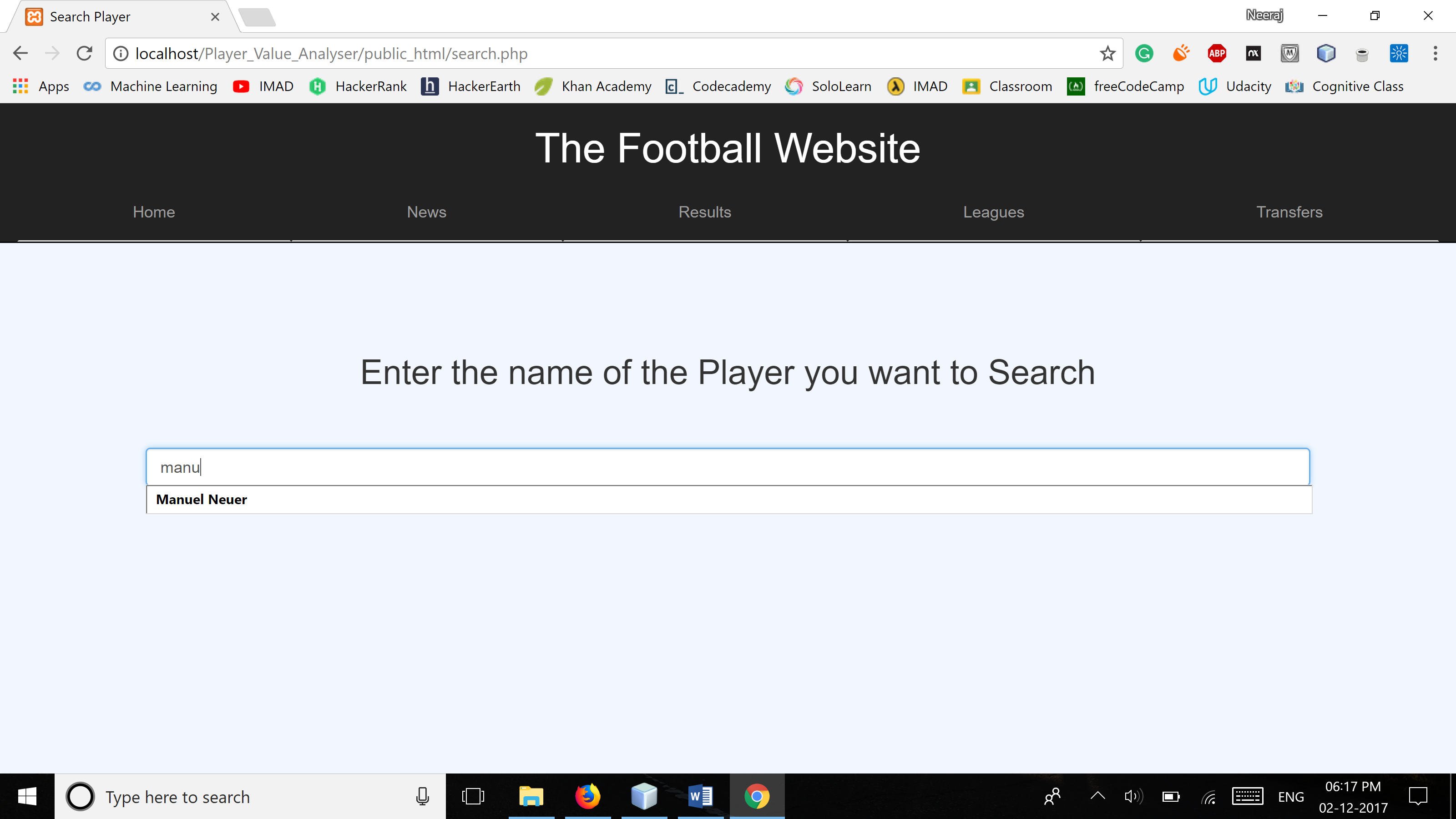
|  |  |
| --- | --- |
| Process-1 | Standardize Query Format |
| Purpose | Indicate purpose of the process here in 3/4/ statements. |
| Target Audience | Customers/ Stakeholders |
| Status | On-going/ Completed |
| Role: | **As a**database manager |
| Verification Steps | 1. Verify large numbers of players can be added. |
|  | 2. Verify squads can be found easily. |
|  | 3. Validate player data in profile page. |
|  | 4. Verify other teams are visible to user. |
|  | 5. Validate squad player data. |
|  | 6. Verify news related to player is displayed. |
|  | 7. Validate market values from reliable sources. |
|  | 8. Verify player age is displayed |
|  | 9. Verify that nationality is listed. |
|  | 10. Validate the insights were used. |

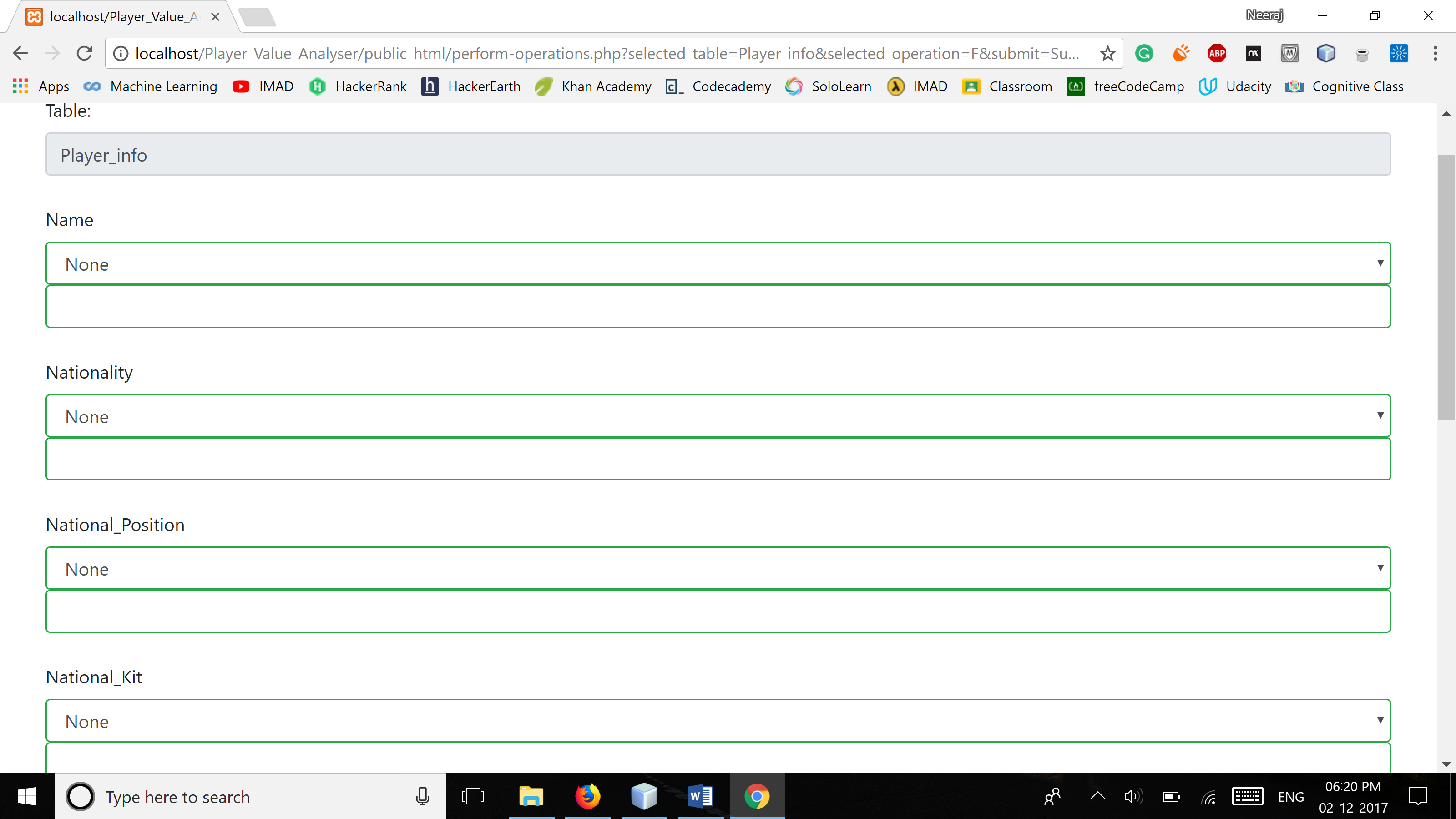
|  |  |
| --- | --- |
| Process-2 | Execute Prediction |
| Purpose | This will enable the output of the system. |
| Target Audience | Internal Stakeholders |
| Status | Completed |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify formatted user queries accessed. |
|  | 2. Verify player data is used for prediction |
|  | 3. Validate used player data. |
|  | 4. Verify input error is found. |
|  | 5. Validate data is normalised. |
|  | 6. Verify predictions process is initiated. |
|  | 7. Validate query data is included. |
|  | 8. Verify model extracts output. |
|  | 9. Verify session is valid after output. |
|  | 10. Verify that the objective is met. |

|  |  |
| --- | --- |
| Objective-2 | Handle User Input Data |
| Purpose | To handle user input data. |
| Target Audience | Customers |
| Status | Completed |
| Role: | **As a** *end user* |
| Verification Steps | 1. Verify all players are accessible. |
|  | 2. Verify anomalies are removed. |
|  | 3. Verify errors are found. |
|  | 4. Verify errors are suggested for correction. |
|  | 5. Validate player attributes in GUI. |
|  | 6. Verify data is provided to the system. |
|  | 7. Verify data is used in formula. |
|  | 8. Validate model to generate predicted value. |
|  | 9. Validate predicted value. |
|  | 10. Verify correct value is displayed for player.. |

|  |  |
| --- | --- |
| Process-1 | Pre-Process User Input Data |
| Purpose | This will pre-process user input data. |
| Target Audience | Internal Stakeholders |
| Status | Completed |
| Role: | As adeveloper |
| Verification Steps | 1.Verify raw player data |
|  | 2.Validate processed player data |
|  | 3.Verify unnecessary attributes are eliminated |
|  | 4. Verify grouping |
|  | 5.Verify domain knowledge gained |
|  | 6.Validate the domain knowledge features representation |
|  | 7.Verify player features standardization |
|  | 8.Validate data dimensionality |
|  | 9.Verify usage of different feature selection strategies |
|  | 10.Validate final data with features |

|  |  |
| --- | --- |
| Process-2 | Append Relevant Dataset |
| Purpose | To append these values in the database. |
| Target Audience | Customers |
| Status | Completed |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify player profiles can be viewed. |
|  | 2. Verify player profiles can be accessed. |
|  | 3. Validate removal of anomalies. |
|  | 4. Validate player data output. |
|  | 5. Validate data is fed to model. |
|  | 6. Verify errors are compared. |
|  | 7. Validate all attributes are viewable. |
|  | 8. Verify important attributes are segregated. |
|  | 9. Verify estimated value is calculated. |
|  | 10. Validate the player profiles are consistent. |





# VERIFICATION STEPS: GOAL-5

|  |  |
| --- | --- |
| Objective-1 | Generate Player Statistics |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Validate the generated performance score |
|  | 2.Validate the generated chemistry score with other players |
|  | 3.Validate the number of goals scored |
|  | 4.Verify the skills the player has |
|  | 5.Verify the popularity of the player |
|  | 6.Verify the told statistics |
|  | 7.Verify the Players past achievements |
|  | 8.Validate the players previous rating |
|  | 9.Validate the net worth of the player |
|  | 10.Validate the collected miscellaneous data |

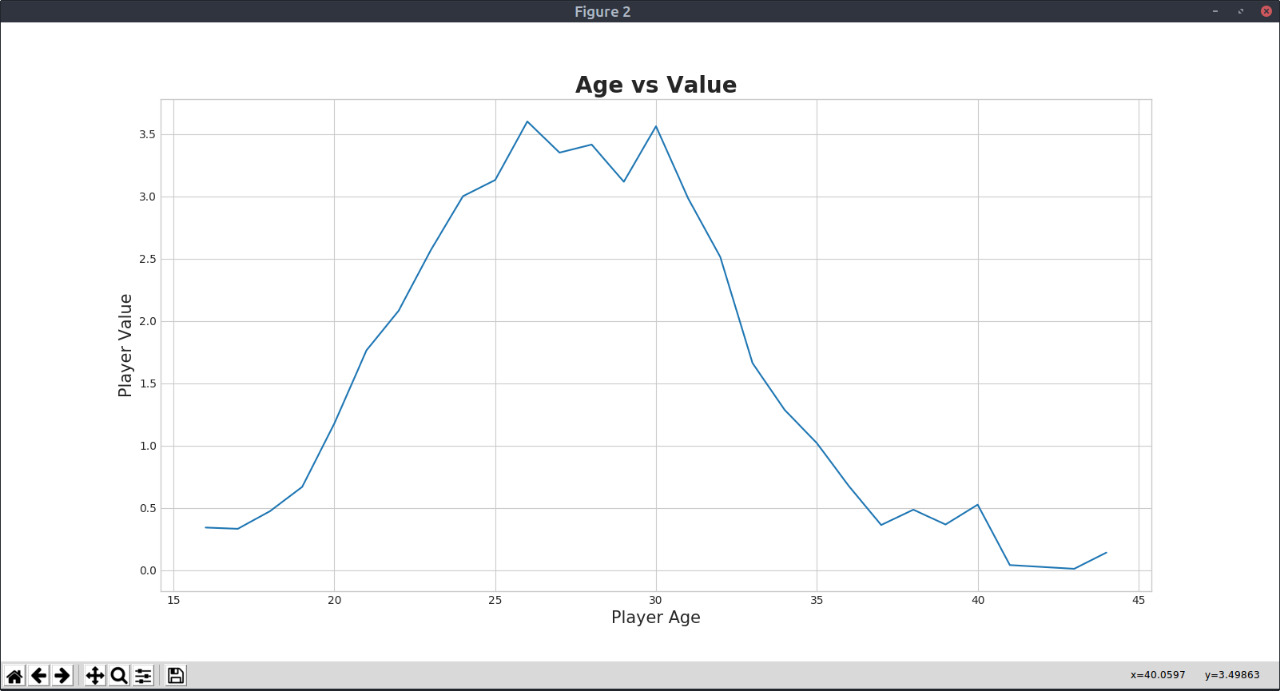
|  |  |
| --- | --- |
| Process-1 | Find Player Statistics |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Validate the collected player data |
|  | 2.Validate the found-out performance score |
|  | 3.Verify the best friends of the player |
|  | 4.Verify the player’s skills |
|  | 5.Verify all the non-basic player details |
|  | 6.Verify the extra details added |
|  | 7.Verify the best details |
|  | 8.Verify the use of social network |
|  | 9.Validate the downloaded photos and videos |
|  | 10.Veify the player’s fan following |

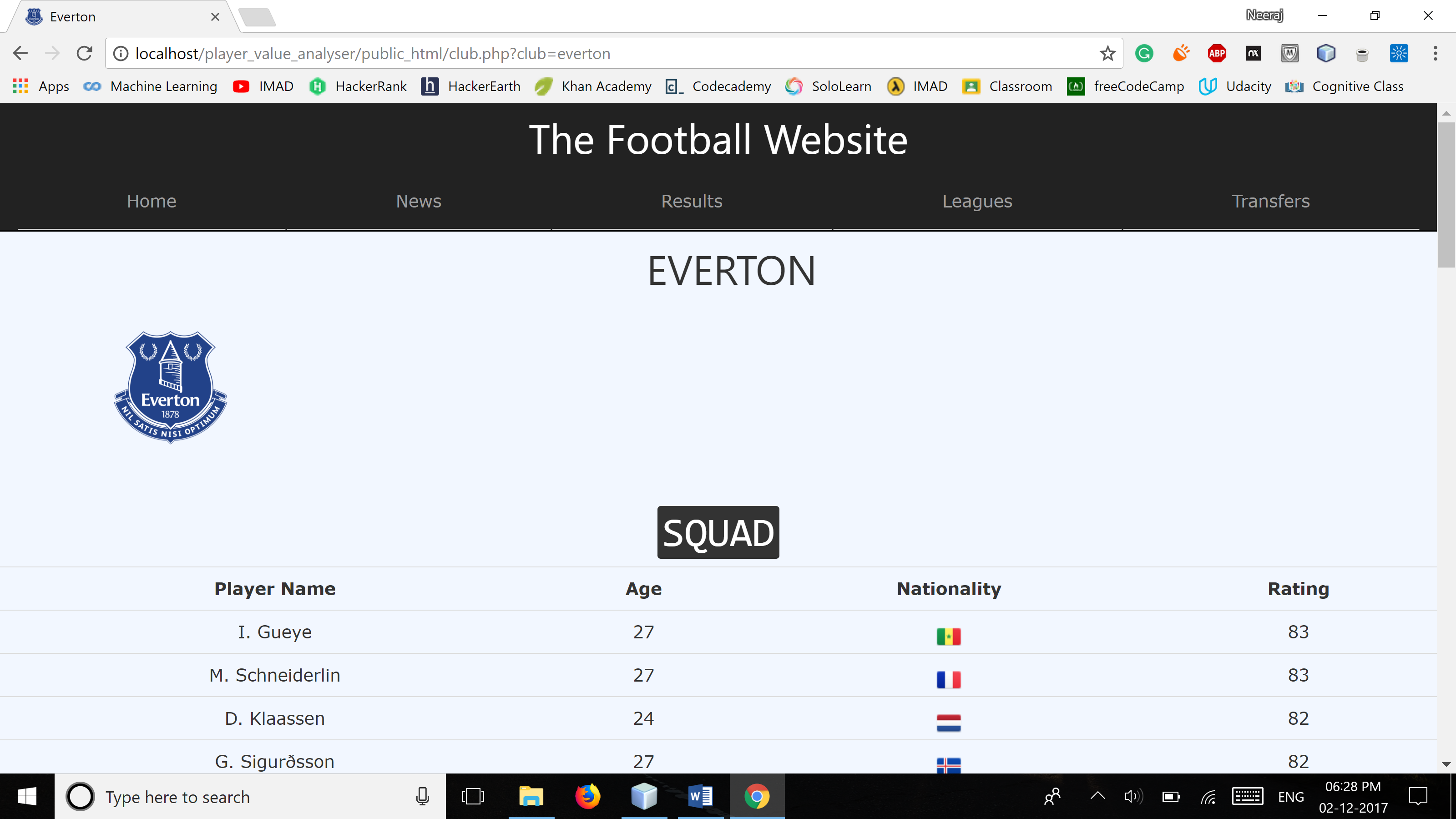
|  |  |
| --- | --- |
| Process-2 | Communicate Relevant Statistics |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers |
| Status | On-going |
| Role: | **As a** *user* |
| Verification Steps | 1.Verify the player’s basic details |
|  | 2.Validate the player’s statistics |
|  | 3.Verify the player’s social life |
|  | 4.Validate the miscellaneous details |
|  | 5.Verify the player’s behaviour |
|  | 6.Verify the player’s past |
|  | 7.Verify the player’s friends |
|  | 8.Verify the player’s controversies |
|  | 9. Verify the player’s crime record |
|  | 10.Validate the player’s relevant data |

|  |  |
| --- | --- |
| Objective-2 | Display Player Statistics |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Customers |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Validate the created bar graph |
|  | 2. Validate the created pie graph |
|  | 3. Validate the created scatter graph |
|  | 4. Validate the created deviation graph |
|  | 5. Validate the created growth chart |
|  | 6.Verify the player’s downfalls |
|  | 7. Validate the player rankings |
|  | 8.Verify the player’s milestones |
|  | 9. Verify the player’s achievements |
|  | 10.Validate the miscellaneous data |

|  |  |
| --- | --- |
| Process-1 | Generate Statistical Graphs |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Validate the sorted player data |
|  | 2.Validate the appropriate library for graph plotting |
|  | 3.Validate the retrieved player data |
|  | 4.Validate the labelled player’s graphs |
|  | 5.Verify whether appropriate scale has been chosen |
|  | 6.Verify the use of correct colours |
|  | 7.Validate the plotted graphs |
|  | 8.Validate the player’s ranking |
|  | 9.Verify the variation in performance of the player |
|  | 10.Verify the player’s net worth graph |

|  |  |
| --- | --- |
| Process-2 | Choose Relevant Statistics |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Validate the sorted player data |
|  | 2. Verify that data segments ae prioritized |
|  | 3. Verify if it is shown in tabular form |
|  | 4. Verify if different categories of data are being made |
|  | 5. Verify whether data with anomalies is hidden |
|  | 6. Verify the links of players |
|  | 7. Verify that his friend’s profiles are shown |
|  | 8. Validate the updated statistics |
|  | 9. Verify that the highlights are shown |
|  | 10. Verify Statistics can be viewed |





# 6 VERIFICATION STEPS: GOAL-6

|  |  |
| --- | --- |
| Objective-1 | Validate Player Data |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Validate the basic player details |
|  | 2.Verify if the player data has been scaled from respected source |
|  | 3.Validate the details entered by the end user |
|  | 4.Verify whether all anomalies have been deleted |
|  | 5.Verify that data admin has been called to delete big mistakes |
|  | 6.Verify that each player category has been appended |
|  | 7.Verify that all players who are not playing are archived |
|  | 8.Verify that a good structure has been made |
|  | 9.Verify that database has been normalized |
|  | 10.Verify that unauthorized users are not able to access the database |

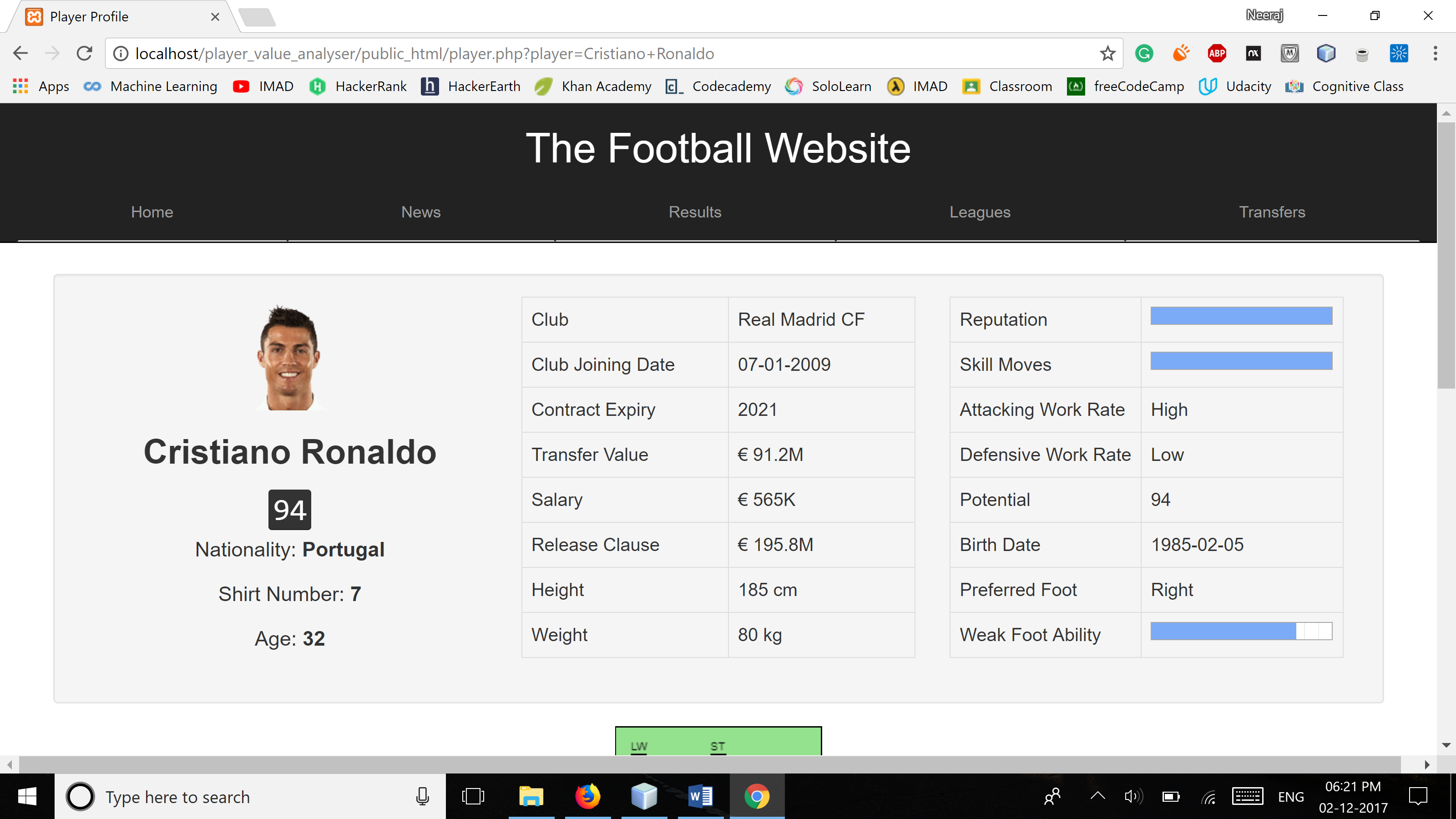
|  |  |
| --- | --- |
| Process-1 | Fetch Player Data |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify that data has been taken from end user |
|  | 2. Verify that data has been taken web scraping |
|  | 3. Validate data appended by admin |
|  | 4. Validate data from dataset repositories |
|  | 5. Validate the parsed data from different API |
|  | 6. Verify that ranking has been taken from FIFA officials |
|  | 7. Validate miscellaneous data from social networks |
|  | 8. Validate data from news |
|  | 9. Validate FIFA records |
|  | 10. Verify that a forum has been created |

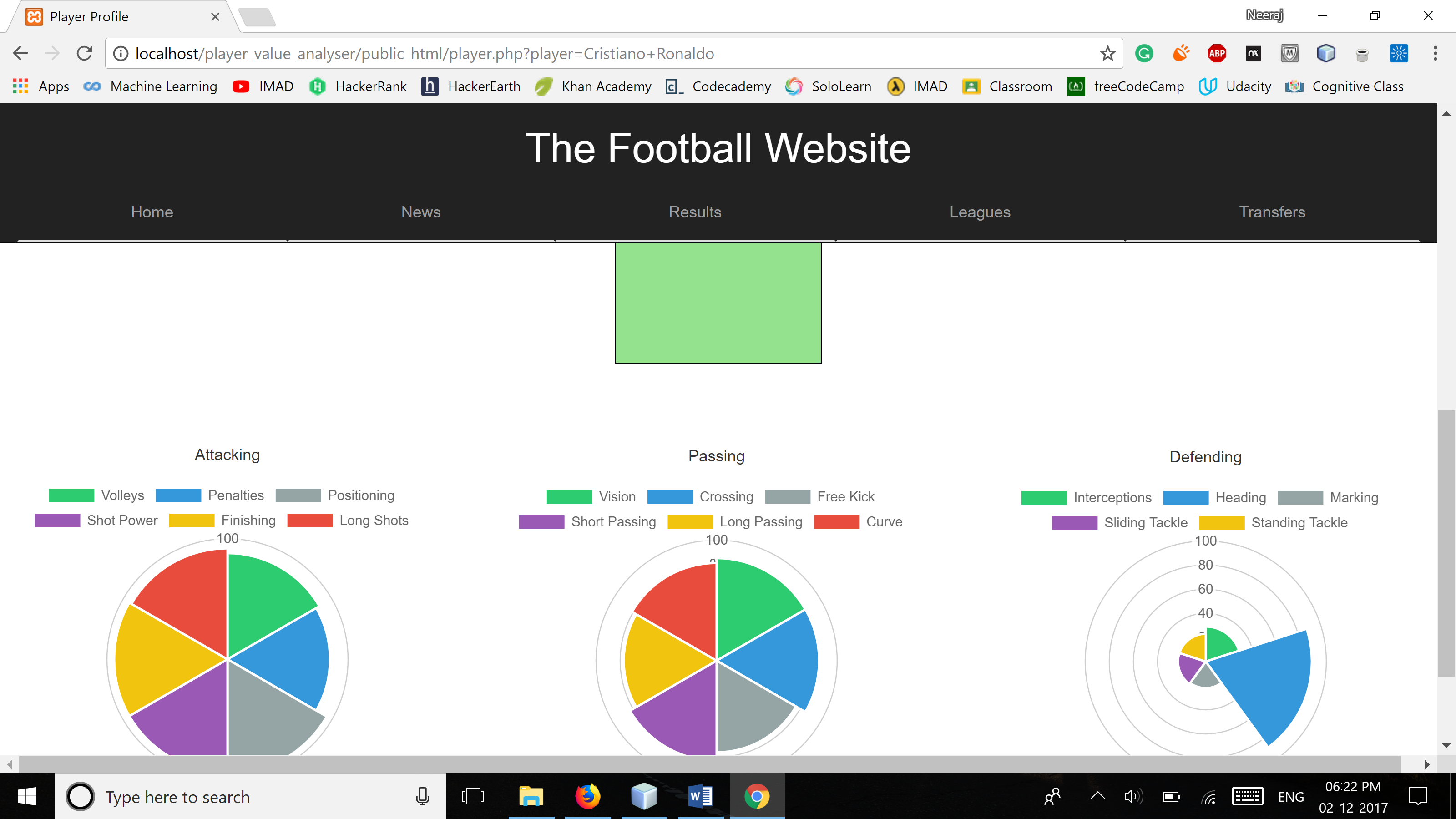
|  |  |
| --- | --- |
| Process-2 | Feed Data Model |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Validate basic data in the main table |
|  | 2. Validate player data in different categories |
|  | 3. Validate the rankings inserted in the other table |
|  | 4. Validate the inserted links about the player |
|  | 5. Validate the photos and videos in the database |
|  | 6. Validate the updated data in database |
|  | 7. Validate statistics that affect prediction |
|  | 8. Verify the R square value for data model |
|  | 9. Validate confusion matrix for predicted values |
|  | 10. Verify the use of dimensional reductionist |

|  |  |
| --- | --- |
| Objective-2 | Generate Predicted Value |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Validate data used for prediction |
|  | 2. Validate the calculated R square data |
|  | 3. Validate the scaled data |
|  | 4. Validate the categorised data |
|  | 5. Verify that only good data has been used |
|  | 6. Validate the trained model |
|  | 7. Verify the use of different regression models |
|  | 8. Validate the predicted test values |
|  | 9. Validate the calculated confusion matrix |
|  | 10. Validate the graph with predicted values |

|  |  |
| --- | --- |
| Process 1 | Communicate Player Value |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *User* |
| Verification Steps | 1. Verify that a new table has been created for values |
|  | 2. Validate the player details on the page |
|  | 3. Verify all terms asked by the player |
|  | 4. Verify the contract duration |
|  | 5. Validate all contract details of the player |
|  | 6. Validate the net worth of player |
|  | 7. Validate the base price of the player |
|  | 8. Validate the current price of the player |
|  | 9. Validate the predicted price of the player till the transfer window |
|  | 10. Validate other player data that is similar |

|  |  |
| --- | --- |
| Process 2 | Display Predicted Value |
| Purpose | Indicate purpose of the objective here in 3/4/ statements. |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *User* |
| Verification Steps | 1. Verify the GUI |
|  | 2. Validate all the player statistics |
|  | 3. Validate all the basic data of the player |
|  | 4. Validate all the player videos |
|  | 5. Validate all the values of the player |
|  | 6. Validate the player contract details |
|  | 7. Validate the similar player details |
|  | 8. Verify whether bidding market Is visible on the site |
|  | 9. Validate the transfer window dates |
|  | 10. Verify the transfer of player from one team to another |





# 3.7 VERIFICATION STEPS: GOAL-7

|  |  |
| --- | --- |
| Objective-1 | Generate feedback mechanism |
| Purpose | It is for taking a feedback from the user regarding the system. Help in determining if there are bugs or if any improvements can be made |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify the form created for feedback |
|  | 2.Verify the acquired user feedback |
|  | 3. Verify the accessed user feedback |
|  | 4. Verify the stored user feedback |
|  | 5.Verify the processed user feedback |
|  | 6.Verify the response given to the feedback |
|  | 7 Verify that necessary changes are applied |
|  | 8.Verify that system has been updated |
|  | 9.Verify the re-released software |
|  | 10.Verify that the feedback mechanism is in place |

|  |  |
| --- | --- |
| Process-1 | Access User Feedback |
| Purpose | It is for taking a feedback from the user regarding the system. Help in determining if there are bugs or if any improvements can be made |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1.Verify that feedback mechanism has been generated |
|  | 2.Verify that a form has been created |
|  | 3. Verify that feedback has been stored |
|  | 4. Verify that the feedback is valid |
|  | 5.Verify whether the feedback has been validated |
|  | 6.Verify that the feedback has been acquired |
|  | 7 Verify that necessary changes are applied in the feedback |
|  | 8.Verify and assess user feedback |
|  | 9.Verify the changes made in the feedback |
|  | 10.Verify that the system has been updated |

|  |  |
| --- | --- |
| Process-2 | Process User Feedback |
| Purpose | Helps to determine whether the feedback is genuine and if the changes are necessary to be made it helps in keeping the system updated |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify that a feedback mechanism has been generated |
|  | 2. Verify that a feedback form is created |
|  | 3. Verify that the feedback has been acquired |
|  | 4. Verify that feedback can be assessed |
|  | 5. verify that the feedback has been checked |
|  | 6. Verify whether the feedback is valid |
|  | 7. Verify whether changes have to made |
|  | 8. Verify that the changes have been finalised |
|  | 9. Verify whether the changes have been approved by the team |
|  | 10. Verify whether the feedback has been processed |

|  |  |
| --- | --- |
| Objective-2 | Apply Improvement Steps |
| Purpose | This will help to update the software with the right changes needed |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify that a feedback form is created |
|  | 2. Verify that the feedback has been acquired |
|  | 3. Verify that feedback can be assessed |
|  | 4. Verify that feedback has been processed |
|  | 5. verify that the response for the feedback has been generated |
|  | 6. Verify whether the changes have been analysed |
|  | 7. Verify that the changes have been finalised |
|  | 8. Verify whether the changes have been approved by the team |
|  | 9. Verify whether the changes have been made in system |
|  | 10. Verify whether the changes have been applied |

|  |  |
| --- | --- |
| Process-1 | Determine Feedback Response |
| Purpose | Know what is to be done with the feedback .If it is genuine changes are applied |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify that a feedback mechanism has been generated |
|  | 2. Verify that a feedback form is created |
|  | 3. Verify that the feedback has been acquired |
|  | 4. Verify whether feedback has been stored in database |
|  | 4. Verify that feedback can be assessed |
|  | 5. verify that the feedback has been checked |
|  | 6. Verify whether the feedback is valid |
|  | 7. Verify whether the feedback has been processed |
|  | 8. Verify what response has been made to the feedback |
|  | 9. Verify what changes have t be made if necessary |

|  |  |
| --- | --- |
| Process-2 | Aplply necessary changes |
| Purpose | Know what is to be done with the feedback .If it is genuine changes are applied |
| Target Audience | Stakeholders |
| Status | On-going |
| Role: | **As a** *developer* |
| Verification Steps | 1. Verify that a feedback mechanism has been generated |
|  | 2. Verify that the feedback has been acquired |
|  | 3. Verify whether the feedback is valid |
|  | 4. Verify whether the feedback has been processed |
|  | 4. Verify what response has been made to the feedback |
|  | 5. Verify what changes have t be made if necessary |
|  | 6. Verify whether the changes have been finalised |
|  | 8. Verify whether the changes have been approved by the team |
|  | 9. Verify whether the changes have been made in system |
|  | 10. Verify the software which is ready to release after making c=necessary changes |

# VERIFICATION MATRIX

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| User Story | Step-1 | Step-2 | Step-3 | Step-4 | Step-5 | Step-6 | Step-7 | Step-8 | Step-9 | Step-10 |
| G1:O1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G1:P1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G1:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G1:O2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G1:P1 | √ | √ | x | x | √ | x | √ | √ | √ | √ |
| G1:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G2:O1 | √ | √ | √ | √ | √ | √ | x | √ | √ | √ |
| G2:P1 | √ | √ | x | x | x | x | √ | √ | √ | √ |
| G2:P2 | √ | √ | √ | √ | √ | √ | √ | x | √ | √ |
| G2:O2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G2:P1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G2:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G3:O1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G3:P1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G3:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G3:O2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G3:P1 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G3:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G4:O1 | √ | √ | √ | x | √ | √ | √ | √ | √ | √ |
| G4:P1 | √ | √ | √ | √ | √ | x | √ | √ | √ | √ |
| G4:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G4:O2 | √ | √ | √ | √ | √ | x | √ | √ | √ | √ |
| G4:P1 | √ | √ | √ | √ | √ | √ | √ | x | √ | √ |
| G4:P2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G5:O1 | x | x | x | √ | √ | √ | x | x | √ | √ |
| G5:P1 | √ | √ | x | x | x | √ | √ | √ | x | x |
| G5:P2 | √ | √ | √ | √ | x | x | x | x | x | x |
| G5:O2 | x | √ | √ | x | x | x | x | x | √ | √ |
| G5:P1 | √ | √ | √ | √ | √ | √ | √ | √ | x | x |
| G5:P2 | √ | √ | √ | √ | √ | √ | x | √ | √ | √ |
| G6:O1 | √ | √ | x | √ | √ | √ | √ | √ | √ | √ |
| G6:P1 | √ | √ | √ | √ | √ | √ | x | x | √ | √ |
| G6:P2 | √ | √ | √ | x | x | x | √ | √ | √ | √ |
| G6:O2 | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| G6:P1 | √ | √ | √ | √ | √ | x | √ | √ | √ | x |
| G6:P2 | √ | √ | √ | x | √ | √ | √ | x | x | X |
| G7:O1 | X | X | X | X | X | X | X | X | X | X |
| G7:P1 | X | X | X | X | X | X | X | X | X | X |
| G7:P2 | X | X | X | X | X | X | X | X | X | X |
| G7:O2 | X | X | X | X | X | X | X | X | X | X |
| G7:P1 | X | X | X | X | X | X | X | X | X | X |
| G7:P2 | X | X | X | X | X | X | X | X | X | X |