Cricket Club Database and Player Selection System

Godalming College

Centre Number: 64395

Benjamin T Hunn

Candidate Number: 9784

Contents

Research and Analysis

Introduction

The purpose of this project is to design and create a system that assists the coach of a cricket club in selecting and managing players in the club. It's main purposes are to provide a database for storing information about the players and matches, to automatically generate teams for upcoming matches and to provide a user interface to interact with the database and the rest of the system.

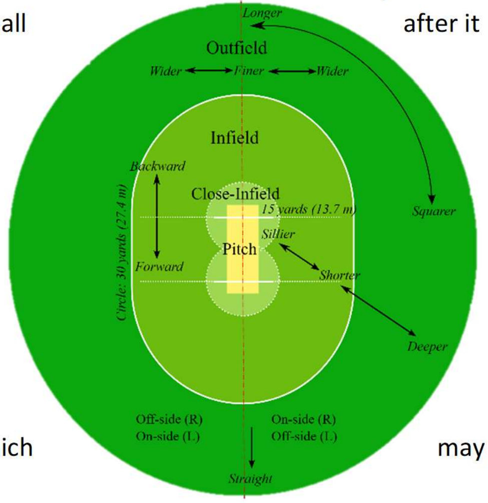
The End-User

Chiddingfold Cricket Club is a local organization that runs multiple cricket teams – from Under 7 to Under 13 junior teams, 2 senior men's teams and one senior women's team. Each team participates in various different local county-level leagues and cups, as well as friendly games against other local clubs. These teams contain a variety of number of players, averaging about 25 players per team for the junior teams and considerably more for the senior teams (40+ players).

The Domain

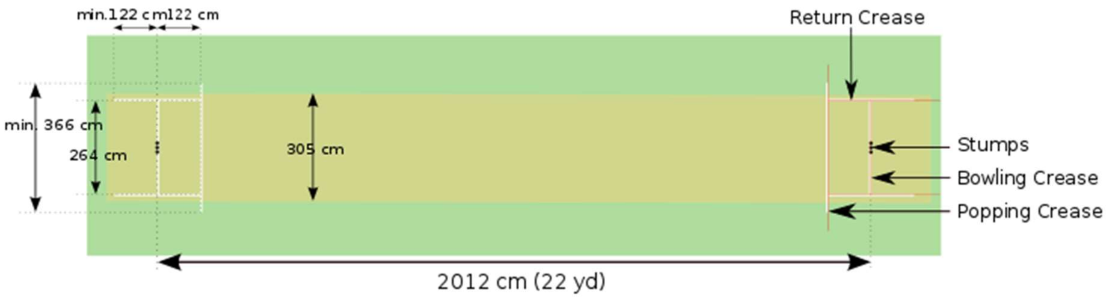
Overview of the sport Cricket, according to the international standard ruleset:

"Cricket is a bat-and-ball game played between two teams of eleven players on a field at the centre of which is a 20-metre (22-yard) pitch with a wicket at each end, each comprising two bails balanced on three stumps. The batting side scores runs by striking the ball bowled at the wicket with the bat, while the bowling and fielding side tries to prevent this and dismiss each player (so they are "out"). Means of dismissal include being bowled, when the ball hits the stumps and dislodges the bails, and by the fielding side catching the ball after it is hit by the bat, but before it hits the ground. When ten players have been dismissed, the innings ends and the teams swap roles." – https://en.wikipedia.org/wiki/Cricket



Playing area

"Cricket is a bat-and-ball game played on a cricket field (see image, right) between two teams of eleven players each.[58] The field is usually circular or oval in shape and the edge of the playing area is marked by a boundary, which may be a fence, part of the stands, a rope, a painted line or a combination of these; the boundary must if possible be marked along its entire length.[59]



In the approximate centre of the field is a rectangular pitch (see image, below) on which a wooden target called a wicket is sited at each end; the wickets are placed 22 yards (20 m) apart.[60] The pitch is a flat surface 3 metres (9.8 ft) wide, with very short grass that tends to be worn away as the game progresses (cricket can also be played on artificial surfaces, notably matting). Each wicket is made of three wooden stumps topped by two bails.[61]

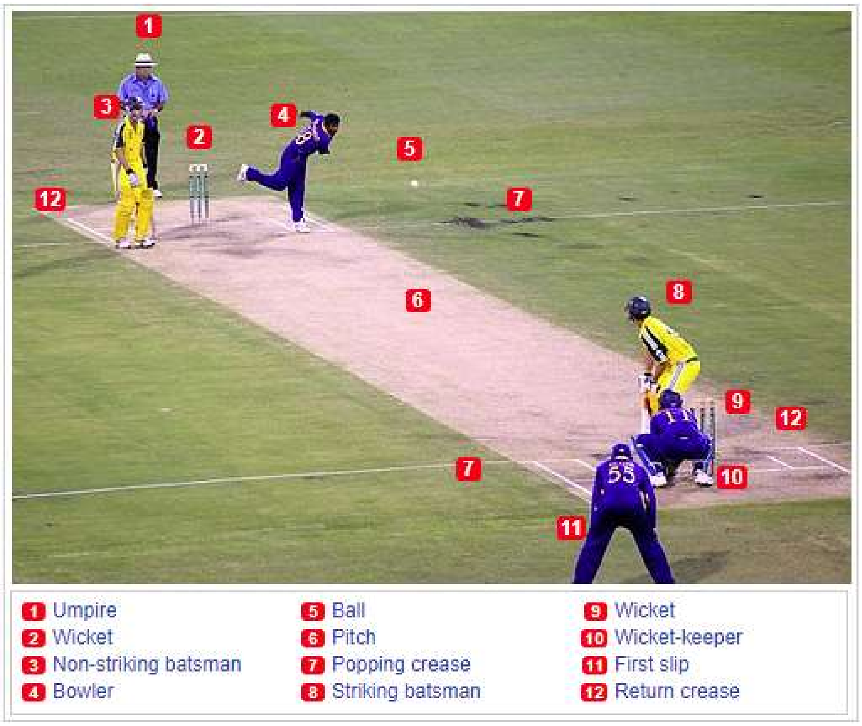
Cricket pitch and creases

As illustrated above, the pitch is marked at each end with four white painted lines: a bowling crease, a popping crease and two return creases. The three stumps are aligned centrally on the bowling crease, which is eight feet eight inches long. The popping crease is drawn four feet in front of the bowling crease and parallel to it; although it is drawn as a twelve-foot line (six feet either side of the wicket), it is in fact unlimited in length. The return creases are drawn at right angles to the popping crease so that they intersect the ends of the bowling crease; each return crease is drawn as an eight-foot line, so that it extends four feet behind the bowling crease, but is also in fact unlimited in length.[62]" - https://en.wikipedia.org/wiki/Cricket#Playing_area

Match structure

"Before a match begins, the team captains (who are also players) toss a coin to decide which team will bat first and so take the first innings.[63] Innings is the term used for each phase of play in the match.[63] In each innings, one team bats, attempting to score runs, while the other team bowls and fields the ball, attempting to restrict the scoring and dismiss the batsmen.[64][65] When the first innings ends, the teams change roles (. . .) During an innings, all eleven members of the fielding team take the field, but usually only two members of the batting team are on the field at any given time (. . .) The order of batsmen is usually announced just before the match, but it can be varied.[58]

The main objective of each team is to score more runs than their opponents but, in some forms of cricket, it is also necessary to dismiss all of the opposition batsmen in their final innings in order to win the match, which would otherwise be drawn.[66] If the team batting last is all out having scored fewer runs than their opponents, they are said to have "lost by n runs" (where n is the difference between the aggregate number of runs scored by the teams). If the team that bats last scores enough runs to win, it is said to have "won by n wickets", where n is the number of wickets left to fall. For example, a team that passes its opponents' total having lost six wickets (i.e., six of their batsmen have been dismissed) have won the match "by four wickets".[66]"



Batting and bowling

“During normal play, thirteen players and two umpires are on the field. Two of the players are batsmen and the rest are all eleven members of the fielding team. The other nine players in the batting team are off the field in the pavilion. The image with overlay below shows what is happening when a ball is being bowled and which of the personnel are on or close to the pitch.

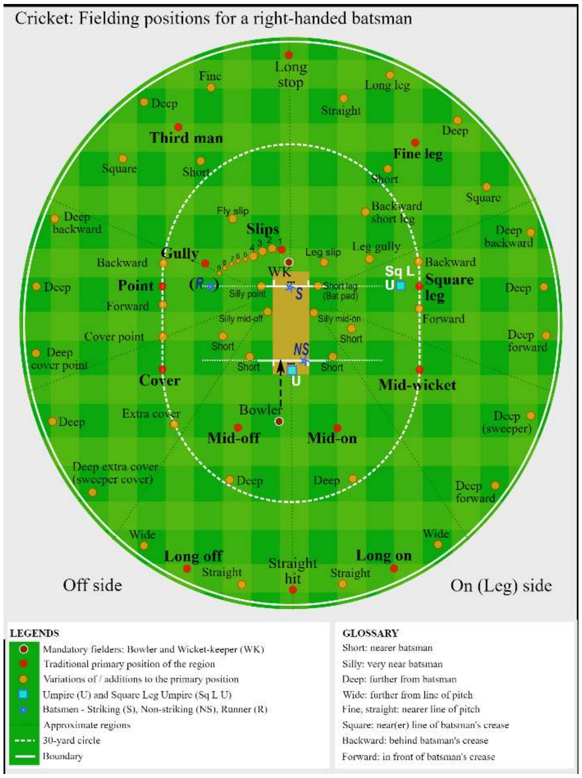
In the photo, the two batsmen (3 & 8; wearing yellow) have taken position at each end of the pitch (6). Three members of the fielding team (4, 10 & 11; wearing dark blue) are in shot. One of the two umpires (1; wearing white hat) is stationed behind the wicket (2) at the bowler's (4) end of the pitch. The bowler (4) is bowling the ball (5) from his end of the pitch to the batsman (8) at the other end who is called the "striker". The other batsman (3) at the bowling end is called the "non-striker". The wicket-keeper (10), who is a specialist, is positioned behind the striker's wicket (9) and behind him stands one of the fielders in a position called "first slip" (11). While the bowler and the first slip are wearing conventional kit only, the two batsmen and the wicket-keeper are wearing protective gear including safety helmets, padded gloves and leg guards (pads).

While the umpire (1) in shot stands at the bowler's end of the pitch, his colleague stands in the outfield, usually in or near the fielding position called "square leg", so that he is in line with the popping crease (7) at the striker's end of the pitch. The bowling crease (not numbered) is the one on which the wicket is located between the return creases (12). The bowler (4) intends to hit the wicket

(9) with the ball (5) or, at least, to prevent the striker (8) from scoring runs. The striker (8) intends, by using his bat, to defend his wicket and, if possible, to hit the ball away from the pitch in order to score runs.

Some players are skilled in both batting and bowling so are termed all-rounders. Adam Gilchrist, pictured above, was a wicket-keeper/batsman, another type of all-rounder. Bowlers are also classified according to their style, generally as fast bowlers, medium pace seam bowlers or, like Muttiah Muralitharan pictured above, spinners. Batsmen are classified according to whether they are right-handed or left-handed." - https://en.wikipedia.org/wiki/Cricket#Basic_gameplay:_bowler_to_batsman

Fielding



"Fielding in the sport of cricket is the action of fielders in collecting the ball after it is struck by the batsman, to limit the number of runs that the batsman scores and/or to get the batsman out by catching the ball in flight or by running the batsman out. There are a number of recognised fielding positions, and they can be categorised into the offside and leg side of the field." - https://en.wikipedia.org/wiki/Fielding_(cricket)

"[The fielders'] positions [are] determined on a tactical basis by the captain or the bowler. Fielders often change position between deliveries, again as directed by the captain or bowler.[72]" - https://en.wikipedia.org/wiki/Cricket#Fielding

There is always one special type of fielder called a wicket keeper, who stands behind the wicket. Their primary role is to stop deliveries that go past the batsman, in order to stop them scoring runs, but they can also dismiss the batsman by catching a ball that clips the batsman's bat or stumping them while they are outside their crease.

Variations on the international standard ruleset

There are variations on the standard ruleset that are often used for reasons such as age of players, number of available players and skill level. Team sizes can be 6, 8, 10 or 11 (generally increasing with age of players) and there are three different gameplay rulesets that are used: Pairs, Hybrid Pairs and 'Out when out'. The Pairs and Hybrid pairs formats are most commonly used for junior or younger players.

Summary of the Pairs cricket format:

"The pitch is two sets of stumps 12-16m apart, with a batting box at each end (see picture). There are 8-12 players in a team, organised into pairs. Each batting pair bats for 2 overs, and then the next pair of batsmen comes in. Every fielder bowls 1 over (an over is 6 balls). Runs are scored by changing ends with your batting partner. If the ball is bowled wide of the box the ball is called wide and the batting team get 2 runs. Each time a batter is out, 5 runs are deducted and the other batter faces the next ball.

A batter may be out if:

- they or the ball hit their stumps when the ball is bowled
- they hit the ball in the air and it is caught
- they aren't safely in their box when the fielders hit the stumps with the ball
- The team with the most runs scored from their overs wins."

- cricketcoachingblog.co.uk/2016/01/18/pairs-cricket

Scoring

Each side takes turns at batting. These are called innings. The batting team tries to score as many runs as they can until they are 'out'. The innings ends once all the batters are out or a certain number of overs have been served.

The problem

The current system used to pick players for each game is done manually – where each member tells the organiser whether they are available or not (via email) and then the organiser must decide who to pick for each game. There are multiple factors that go into selecting players. These are:

1. How many games they have been available for
2. How many games they've played
3. What their skill level in each aspect of the game is (batting, bowling, fielding)
4. Significance of the game
5. Exceptions and other factors

Currently the selection relies on the best judgement of the organiser who tries to make fair picks that give everyone a chance to play, depending on how many times they've been available. They will also try to pick their best players for important league games. Currently there is no reliable formula or system to ensure that fair picks are made. This can lead to suboptimal decisions resulting in problems such as some people playing significantly more or less games than others and not optimising the selection of players for the significance of the matches. For example, the 'Chiddingfold CC under 11s' participate in the 'West Surrey Youth Cricket League' in two divisions – the 'Under 11 division 1' and 'Under 11 development south west', but also play friendly games against other clubs. This means that for important league games the club organiser will want the best possible players whereas for less important games or friendlies the organiser will want to give other players a chance to play. It can be tricky to ensure that both of these criteria are met with the current system and it is possible for some people to play an unfair number of games.

Observation

Thought observation of the system I have been able to deduce a number of important things about it. I have gained an understanding of the email availability system in place. There is an email that is sent out at the beginning of each season. It provides all the fixtures for the season and the opportunity for parents to decide what matched they will be available for the entire season. This email is directly integrated with the club website and when the parents click the link to submit their availability, it is added to a database and is accessible to the coach. Two weeks before each game there is an automatic reminder email that is sent out to any parents who haven't submitted their availability, and then the coach makes the picks one week before the game.

In order to perform the tasks required by the user the system will need to collect various information over time. The aforementioned data can be separated in to two main categories: Player availability data and information about future games. In addition, data collected by the user on player performance per game may be considered separately but I won't address this here as this data will be collected directly through the UI. The data on player availability will be stored in a database where each available game for each player will be stored. This data is currently collected by email so to avoid confusion and make the transition to the new system easier; I will also be collecting this data via email however my system will be integrated with the database.

Interview

And interview involves talking to one specific person and asking them questions in order to gather information. An interview can be used to gain a detailed and specific insight into the organisation and the current system, as well as to gain a better understanding of the problem.

I will be interviewing a volunteer at Chiddingfold Cricket club – Joe McCarthy-Holland. He is directly involved in the process of picking players within the club and by interviewing him I hope to gain a clear view on the current system, how it works and what problems he has found with the current system. By getting this information directly from the end user I hope to gain a clear view on these things

These are the questions I am going to ask in my interview, what purpose they serve and how they will improve my understanding of the system.

- What is the current process used for selecting players for games?
  This question will give me information on how the current process works, therefore providing information on what systems and processes I have to work with as well as highlighting the flaws of the current system and giving me a better idea of what kind of system I need to develop.
- (How do players currently let you know when they're available? What third party software is used and how? How will my system need to interact with third party software or external data?)
  This question will tell me what process is used to gather the data I need – so I know how my system will need to interact (and be compatible) with external data sources and/or third-party systems, since I do not know how player availability data is collected.
- (How is player information and other relevant data currently stored?)

- What are the main issues with the current system that you would want to be resolved in a new system?
  This question will pinpoint the main issues with the current system and therefore help me create requirements and aims for the new system, including what problems with the old system to avoid/remove and what features to replace or improve.
- What works well with the current system and what functionality do you want to keep?
  This question will inform me on what aspects of the current system work as well as giving me a better idea of the functions of the current system, therefore telling me what functions the new system need to keep, thus ensuring the new system does still fundamentally provide the service required.
- What additional features and functionality would you want from a new system?
  This question will tell me what additional features should be added or incorporated to a new system in addition to its current features, therefore allowing me to specify my requirements and giving me a good idea of what problems I will need to solve and what features I will need to include. It will also help me ensure that I am meeting the requirements of the user.

This is my summary of the points mentioned in the interview (25/09/2019):

- **What is the current process used for selecting players for games? (And)**
- **How do players currently let you know when they're available?**
  - An email goes out and players can select which games they are available for.
  - When it is known who will be available for each game the players a picked manually
  - One factor affecting who is picked is how many games players have been available for – so players that have been available every game will play more than players who have only been available for a few games. Joe tries to keep the ratio of number of games available to the number of games played constant.
  - Another factor is the type of game being played. For example, if the game is a friendly then players who aren't as good may be picked, however for a Cup final ideally the best players will be picked.
  - One exception is if a player has been available for very few games, they should be picked for as many games as possible (unless the game is important, in which case they may not be picked).
  - Fair picks are the first priority but winning is preferable as this is more enjoyable for the players.
  - If two players have the same number of games picked and the same number of available games then the more 'enthusiastic' player may be selected.
  - Player availability is provided in advance. It can be given a long time in advance but it can also be soon before the game.
  - There is a 'maybe' option in the email if people aren't sure if they can make it

- **What third party software is used and how? How will my system need to interact with third party software or external data?**
  - Currently an email system is used to gather the availability data
  - Other apps and websites, such as Teamer, Pitchero and Playcricket are used

- It shouldn't be necessary to receive availability data from external software

- **What are the main issues with the current system that you would want to be resolved in a new system?**
- It is difficult to keep records of number of games played
- Past and future availability of players should be stored
- Each players ability for batting, bowling and possibly keeping should be stored (since each game needs one keeper)
- Since players are currently chosen manually, sometimes the picks are unfair. The new system should be able to help fair picks be made.

- **What works well with the current system and what functionality do you want to keep?**
- The factors currently taken into account should be taken into account when choosing players in the new system:
  - Ratio of player availability : games played
  - Players with very few available games should ideally get chosen for the games they're available for
  - Higher ability players should be chosen for Important games and lower ability players should be chosen for friendlies or less important games
  - Players that fail to turn up for a game they have been selected for should be recorded and possibly affect future selections
  - Players currently have the ability to play in game up to two years above their category. This flexibility should remain however players in the correct age bracket should have priority over younger players.
- Player picks should still be manually changeable. The team selected by the system should act as a suggestion or 'ideal' team
- The system should also be able to handle the 'maybe' option for availability

- **What additional features and functionality would you want from a new system?**
- There should be a database that stores player data such as number of games available, number of games played, number of games missed, batting skill, bowling skill and possibly keeping skill too.
- An email should be sent out that links to the database and allows people to select what games they will be available for. Perhaps also an alert system for deadlines.
- The system should try to maintain a constant ratio of games available to played games but should also handle exceptions (such as players with very few available games)
- The system should determine the optimum selections and then allow for any changes that the user wishes to make and there should be an interface for the user.
- The system should look ahead at future games and select players based on what type of games are coming up and who will be selected for those
- The system should give relevant information to inform decisions and perhaps a warning when changes are made to the suggested selections
- There should be some method of ranking player skill. This could be based of match statistics or entered manually but Joe suggested that a feature where the user is shown player

statistics and then enters the performance of each player each game might be a good solution

- Player attendance should be assumed as true but this should be manually changeable by the user.

Interview analysis

My interview was very successful and I gathered a lot of useful information about what the end user requires from the system.

The first question gave me a very clear idea of how the current system works and also clarified the current process and the factors that I will need to consider in a new system. The current solution is for the organiser to send out an email to all the players (or parents of the players) which contains a link that they can click to indicate which games they will be available for. This information may be received months or weeks in advance. The team organiser then takes the set of available players for a game and manually selects each player for each game. They make their selection based on number of games played and number of games available by each player, and also other factors such as the significance/difficulty of the game and skill of the players. They try to make fair picks (each player plays a number of games proportional to the number of games they've been available) while also choosing the most suitable players for each game, however all this is based on the judgement of the person choosing the players. They don't have much data available to inform these decisions and so the selections are largely based on the judgement of the person making the picks and their knowledge of the players.

The second question confirmed that I will not have to integrate with third party software in terms of collecting data on player availability, however the database system will need incorporate email integration. Third party software may, however, be used for collecting statistical data about games or players.

The fourth question helped to make clear the issues with the current system and therefore the problems that will need to be solved. One of the main issues with the current system is that it relies almost entirely on human judgement, therefore making the system vulnerable to human error and inconsistencies - leading to unfair or suboptimal picks. Another issue is that there is currently no way of accurately tracking the various player statistics needed to make an accurate judgement. These include number of games played, number of games available, batting skill, bowling skill, keeping skill and number of games missed. A record of the past and future availability of players should also be stored in order to be able to choose players for future games.

The fifth question told me what exactly the purpose and function of the current system is so that when a new system is created it still performs the necessary tasks that the user needs. I was able to gather more concise information on the current factors that the user considers when choosing players. These factors (listed in the interview summary) give me a clear idea on what things I will need to consider when designing a system to pick players. It also told me that the user wants to have an interface with which they can alter the selection suggested by the system – thus giving the user the benefits of an automated system while still maintaining the flexibility of manual selection.

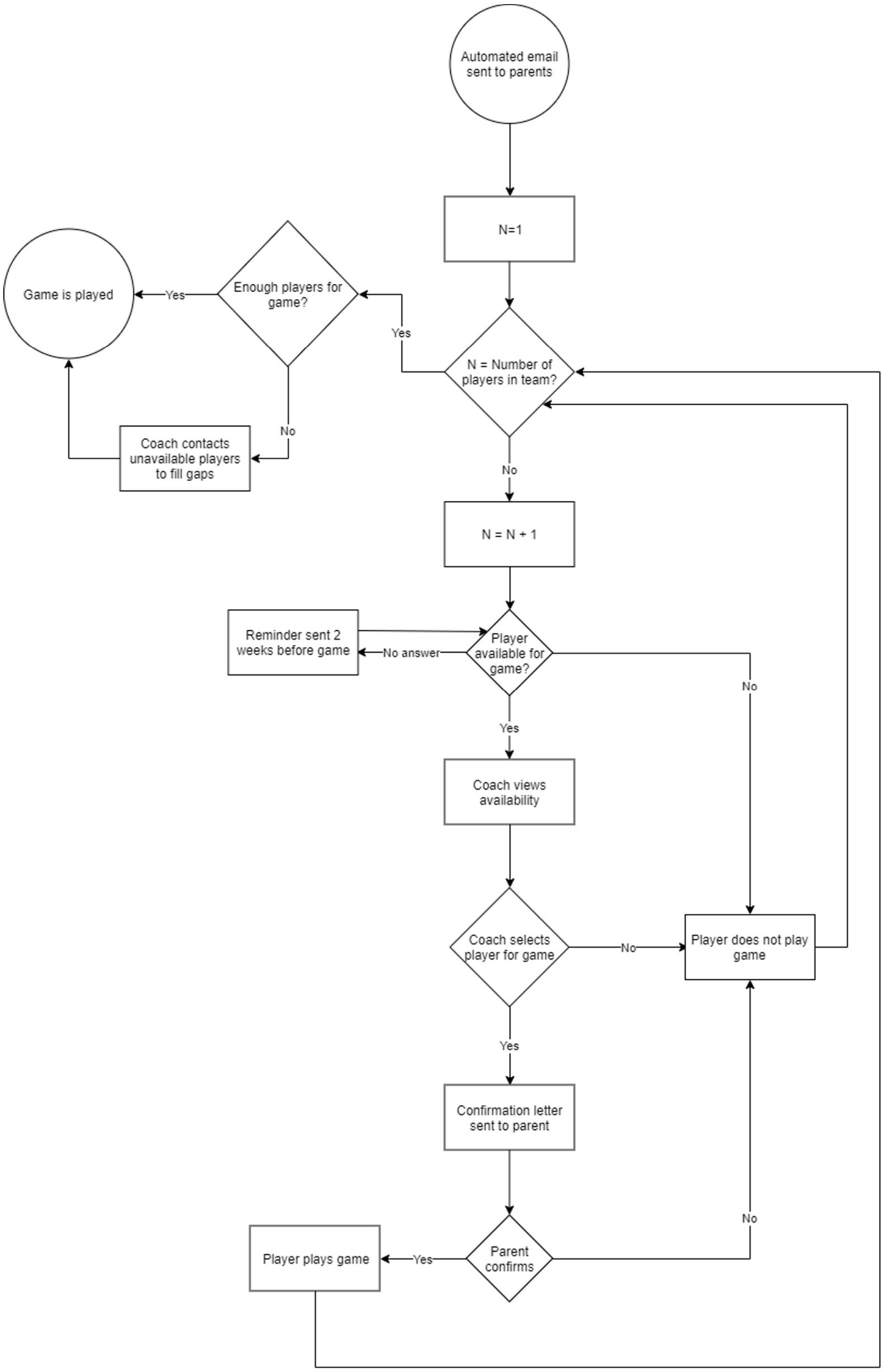
The Sixth question told me what new features (in addition to the current ones) should be incorporated into a new system. Combined with the previous question, I will have gained a good idea of what the whole system will need to do. One of the things that should be added is a database system in which player data should be stored. This will allow the user (and the system) to make informed decisions, supported by data and statistics. Email addresses will also need to be stored to facilitate email integration. An automatically generated email should be sent out to players/parents which will link to the database, therefore collecting and storing player availability data. This information will need to be combined with data from an API about upcoming matches so that informed decisions can be made. Joe also requested that it would be useful to have a feature whereby the user ranks each players performance for each game and then have the database record these scores over time, calculate a value for their performance level, and then use them for future picks. This question also gave me a better idea of how the stored data should be used when selecting players.

Interview summary

In conclusion I have gathered a lot of information for this interview and I now have a good idea of what kind of system the user wants. To summarise: There should be a database storing player information, email addresses and information about player ability. There should also be email integration with the database that collects data on what games players are available for. The system should be able to pick players based on data stored in the database and the ‘playcricket’ API and then present it to the user to make any changes, then update accordingly and perhaps give warnings when edits are made. There should be a feature that allows the user to enter player performance for each game and then store these scores in the database to create a record of player ability over time. This should all be presented through a user interface.

Current system flowchart

This is a flowchart for the process of selecting players for a single game. ‘Number of players in team’ refers to all the players eligible to play that particular game.

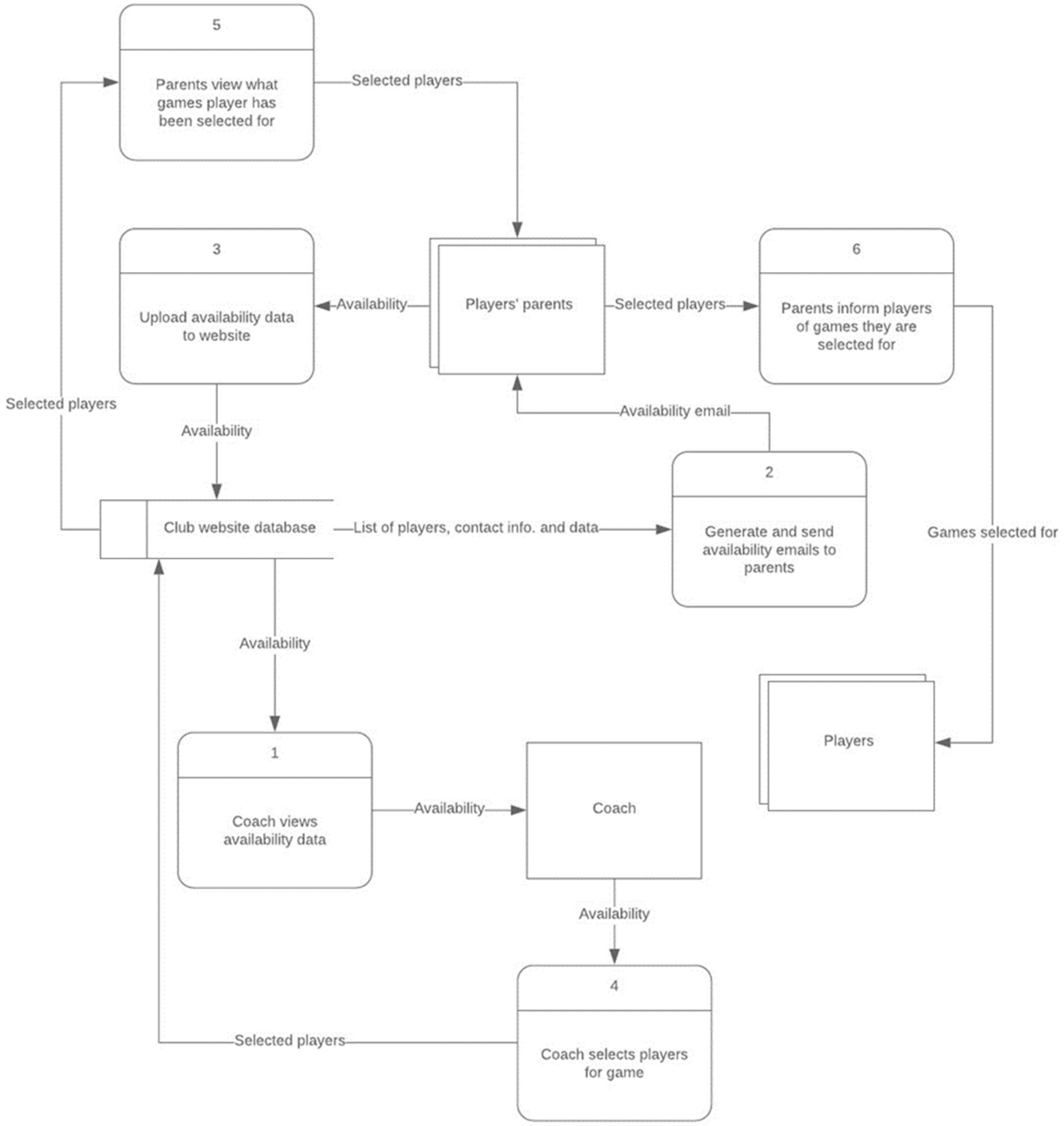

Automated email sent to parents
N=1
N = Number of players in team?
Yes
Enough players for game?
Yes
Game is played
No
Coach contacts unavailable players to fill gaps
No
N = N + 1
Player available for game?
No answer
Reminder sent 2 weeks before game
No
Yes
Coach views availability
Coach selects player for game
No
Player does not play game
Yes
Confirmation letter sent to parent
Parent confirms
No
Yes
Player plays game

IPSO chart for the current system

Inputs	Processes
• Player availability • Fixture details: o Date o League/Cup o Other team o Location • Player ability o Batting o Bowling o Wicket-keeping	• Coach views available players • Coach selects from available players to play in each game, based upon player availability and ‘number of games played’ • ‘number of games played’ updated for each player on database
Storage	Outputs
• Number of games played by each player • Number of games each player has been available for • Record of previous game results • Past player performance data	• Players that will play in each game • Player performance data • Player attendance and availability

Current system data flow diagram

This data flow diagram illustrates how data is exchanged between the website database, coach, parents and players and the processes involved.


5
Parents view what games player has been selected for
Selected players
3
Upload availability data to website
Availability
Players' parents
Selected players
6
Parents inform players of games they are selected for
Selected players
Availability
Availability email
2
Club website database
List of players, contact info. and data
Generate and send availability emails to parents
Games selected for
Availability
Players
1
Coach views availability data
Availability
Coach
Availability
4
Selected players
Coach selects players for game

Summary of user needs

The coach needs:

- A database to store player information, availability, contact info and performance data
- A system that generates suggested 'ideal' players for each game, based on:
    - Player availability
    - Player availability to games played ratio
    - Significance of the game
    - Player skill and ability
- A UI that:
    - presents these calculated picks to the user
    - allows them to make changes to the picks
    - presents them with player data to inform their decisions
    - allows them to enter player performance for each game
    - Stores this performance data in the database for calculating future player picks

Constraints

Technical:

A significant limitation of this project will be my programming, database and forms skills.

The solution will most likely be implemented using the object-oriented paradigm (OOP) and, although I do have some knowledge and experience of the OOP system, I have never worked on a project of this size and complexity. This means there will likely be programming concepts and techniques that I don't fully understand but will need for this project, however I am confident that I am sufficiently competent with OOP so that I will be able to develop my knowledge of it to the required level while working on this project and so I don't believe this will be a significant constraint.

I will need to create a fully functioning database and corresponding server so that it is accessible to the users. I do have a degree of understanding with creating, altering and retrieving with databases using SQL but I have never created a fully functional database of this scale and I have no experience with using databases on a remote server, however again I believe that my understanding is sufficient for me to fill in any gaps while working on the project but I will need to learn how to create and interact with database servers.

I will most likely be using a windows forms application for the user interface and I believe that my understanding of this is sufficient to implement the required features, so this should not be an issue.

From my research and observation, I have decided that it will not be viable to use the existing website database and email system as part of my solution and therefore I will need to provide these services in the new system.

User end:

The only requirements for the users to be able to use this system will be that the coach has access to a computer with internet access, and that the players/parents have access to their emails which I already know they do since the current system requires this.

Legal:

I will have to consider data protection and privacy laws as I will be using electronic database systems that will contain personal information such as email addresses. This means my database will need to be secure so that only authorised users can access it.

Proposed solution

The idea that I have come up with for the new system will use a MySQL remote database which will entirely replace the current website database. It will contain all the necessary player information, such as Name, parent's email, availability, number of games played, measures of performance/skill and players picked for each game.

The current emailing system will be replaced with one that directly links to the new database and allows parents to select which games players will be available for and then updates the database accordingly. This email will be sent out at the beginning of the season and contain all games for the upcoming season (the same way it is currently done). There will also be reminder emails sent at intervals before games if parents haven't submitted the availability for that game yet.

The system will ideally consider all games as far in advance as it has the necessary data for and calculate the ideal player selections for each game, with the objective of making fair picks. To do this it will consider how many games each player has been available for, how many games they have played so far, how important the game is and the ability of the player.

All information will be displayed to the coach through a windows forms application. It will show the suggested picks for a given game, with the ability to view statistics and information on each player. There will also be the ability to adjust and change these picks, with relevant information displayed to help inform the decisions.

Foreseeable challenges/issues:

The process of taking all the parameters and selecting players for each game will be one of the biggest challenges as it requires a way of selecting players over a set of games so that they are as fair as possible. This is made difficult by the fact that the system will have to consider that there are future games that it does not have the data for yet – since everyone will submit their availability at different times. There is also the issue of determining what parameters are the most significant, and how this affects the selections.

I must also be able to extract information on each upcoming game from the PlayCricket API and interpret this to gain a measure of significance of the game.

Current system data dictionary

Player Data:

Name	Description	Data type	RegEx

Player Forename		Varchar	
Player Surname		Varchar	
Contact Email	Parents' email for availability or contact	Varchar	
Player's team	Name of the team that the player plays for. E.g. 'U10s'	Varchar	
No. Games played	Number of games attended by the player	Int	
No. Games available	Number of games the player has been available for	Int	
Batting average	Total runs scored divided by total times out	Real	
Bowling average	Runs conceded divided by wickets taken	Real	

Match Data

Name	Description	Data type	RegEx
Team	Name of team that played match. E.g 'U10s'	Varchar	
Opponent	Name of opposing team in match. E.g. 'Aberdeen U11 A'	Varchar	
Win or loss		Boolean	
Date	Date match occurred	Date	
Type	Was the match a league, friendly or other type of match?	Varchar	
Runs	Number of runs scored by the team in the match	Int	
Wickets lost	Number of wickets conceded by the team in the match	Int	
Opponent's runs	Number of runs scored by the opponent in the match	Int	
Opponent's wickets lost	Number of wickets conceded by the opponent in the match.	Int	

Requirements

1. To be able to store player and match data
    1.1. All data to be stored on a database
    1.2. Stores relevant personal player data, such as player name and team
    1.3. Store player match statistics
        1.3.1. Stores past and future availability of players
        1.3.2. Stores players' past match performance
        1.3.3. Store number of games played and number of games available
    1.4. Store Availability of players for future fixtures

2. Collect and store player availability
    2.1. Automatically send an email to players requesting availability at the start of season
    2.2. The email can be easily responded to by clicking a link
    2.3. The email links to the database and automatically stores the availability data for each player for each game
    2.4. Acceptable responses to email: 'Available', 'Not available', 'Maybe' and 'No answer'
    2.5. At a fixed interval before each game, an email reminder email should be sent if a response has not been submitted for the game

3. Select players for games
    3.1. System can generate an ideal selection of players for each upcoming game
    3.2. The selection process aims to make fair picks – each player plays a number of games approximately proportional to the amount of games they have been available for
    3.3. The system should have a means of storing, producing and accounting for differences in player ability
    3.4. As well as making fair picks, the system should also select players according to their ability – so that better players play more difficult/significant games.
    3.5. The type of fixture (friendly, league, etc) should be accounted for when selecting players so that friendly games prioritise fair picks and competitive games prioritise high performing players
    3.6. When making selections, the system takes into account multiple factors:
        3.6.1. Ratio of number of games available to number of games played
        3.6.2. The type of game (League, friendly, etc)
        3.6.3. The team that the player is in
        3.6.4. The performance of the player over time (This should be calculated using the past performances of the player, with recent games having a more significant impact than older games)
    3.7. Fair picks should generally be prioritised over a strong team
    3.8. Generated selections are changeable and not fixed
    3.9. Data on future fixtures should be automatically retrieved via an API
    3.10. System should be able to pick players based on multiple upcoming games

4. There should be a user interface
    - 4.1. The interface should be easy and intuitive to use
    - 4.2. The interface should not require the user to enter SQL or VB.Net commands so that it is accessible and convenient to use
    - 4.3. The application should be accessible to only the coach
    - 4.4. The user should be able to view player data through the interface – including availability and player performance
    - 4.5. The program should present relevant data about fixtures to the user, including the ID, date, type and state
    - 4.6. For a future fixture: The program should present the user with a list of players playing in each fixture (or players assigned to play in a future fixture)
    - 4.7. For a future fixture: The program should present the user with a list of available players to select from
    - 4.8. For a past fixture: The program should present the user with the list of players that played in the game
    - 4.9. The program should tell the user what fixtures need data to be input
    - 4.10. The user should be able to enter player performances through the interface – and this data should be stored on the database
    - 4.11. The program should allow the user to search for fixtures
    - 4.12. The user should be presented with the picks generated by the system
    - 4.13. The user should be able to change the suggested picks through the interface
        - 4.13.1. The user should be given flexibility when adjusting selections – including the ability to assign players to older age-group games
        - 4.13.2. The changes made by the user should not be restricted and the system should update data accordingly when changes are made
        - 4.13.3. Relevant information should be presented to the user when they are adjusting selections, to inform their decisions
    - 4.14. Any data that is required to be entered by the user should be able to be entered through the interface
    - 4.15. The user should be able to add new players to the system through the interface
    - 4.16. The user should be able to add new fixtures to the system through the interface

New System Design

In this section I will create a comprehensive design of the solution that I will be implementing, and to do this I will break up the system into its main components and then specify the inputs, outputs and processes involved in each. I will be creating diagrams to describe the properties of the user interface and diagrams to fully represent the functions of the interface. I will also be designing the fully normalised database structure that will be required for this solution. Finally, I will be presenting the designs of the key algorithms used in this solution; primarily the selection algorithm, but also key SQL statements as well.

Storage and data inputs specification

In this section, I will present all the data that will stored in the system in the form of specification sheets that include all the necessary information on the data.

Volumetrics				
Document description	System	Document	Name	Sheet
Database	Cricket club team selection			
Stationery ref.	Size	Number of parts	Method of preparation	
Programme				

Filing sequence	Medium	Prepared by
	Digital database	Coach
Frequency of preparation	Retention period	Location of file

	Minimum	Maximum	Av/Abs	Growth rate/fluctuations
Volume	1	1		Depends of number of players and number of upcoming games

Users/receipts	Purpose	Frequency of use
Coach	Store necessary data such as player availability, match dates and contact information	

Data Dictionary

Ref	Name	Data Type	Length	Occurrence	Source of data
1	Player ID	Varchar	6	Per player	System
2	Player Forename	Varchar		Per player	User
3	Player Surname	Varchar		Per player	User
4	Player's team	Varchar		Per player	User
5	Team	Varchar		Per fixture	User
6	Opponent	Varchar		Per fixture	User
7	Date of game	Date		Per fixture	User
8	State	Integer		Per fixture	User

9	Availability	Integer		Per fixture per player	Parents/Players
10	Total runs scored	Integer		Per fixture	User
11	Total wickets taken	Integer		Per fixture	User
12	Total runs conceded	Integer		Per fixture	User
13	Runs Scored	Integer		Per player per fixture	User
14	Wickets taken	Integer		Per player per fixture	User
15	Runs conceded	Integer		Per player per fixture	User
16	Player position	Integer		Per player	User
17	Player rating	Double		Per player	User
18	Game rating	Double		Per player per fixture	User
19	Number of games played	Integer		Per player	System
20	Number of games available	Integer		Per player	System
21	Type of game	Integer		Per fixture	User
22	Number of players	Integer		Per fixture	User
23	Age group	Varchar		Per fixture	User
24	Fixture ID	Varchar	8	Per fixture	System

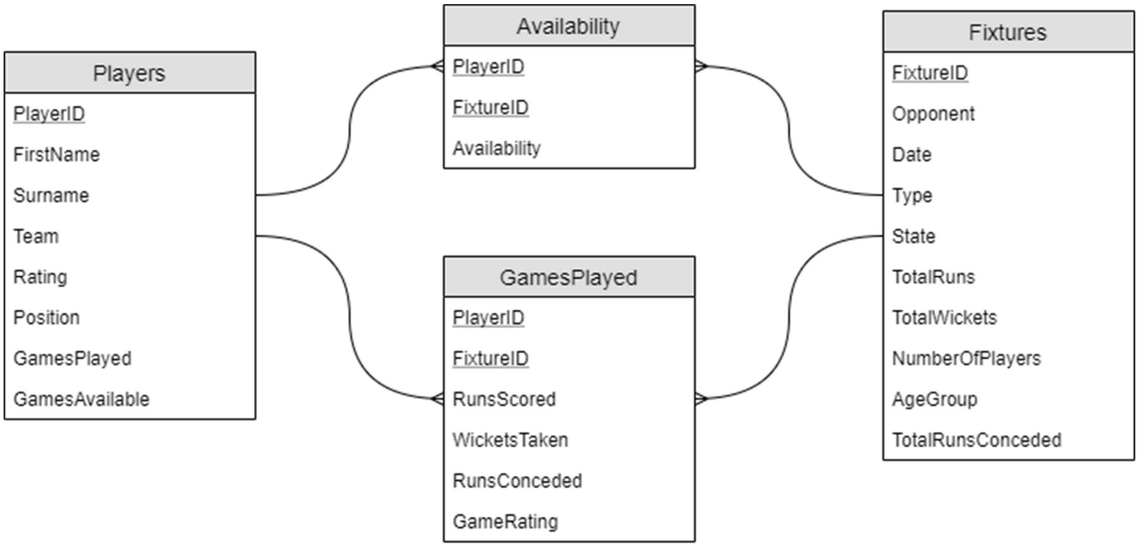
Database Design

This section includes all necessary information about the database and its design.

The database for this system is a web database on MySQL. This gives it flexibility over a local database as it is easier to access and manage from different devices. It is the sole database used in this system. The VB program uses the MySql.Data.MySqlClient library to make connections and queries to the database.

Database entity relationship diagram:

This is the structure of my database. ‘Players’ and ‘Fixtures’ are fairly straightforward, the former stores information on each individual player, and the latter stores information on each individual fixture. The other two tables, however, require some explanation since their distinct purposes are not initially obvious. The ‘Availability’ table contains a record for every player that is available to play in a fixture, whereas the ‘GamesPlayed’ table only stores the players that have been selected. I decided to do this because it will reduce the complexity of the queries needed and it will be much easier to work with, since the availability table will be much larger and potentially more difficult to work with than 'GamesPlayed'. 'GamesPlayed' is also used for storing the performance of the players in the fixtures they play in.



Key:

<u>Abc</u> = Primary key

Database DDL:

Table	**DDL**
Players	CREATE TABLE `players` (`PlayerID` varchar(6) NOT NULL, `FirstName` varchar(255) DEFAULT NULL, `Surname` varchar(255) DEFAULT NULL, `Team` varchar(255) DEFAULT NULL, `Rating` double NOT NULL DEFAULT 0, `Position` int(11) DEFAULT 0, `GamesPlayed` int(11) DEFAULT 0, `GamesAvailable` int(11) DEFAULT 0, PRIMARY KEY (`PlayerID`)
Fixture	CREATE TABLE `fixtures` (`FixtureID` varchar(8) NOT NULL, `Opponent` varchar(255) DEFAULT NULL, `Date` date DEFAULT NULL, `Type` smallint(6) DEFAULT 0, `State` smallint(6) DEFAULT 0, `TotalRuns` int(11) DEFAULT -1, `TotalWickets` int(11) DEFAULT -1, `NumberOfPlayers` int(11) DEFAULT 8, `agegroup` varchar(10) DEFAULT NULL, `TotalRunsConceded` int(11) DEFAULT -1, PRIMARY KEY (`FixtureID`)
GamesPlayed	CREATE TABLE `gamesplayed2` (`PlayerID` varchar(6) NOT NULL,

	`FixtureID` varchar(8) NOT NULL, `RunsScored` int(11) DEFAULT -1, `RunsConceded` int(11) DEFAULT -1, `WicketsTaken` int(11) DEFAULT -1, `GameRating` double DEFAULT -1, PRIMARY KEY (`PlayerID`,`FixtureID`)
Availability	CREATE TABLE `availability2` (`PlayerID` varchar(6) NOT NULL, `FixtureID` varchar(8) NOT NULL, `Availability` int(11) DEFAULT NULL, PRIMARY KEY (`PlayerID`,`FixtureID`)

SQL Queries:

This section will include the important SQL queries used in the system.

A pseudocode example of a simple, generic query to the database from the system (to showcase the connection method). This type of procedure is used to execute most SQL commands throughout the program, with the contents of 'SQLString' changing:

```
SQLString = "SELECT COUNT (FixtureID) FROM fixtures"
Try
    Open Connection
    Cmd = New MySqlCommand(SQLString, Connection)
    TotalGames = Cmd.ExecuteScalar
    Close Connection
Catch Exception
    [Exception message]
End Try
```

Example in VB:

```
    Sub CalculateTotalGames()
        Dim Cmd As MySqlCommand
        Dim SQLString As String = "SELECT COUNT(FixtureID) FROM fixtures"

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)

            Me.TotalGames = Cmd.ExecuteScalar

            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & " On General Info")
        End Try
    End Sub
```

Notable SQL statements:

The following queries demonstrate the commands used and the structure of the SQL in the program.

This SQL query selects players from the ‘players’ table that have been selected for a particular fixture. An ‘Inner Join’ is used to link the ‘players’ and ‘gamesplayed’ tables. The ‘Where’ and ‘Select from’ commands are also used:

```
"SELECT * FROM players INNER JOIN gamesplayed2 ON players.PlayerID = gamesplayed2.PlayerID
WHERE gamesplayed2.FixtureID = " & FixtureID
```

A similar query selects players that are available for a particular fixture, except the ‘Order by’ command is used as wel:

```
"SELECT * FROM players INNER JOIN availability2 ON players.PlayerID = availability2.PlayerID WHERE
availability2.Availability = 2 AND FixtureID = " & FixtureID & “ ORDER BY players.Surname ASC”
```

This statement is used to add a player to the ‘gamesplayed’ table, which means they will be playing in the fixture. It uses the ‘Insert Into’ command.

```
"INSERT INTO gamesplayed2 (PlayerID, FixtureID) VALUES (" & plr.PlayerID & ", " & FixtureID & ")"
```

This Statement is used after calculating the ‘GameRating’ of players and utilises the command ‘Update’:

```
"UPDATE gamesplayed2 SET GameRating = " & plr.GameRating & " WHERE PlayerID = " &
plr.PlayerID & " AND FixtureID = " & FixtureID
```

This statement is used to remove all players that are assigned to play in a fixture by using the ‘Delete From’ command:

```
"DELETE FROM gamesplayed2 WHERE FixtureID = " & FixtureID
```

This is a statement that counts the number of fixtures in the database, using the ‘Select count’ command:

```
"SELECT COUNT(FixtureID) FROM fixtures"
```

All SQL statements:

GeneralInfo:

```
"SELECT COUNT(FixtureID) FROM fixtures"
```

Player:

```
"SELECT RunsScored, RunsConceded, WicketsTaken, GameRating FROM gamesplayed2 WHERE PlayerID = " & Me.PlayerID & " AND FixtureID = " & FixtureID
```

```
"SELECT gamesplayed2.GameRating, fixtures.Date FROM gamesplayed2, fixtures WHERE gamesplayed2.FixtureID = fixtures.FixtureID AND gamesplayed2.PlayerID = " & PlayerID & " AND fixtures.Date < " & DateStr
```

Fixture:

```
"SELECT * FROM players INNER JOIN gamesplayed2 ON players.PlayerID = gamesplayed2.PlayerID WHERE gamesplayed2.FixtureID = " & FixtureID & ""
```

```
"SELECT * FROM players ORDER BY Surname ASC"
```

Report:

```
“SELECT * FROM fixtures “ & WhereClause & “ “ & “ORDER BY “ & Me.SortByString & " " & Me.SortOrderString & ""
```

GeneratedTeam:

```
"DELETE FROM gamesplayed2 WHERE FixtureID = " & FixtureID & ""
```

```
"SELECT * FROM players INNER JOIN availability2 ON players.PlayerID = availability2.PlayerID WHERE availability2.Availability = 2 AND FixtureID = " & FixtureID & " ORDER BY players.Surname ASC"
```

```
"INSERT INTO gamesplayed2 (PlayerID, FixtureID) VALUES (" & plr.PlayerID & ", " & FixtureID & ")"
```

CalculateMatchRatings:

```
"UPDATE gamesplayed2 SET GameRating = " & plr.GameRating & " WHERE PlayerID = " & plr.PlayerID & " AND FixtureID = " & FixtureID
```

Form1 (Upcoming fixtures):

```
"INSERT INTO fixtures (FixtureID, Opponent, Date, Type, agegroup) VALUES (“ & NewFixID & "', '" & TBox_Opponent.Text & "', '" & DateStr & "', " & CInt(TBox_Type.Text) & ", '" & TBox_Age.Text & "')"
```

Form2 (Past fixtures):

Form3 (View Fixture):

```
"SELECT Availability FROM availability2 WHERE FixtureID = " & SelectedFix.FixtureID & " AND PlayerID =" & plr.PlayerID
```

```
"DELETE FROM gamesplayed2 WHERE FixtureID = " & SelectedFix.FixtureID & ""
```

```
"UPDATE gamesplayed2 SET RunsScored = " & CInt(Tbox_RunsScored.Text) & ", RunsConceded = " & CInt(Tbox_RunsCon.Text) & ", WicketsTaken = " & CInt(Tbox_WicketsTaken.Text) & " WHERE PlayerID = '" & CurrentTeam(PlrIndex).PlayerID & "' AND FixtureID = '" & SelectedFix.FixtureID & "'"
```

```
"DELETE FROM gamesplayed2 WHERE FixtureID = " & SelectedFix.FixtureID & " AND PlayerID = " & PlrSwap(1)
```

```
"INSERT INTO gamesplayed2 (PlayerID, FixtureID) VALUES (" & PlrSwap(2) & ", " &
SelectedFix.FixtureID & ")"
```

Form4 (View Players):

```
"INSERT INTO players (PlayerID, FirstName, Surname, Team, Position) VALUES " & "('" & NewPlrID &
"', '" & TBox_PlrNameF.Text & "', '" & TBox_PlrNameS.Text & "', '" & TBox_Team.Text & "', " &
CInt(TBox_Pos.Text) & ")"
```

Form5 (Select Available Players):

```
"SELECT Availability FROM availability2 WHERE FixtureID = " & SelectedFix.FixtureID & " AND
PlayerID =" & plr.PlayerID
```

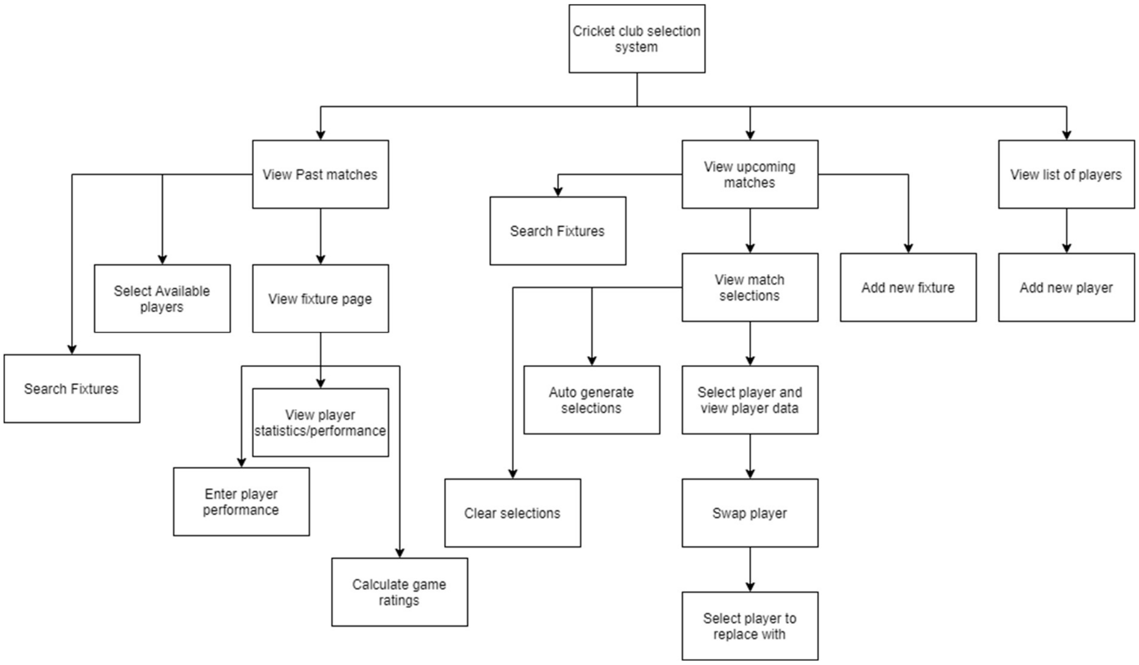
```
"DELETE FROM availability2 WHERE FixtureID = " & SelectedFix.FixtureID
```

```
"INSERT INTO availability2 VALUES (" & av.ThisPlayer.PlayerID & ", " & SelectedFix.FixtureID & ", " &
av.Available & ")"
```

System Design

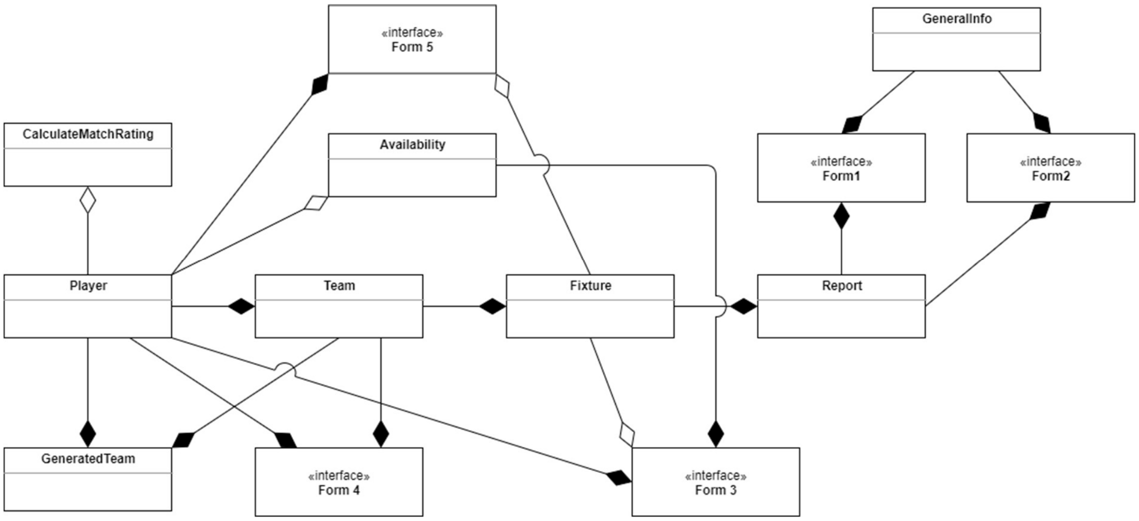
Top Down system chart

This top down diagram represents all the functions of the program as presented to the user. This means that there is a significant amount of processing and movement of data that occurs within many of these steps (notably the generation of lists of players that happens as part of the match selection process, which involves many complex steps that happen automatically), however the main purpose of this diagram is to present the top level functions of the program to the user, rather than explain the processes that underlie them.



System Object Model

This is a class diagram showing how the various classes relate to each other and also the interface. Composition and Aggregation are heavily utilised in this system model. There are five forms that constitute the interface. The four main classes used are Player, Team, Fixture and Report. The Classes 'CalculateMatchRating' and 'GeneratedTeam' contain procedures designed to produce specific outputs from algorithmic calculations. The class 'GeneralInfo' simply contains general data and a procedure that counts the number of fixtures in the database.



Classes

This section will elaborate on the object model and give detail on the construction of the individual classes

Player
Properties: PlayerID : String FirstName : String Surname : String Team : String Position : Integer Rating : Single GamesPlayed : Integer GamesAvailable : Integer RunsThisGame : Integer WicketsThisGame : Integer RunsConcededThisGame : Integer GameRating : Integer SelectionScore : Integer
Methods: AddPlayerPerformanceDB CalculateRating

This class represents an individual player. All the fields in the 'Players' table in the database are represented as properties in this class.

This class also contains the procedure that calculates the rating for the player .

Team
Properties: Squad : List(Of Player) CurrentTeam : List(Of Player)
Methods: FillTeam ReturnTeam : List(Of Player) GetAllPlayers CountPs : Integer CheckForEmptyStats : Boolean AddPlrStatsToPlrs ReturnPlayer : Player ReturnSquad : List(Of Player) RefreshPlrRatings

This class is not represented in the database and is only used in the system object model. It has two properties: Squad – which is a list of all the players in the database, and CurrentTeam – which only contains players in the team.

GetAllPlayers and FillTeam are methods that get players from the database and add them to lists of players in the system. See SQL statements.

Other methods are self-explanatory except AddPlrStatsToPlrs which gets the player performance data from the database, given a fixture

Fixture
Properties: ThisTeam : Team AgeGroup : Integer FixtureID : String Opponent : String FixtureDate : Date State : Integer

The fixture class represents the fixture table in the database and contains the fields from the database as properties.

This class has no methods.

TotalRunsScored : Integer
TotalWicketsTaken : Integer
TotalRunsConceded : Integer
NumberOfPlayersRequired : Integer
Type : Integer

Methods:

Report

Properties:
SortByString : String
SortOrderString : String
Fixtures : List(Of Fixture)

Methods:
GenHTML : String
GetFixtures

The Report class is used to generate the tables seen on the forms interface. The Report can be altered to generate past or future fixtures, as well as searching for fixtures and sorting by different fields.

GetFixtures gets the fixtures from the database, subject to the conditions specified in the parameters.

GeneratedTeam

Properties:
ThisTeam : Team
FixtureID : String
EligiblePlayers : List(Of Player)
SortList : List(Of Player)
ExistingTeam : List(Of Player)

Methods:
CreateNewTeam
CheckExistingTeam
DeleteExistingTeamDB
GetEligiblePlayers
SelectPlayers
QuickSortScores
ReturnSortList : List(Of Player)
RemoveExcessPlayers
WriteLineUpToDB
ReturnEligiblePlayers : List(Of Player)

The GeneratedTeam class is used when the user clicks the generate team button on the fixture page. If contains all the necessary methods to automatically generate fair picks for the fixture and upload them to the database. See Main selection algorithm and rating calculation for more detail.

CalculateMatchRatings

Properties:
CurrentTeam : List(Of Player)
TotalRuns : Integer
TotalWickets: Integer
TotalRunsConceded : Integer
FixtureID : String

Methods:
CalculateRatings
UploadMRatingDB

CalculateMatchRatings contains all the methods necessary to calculate the MatchRating for each player in a fixture. This is used when calculating player ratings.

ReturnTeam : List(Of Player)

Availability

Properties:
ThisPlayer : Player
Available : Integer

Methods:

The Availability class simply contains a player and an integer to specify if that player is available.

GeneralInfo

Properties:

Methods:
CalculateTotalGames

This Class contains the method CalculateTotalGames which returns the number of fixtures stored in the database

Main selection algorithm and rating calculation

One of the main functions of this system is to automatically generate an ‘ideal’ team for each fixture. An ideal team is one that is as fair as possible for the players and yet suits the nature of the fixture being played.

There are two main calculations that are performed when selecting players for a fixture:

- Calculating the ‘Rating’ of a player
- Selecting players for a fixture

The rating of the player is an important factor influencing which players are picked, and is a value that represents the players performance over time

Part 1: Calculating player ratings

1.1 Game Ratings:

In order to come up with an overall judgement of a player’s ability, I will firstly calculate the individual performances of players for each game and then I will combine these performances to generate a number that represents the player’s performance over time.

All the procedures involved in calculating the ‘GameRating’ values are encapsulated within the class ‘CalculateMatchRatings’. The fundamental concept is that the GameRating depends on the performance of the player compared with the total performance of the team as a whole. ‘CalculateMatchRatings’ is passed the list of players from the game as a parameter, as well as the total runs scored, total wickets taken, and total runs conceded.

The GameRating of a player is dependent on:

- The runs they scored
- The total runs scored by the team
- The wickets they hit
- The total wickets hit by the team
- The runs they conceded
- The total runs conceded by the team

This is a pseudocode representation of the algorithm that will be used in the program (The actual mathematical function is highlighted):

```
ScoreMultiplier = (50 * NumberOfPlayersInTeam)

For each player as player in CurrentTeam

    Player.GameRating =

     ((player.RunsThisGame / TotalRuns) +

    (player.WicketsThisGame / TotalWickets * 2) +

    (0.5 * (1 – (player.RunsConcededThisGame / TotalRunsConceded)))) *

    ScoreMultiplier * 0.25

Next
```

A general formula representing the calculation:

$X = (a/b) + (c/2d) + (0.5(1-(e/f))) * 0.25g$

Once the GameRating has been calculated, it is uploaded to the database so that it can be used when calculating the player ratings

1.2 Player Ratings:

Now that the performance of each player in each game has been calculated and is available for use from the database, they can be used to calculate a value that is representative of the players performance across games, over time. Instead of simply taking an average of all the player's GameRatings, I decided to bias the overall Rating towards games that have occurred more recently. This rating will change with each game played by the player; with good performances raising the rating and bad performances lowering it. The effect of specific fixtures on the player's overall rating will decrease over time.

Since the GameRating values are held on the database, the formula is situated inside a database connection. This means that in the program this formula is situated within a database connection sub procedure, but for the sake of clarity I have replaced these values (that are usually retrieved by a MySqlDataReader) with standard variables.

```
For each Fixture in ListOfPlayedFixtures
    TotalScore += GameRating / Math.Sqrt(DateDiff(DateInterval.Day, Date.Today, GameDate))
    Divisor += 1 / Math.Sqrt(DateDiff(DateInterval.Day, Date.Today, GameDate))
Next
Rating = TotalScore / Divisor
```

What this formula does is calculate the sum of the GameRatings from each fixture played by the player, except each GameRating is divided by the square root of the number of days ago the fixture was played. The divisor is needed because in order to calculate an average, it is insufficient to divide by the number of fixtures as this would cause past fixtures to unfairly detract from the players ratings. Instead the 'TotalScore' is divided by the sum of the reciprocals of the days since each fixture. This way I can calculate an average as well as bias the score towards more recent fixture performances.

The Player rating is dependent on the GameRating and number of days since the game was played, for each fixture.

Part 2: Selecting players

The purpose of this part of the system is to generate as fair picks as possible, and is contained within the class 'GeneratedTeam'. There are three main steps involved:

1. Get a list of all the eligible players to be picked
2. Calculate the 'fairness' of each player to be picked for the fixture
3. Create an ordered list of players and select the required number off the top

2.1 Generating a list of eligible players

Before any calculations can be made, the system must have a list of players to select from. The sub procedure 'GetEligiblePlayers' executes the SQL query:

```
"SELECT * FROM players INNER JOIN availability2 ON players.PlayerID = availability2.PlayerID WHERE availability2.Availability = 2 AND FixtureID = " & FixtureID & " ORDER BY players.Surname ASC"
```

Which is followed by a loop that adds each player that is in the 'Availability' table to a list of the object 'Player'. Only the players that are available for the fixture are added to the list, and so it is impossible for a player who isn't available for the fixture to be selected by the system

2.2 Calculating the order of priority of the players to be picked

Now that the system has a list of eligible players, the system needs to calculate which players should be chosen. The way I decided to do this is to calculate a 'SelectionScore' value for each player – which represents how fair it is for that player to play in the game. 'Fair picks' refer favouring players who have a low 'GamesPlayed' to 'GamesAvailable' ratio, and therefore haven't played many games relative to the amount they 'signed up' for.

There are two separate formulae used for calculating selection scores: One is for friendly games and one is for 'league' (competitive) games. The formula for friendly games does not take the players rating into account; meaning that the selections are designed to be as fair as possible without considering the players skill. The second formula is for competitive games, and this formula considers player Rating and fairness of picks, although player rating takes priority. Below is the pseudocode representation of this algorithm:

Highlighted in yellow is the formula for 'friendly' games

Highlighted in green is the formula for 'league' games

```
Select Case FixtureType
    Case 'Friendly'
        For each player in EligiblePlayers
            If Player.GamesPlayed = 0 Or Player.GamesAvailable = 0 Then
                Player.SelectionScore = 100
            Else
                Player.SelectionScore =
                100 / (Player.GamesPlayed / Player.GameAvailable)
            End If
        Next
    Case 'League'
        For each Player in EligiblePlayers
            If Player.GamesPlayed = 0 Or Player.GamesAvailable = 0 Then
                Player.SelectionScore = (Player.Rating ^ 2) / 100
            Else
                Player.SelectionScore =
                (Player.Rating ^ 2) / (100 * (Player.GamesPlayed /
                Player.GamesAvailable))
            End If
        Next
    End Select
```

After this sub procedure, each eligible player has a 'SelectionScore' assigned to them, indicating how fair it would be for them to play in that particular fixture.

2.3 Create an ordered list of players and select the required number off the top

Now that each player has been evaluated, the most eligible players need to be selected. To achieve this I decided to sort the list of players with respect to their 'SelectionScore'. I chose to use a recursive quicksort for this. Here is the pseudocode representation:

```
Sub QuickSortScores(min, max)
    RandNum = New Random
    Mid as integer
    Top as integer
    Bot as integer
    i as integer
    MidPlayer as player

    If Min > Max Then Exit Sub
    i = RandNum.Next(Min, Max + 1)
    MidPlayer = PlayerList(i)
    mid = PlayerList(i).SelectionScore
    PlayerList(i) = PlayerList(min)
    Bot = Min
    Top = Max

    Do
        Do While PlayerList(Top).SelectionScore >= mid
        Top = Top + 1
        If Top <= Bot Then Exit Do
        Loop
        If Top <= Bot Then
            PlayerList(Bot) = MidPlayer
            Exit Do
        End If

        PlayerList(Bot) = PlayerList(Top)
        Bot = Bot + 1

        Do While PlayerList(Bot).SelectionScore < Mid
            Bot = Bot + 1
            If Bot >= Top Then Exit Do
        Loop
        If Bot >= Top Then
            Bot = Top
            PlayerList(Top) = MidPlayer
            Exit Do
        End If
        PlayerList(Top) = PlayerList(Bot)
    Loop
    QuickSortScores(Min, Bot – 1)
    QuickSortScores(Bot + 1, Max)
End Sub
```

Once the QuickSort is finished, the system has a list of eligible players in order of their ‘SelectionScores’ and so all it needs to do to extract an ‘ideal’ team is take the required number of players from the top of the list. This is executed by the sub procedure ‘RemoveExcessPlayers’:

```
Sub RemoveExcessPlayers(NumPlayers)
    TeamList = New List(Of Player)
    If EligiblePlayers.Count < NumPlayers Then
        MsgBox(“Error, not enough eligible players”)
    Else
        For i = 0 To NumPlayers – 1
            TeamList.Add(EligiblePlayers(i))
        Next
    End If
    EligiblePlayers.Clear()
    EligiblePlayers = TeamList
End Sub
```

The system has now finished generating the team, which is in the form of a list of objects of type ‘Player’

Other Notable Algorithms and Procedures

Pseudocode representations of notable algorithms and procedures in the program

A linear search from the class ‘team’:

```
Function ReturnPlayer(PlayerIDToFind as string)
    SearchedPlayer = Nothing
    For Each Player In Squad
        If Player.PlayerID = PlayerIDToFind Then
            SearchedPlayer = Player
        End If
    Next
    Return SearchedPlayer
```

```
End Function
```

A part of a sub procedure from Form1 that generates a FixtureID for a new Fixture:

```
NewFixID = “”
NewFixDate = CDate(TBox_Date.Text)
NewFixID &= NewFixDate.Day.ToString(“00”)
NewFixID &= NewFixDate.Month.ToString(“00”)
NewFixID &= Mid(NewFixDate.Year.Tostring, 3)
Count = 0
For Each Fixture in FixReport.Fixtures
        If Fixture.FixtureDate = NewFixDate Then
                Count = Count + 1
        End IF
Next
Count = Count + 1
NewFixID &= Count.ToString(“00”)
```

A part of a sub procedure from Form4 that generates a PlayerID for a new player:

```
NewPlrID = “”
NewPlrID &= Mid(Date.Today.Year, 3)
Count = 0
For Each Player In CurrentSquad
        Count = Count + 1
Next
Count = Count + 1
IndexStr = Count.ToString(“0000”)
NewPlrID &= IndexStr
```

Interface structure and design

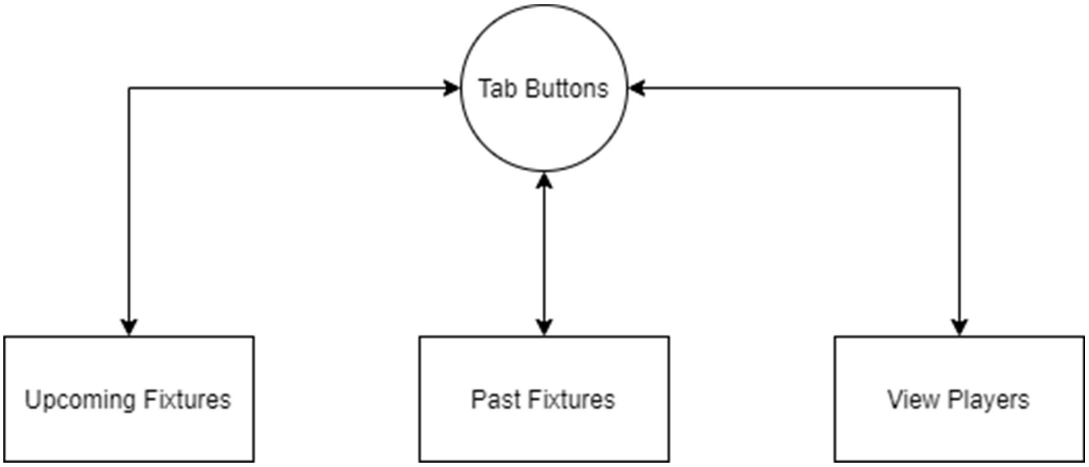
Navigation Diagrams

Key:

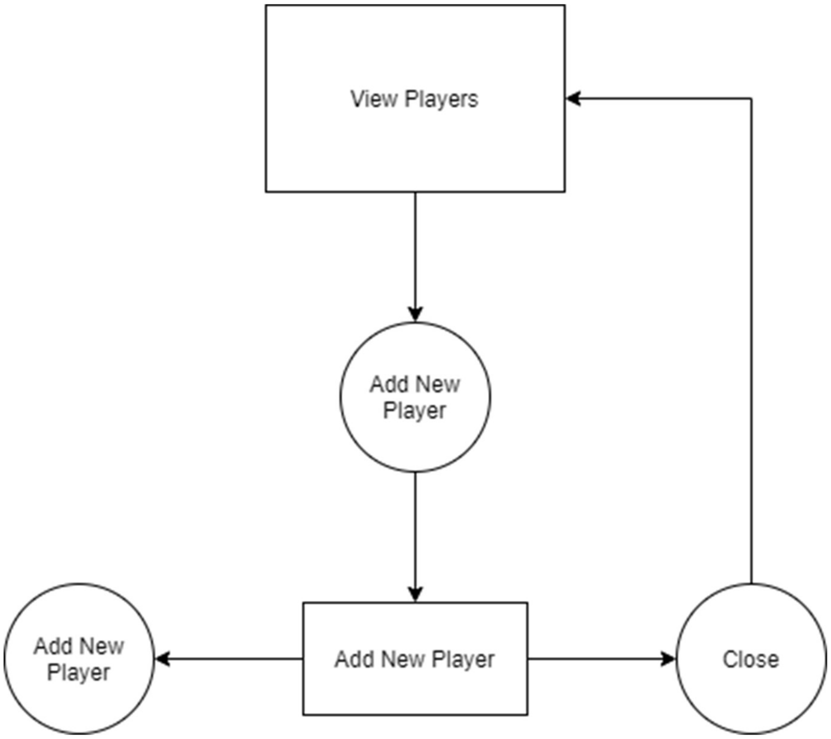
○ = Button

□ = Window/Panel

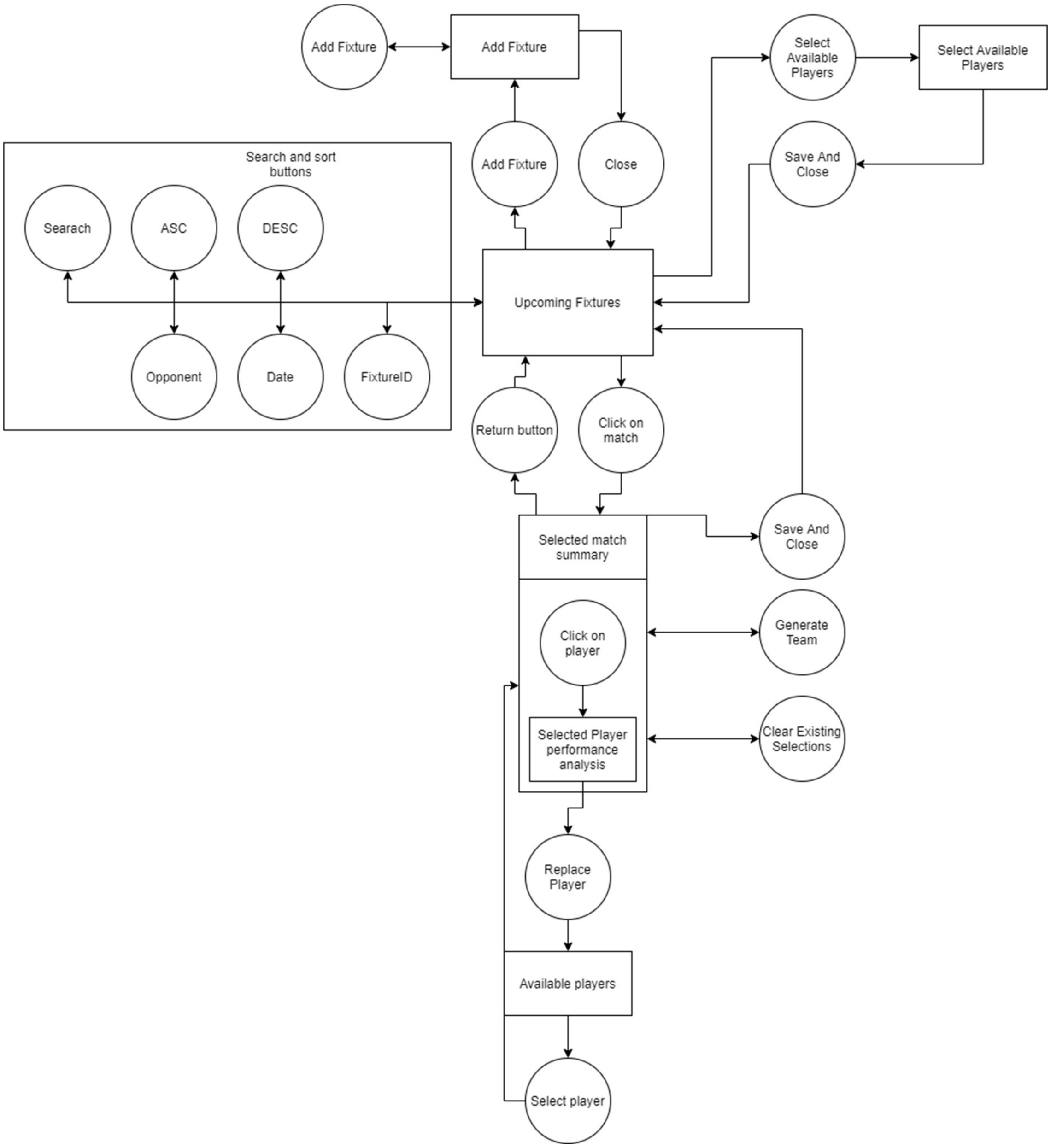
Top-Level Navigation diagram:



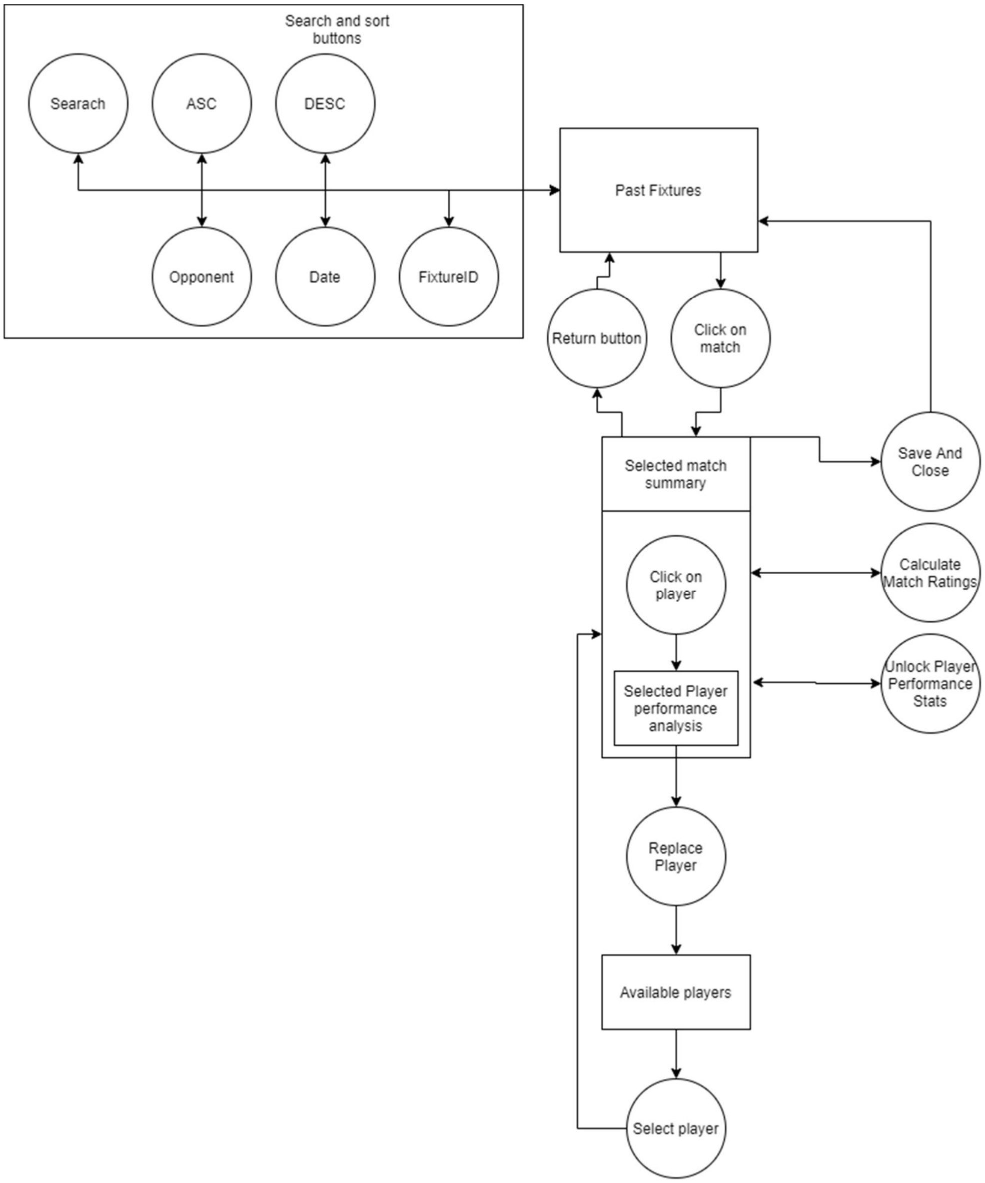
View Players:



Upcoming Fixtures:



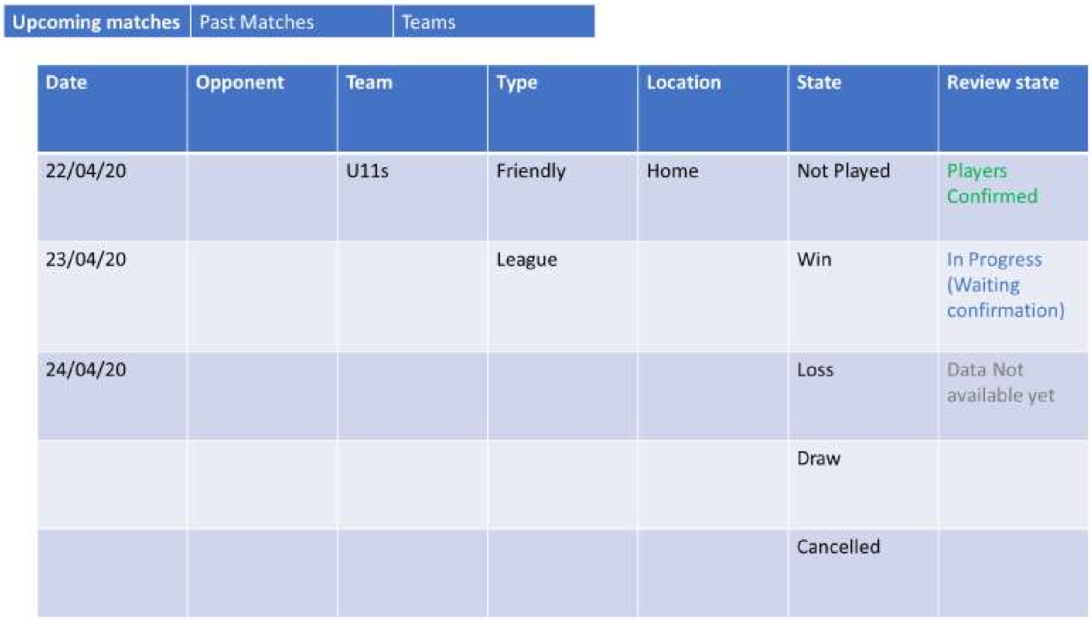
Past Fixtures:



Interface Mock-Ups

These give a general Idea of what the user Interface will look like

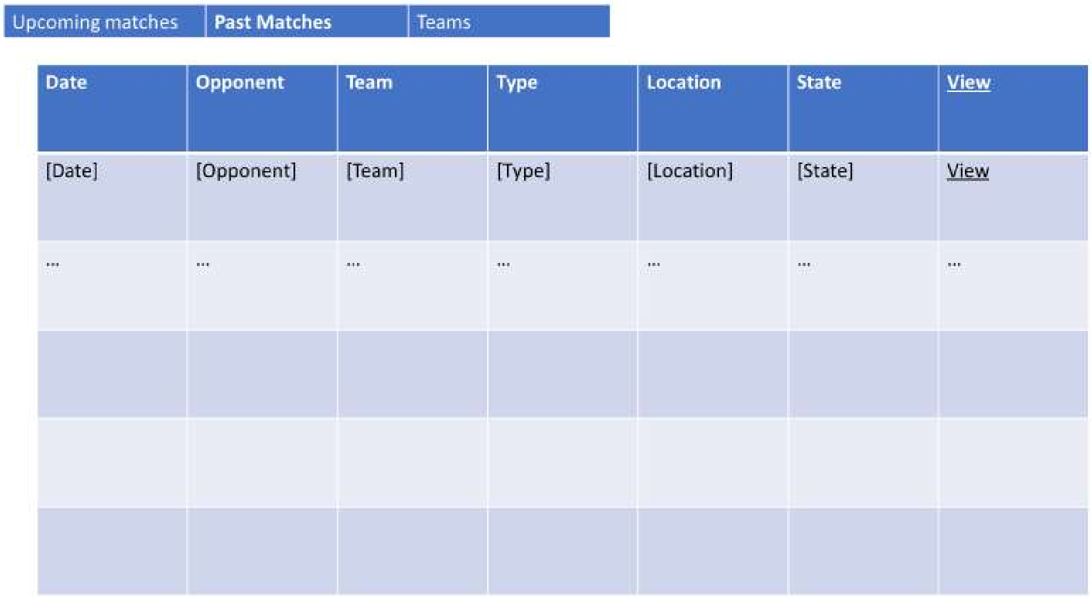
Upcoming matches (default page):



Upcoming matches | Past Matches | Teams

Date	Opponent	Team	Type	Location	State	Review state
22/04/20		U11s	Friendly	Home	Not Played	Players Confirmed
23/04/20			League		Win	In Progress (Waiting confirmation)
24/04/20					Loss	Data Not available yet
					Draw	
					Cancelled	

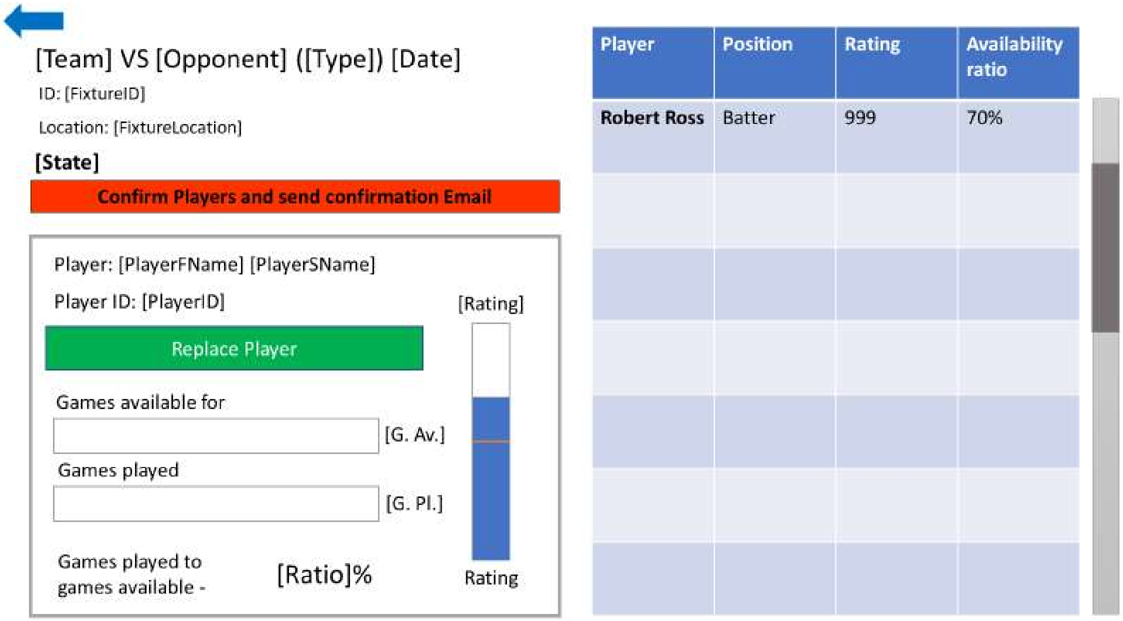
Past Matches:



Upcoming matches | **Past Matches** | Teams

Date	Opponent	Team	Type	Location	State	View
[Date]	[Opponent]	[Team]	[Type]	[Location]	[State]	View
…	…	…	…	…	…	…

View match:



[Team] VS [Opponent] ([Type]) [Date]

ID: [FixtureID]

Location: [FixtureLocation]

[State]

Player: [PlayerFName] [PlayerSName]

Player ID: [PlayerID]

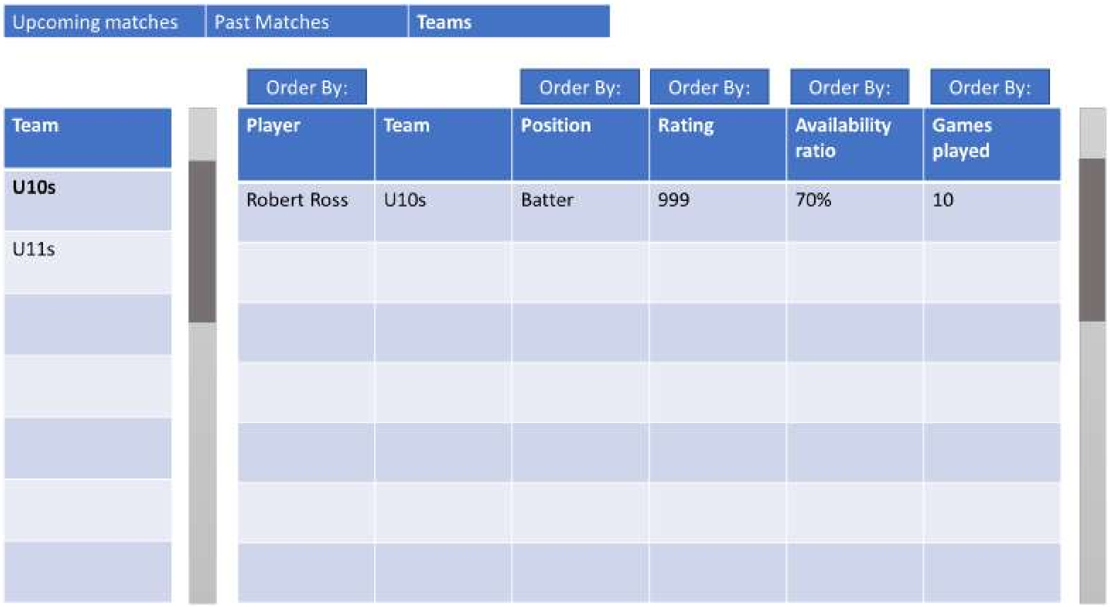
Games available for [G. Av.]

Games played [G. Pl.]

Games played to games available - [Ratio]%

Player	Position	Rating	Availability ratio
Robert Ross	Batter	999	70%

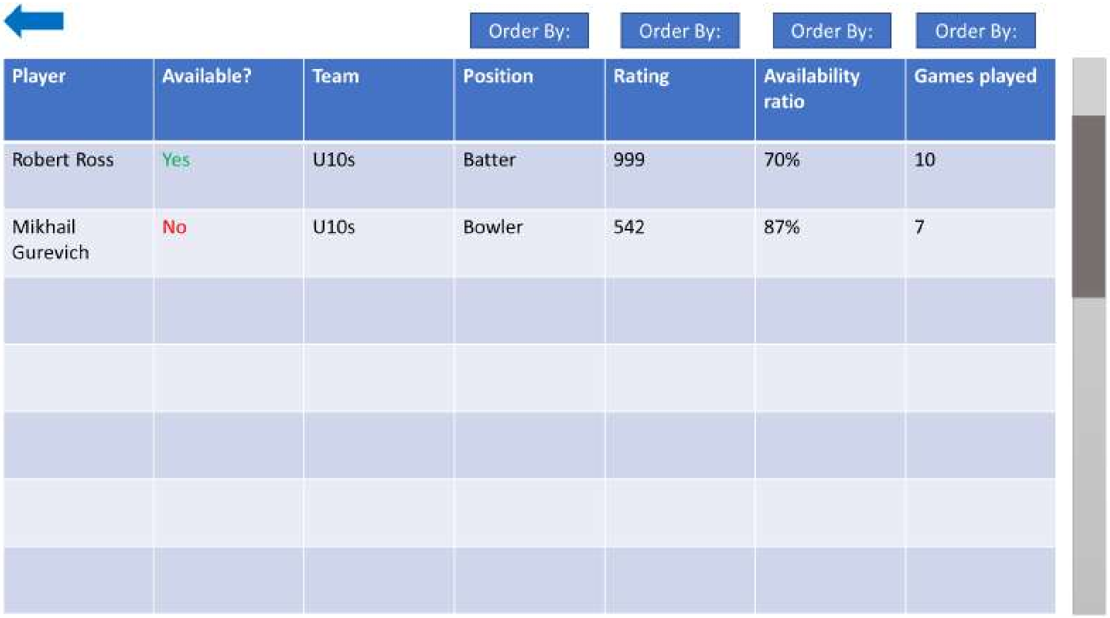
Teams:



Team
U10s
U11s

Player	Team	Position	Rating	Availability ratio	Games played
Robert Ross	U10s	Batter	999	70%	10

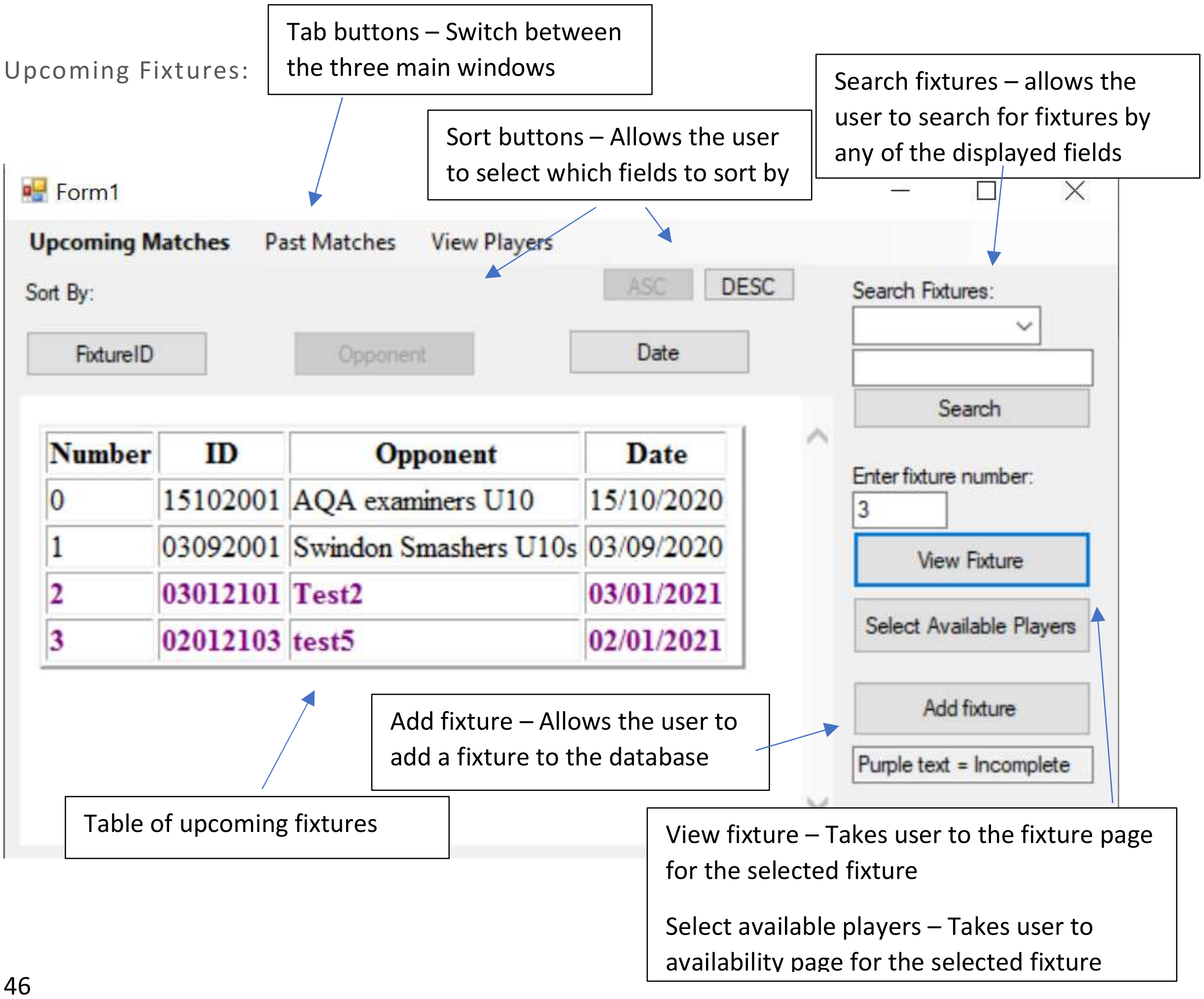
Replace player:



Player	Available?	Team	Position	Rating	Availability ratio	Games played
Robert Ross	Yes	U10s	Batter	999	70%	10
Mikhail Gurevich	No	U10s	Bowler	542	87%	7

Actual User Interface (Revised)

Upcoming Fixtures:



Number	ID	Opponent	Date
0	15102001	AQA examiners U10	15/10/2020
1	03092001	Swindon Smashers U10s	03/09/2020
2	03012101	Test2	03/01/2021
3	02012103	test5	02/01/2021

Tab buttons – Switch between the three main windows

Sort buttons – Allows the user to select which fields to sort by

Search fixtures – allows the user to search for fixtures by any of the displayed fields

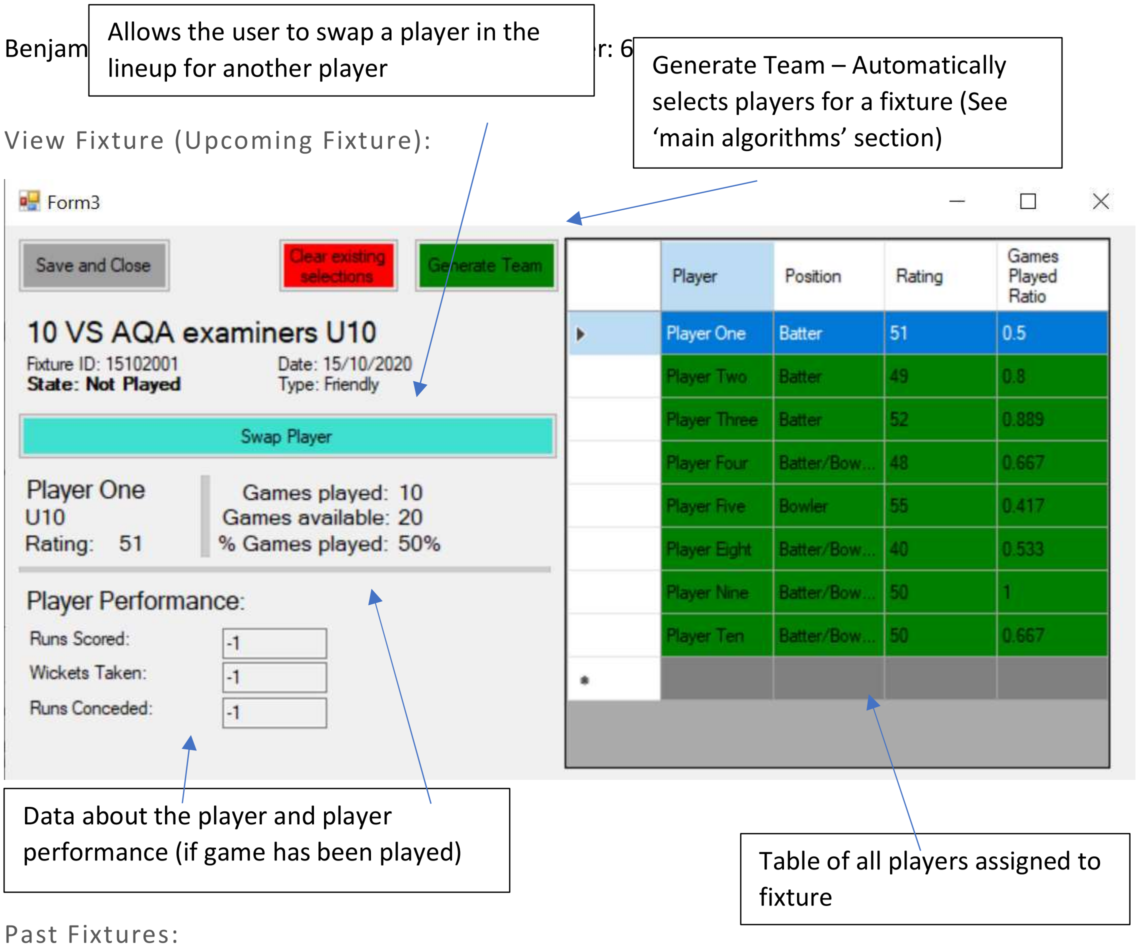
Add fixture – Allows the user to add a fixture to the database

Table of upcoming fixtures

View fixture – Takes user to the fixture page for the selected fixture

Select available players – Takes user to availability page for the selected fixture

View Fixture (Upcoming Fixture):



Allows the user to swap a player in the lineup for another player

Generate Team – Automatically selects players for a fixture (See 'main algorithms' section)

10 VS AQA examiners U10

Fixture ID: 15102001
State: Not Played
Date: 15/10/2020
Type: Friendly

Player One
U10
Rating: 51

Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

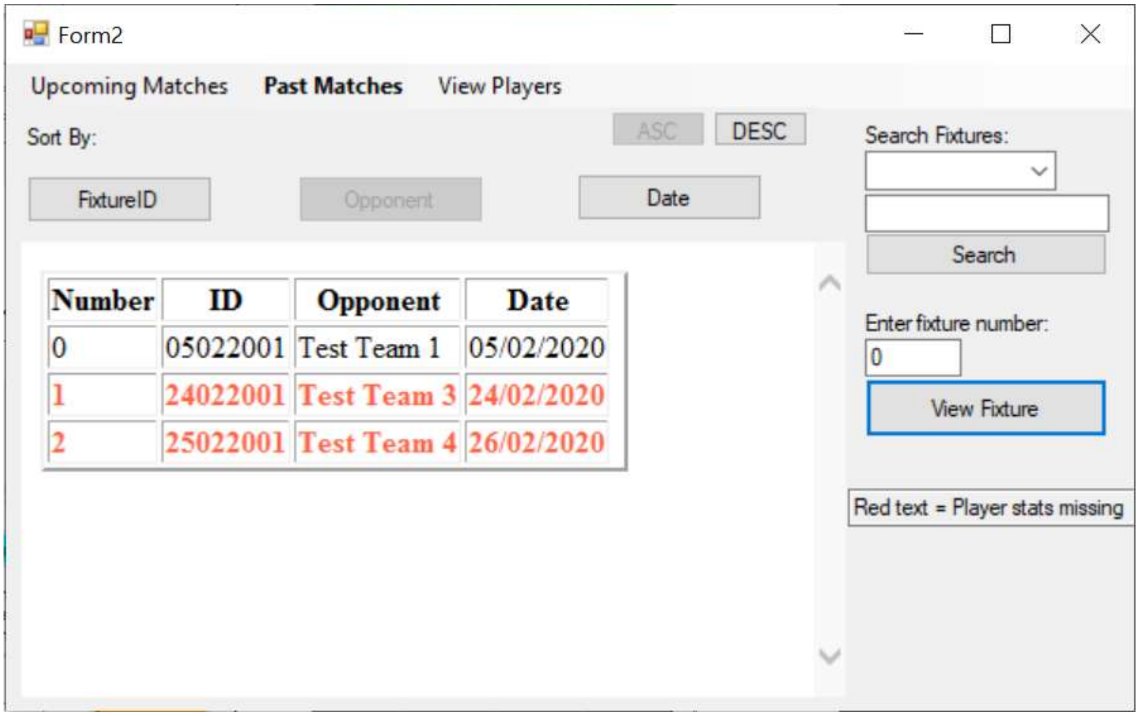
Runs Scored: -1
Wickets Taken: -1
Runs Conceded: -1

Player	Position	Rating	Games Played Ratio
Player One	Batter	51	0.5
Player Two	Batter	49	0.8
Player Three	Batter	52	0.889
Player Four	Batter/Bow...	48	0.667
Player Five	Bowler	55	0.417
Player Eight	Batter/Bow...	40	0.533
Player Nine	Batter/Bow...	50	1
Player Ten	Batter/Bow...	50	0.667

Data about the player and player performance (if game has been played)

Table of all players assigned to fixture

Past Fixtures:



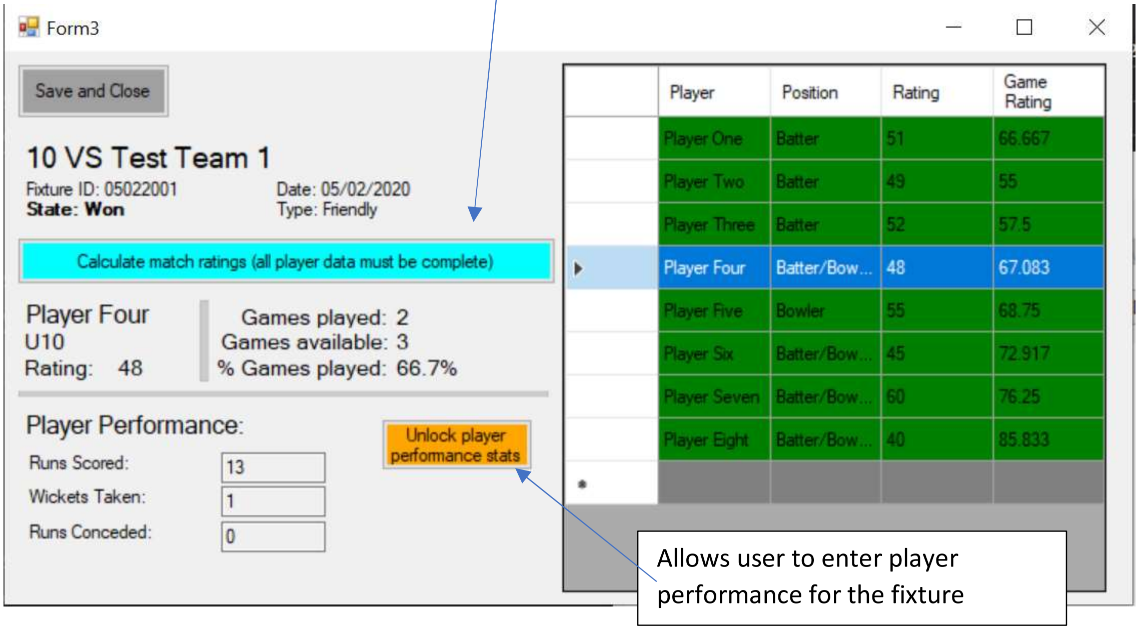
Number	ID	Opponent	Date
0	05022001	Test Team 1	05/02/2020
1	24022001	Test Team 3	24/02/2020
2	25022001	Test Team 4	26/02/2020

Enter fixture number: 0

Red text = Player stats missing

Calculates a match rating for each player in the fixture (See ‘main algorithms’ section)

View Fixture (Past Fixture):


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 1

Fixture ID: 05022001
State: Won
Date: 05/02/2020
Type: Friendly

Player Four
U10
Rating: 48

Games played: 2
Games available: 3
% Games played: 66.7%

Player Performance:

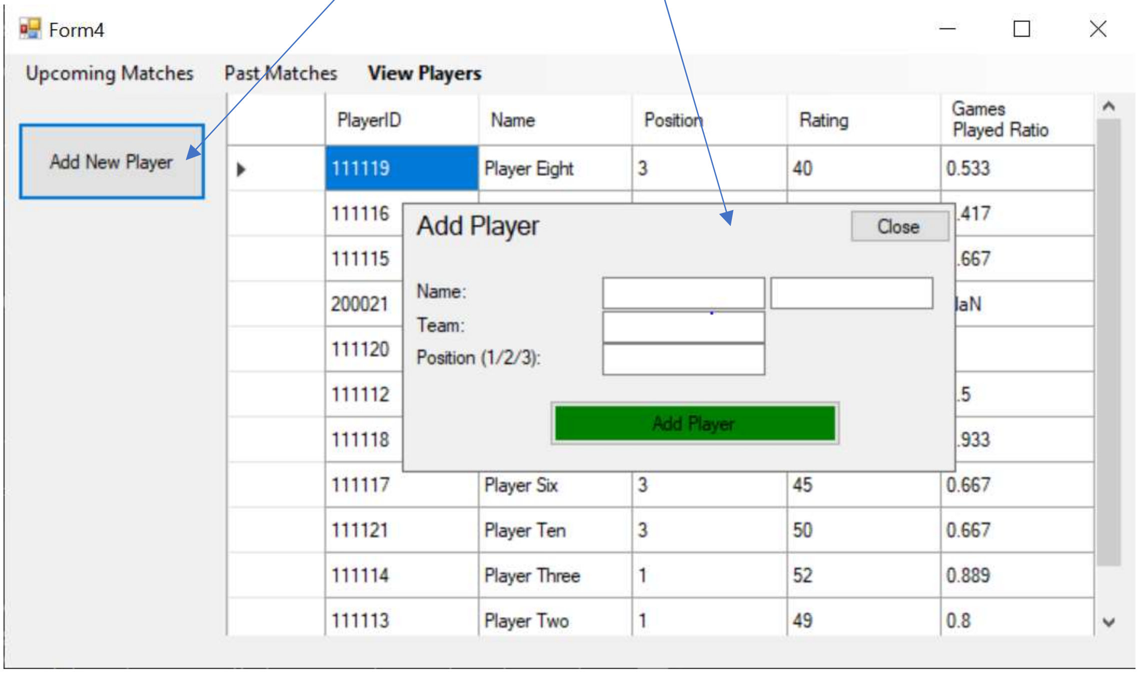
Runs Scored:	13
Wickets Taken:	1
Runs Conceded:	0

	Player	Position	Rating	Game Rating
	Player One	Batter	51	66.667
	Player Two	Batter	49	55
	Player Three	Batter	52	57.5
▸	Player Four	Batter/Bow...	48	67.083
	Player Five	Bowler	55	68.75
	Player Six	Batter/Bow...	45	72.917
	Player Seven	Batter/Bow...	60	76.25
	Player Eight	Batter/Bow...	40	85.833
*				

Allows user to enter player performance for the fixture

Enables user to enter a new player into the system

View Players:



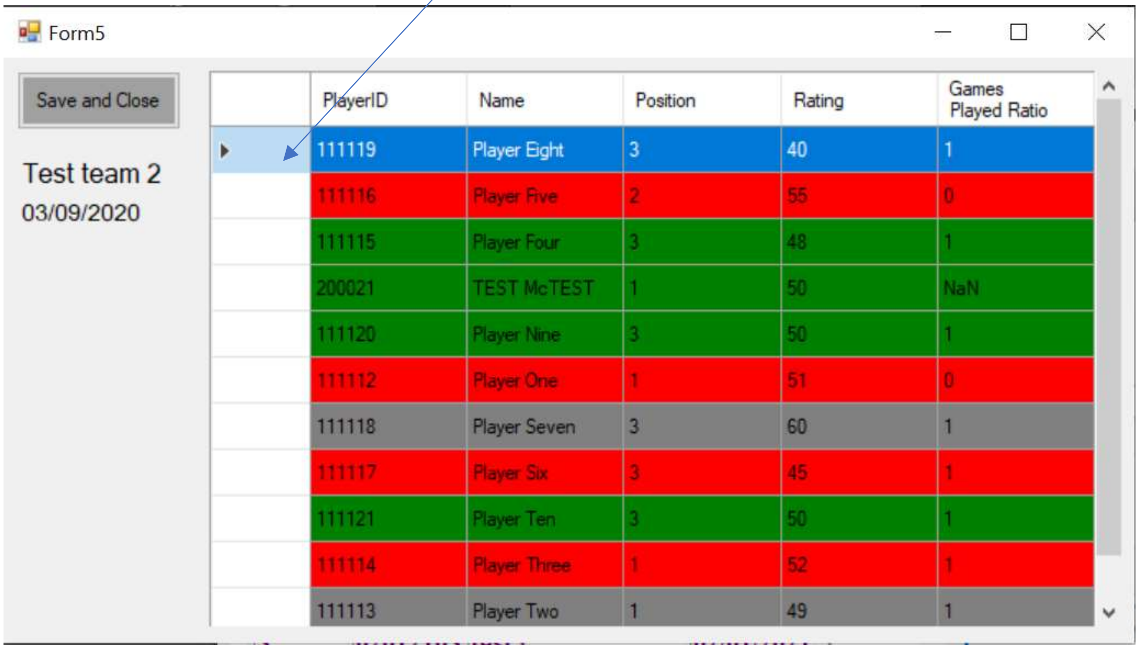
	PlayerID	Name	Position	Rating	Games Played Ratio
▸	111119	Player Eight	3	40	0.533
	111116				.417
	111115				.667
	200021				laN
	111120				
	111112				.5
	111118				.933
	111117	Player Six	3	45	0.667
	111121	Player Ten	3	50	0.667
	111114	Player Three	1	52	0.889
	111113	Player Two	1	49	0.8

Add Player

Name:
Team:
Position (1/2/3):

Allows the user to select which players are available (green), unavailable (red), or unknown (grey)

Select Available players:

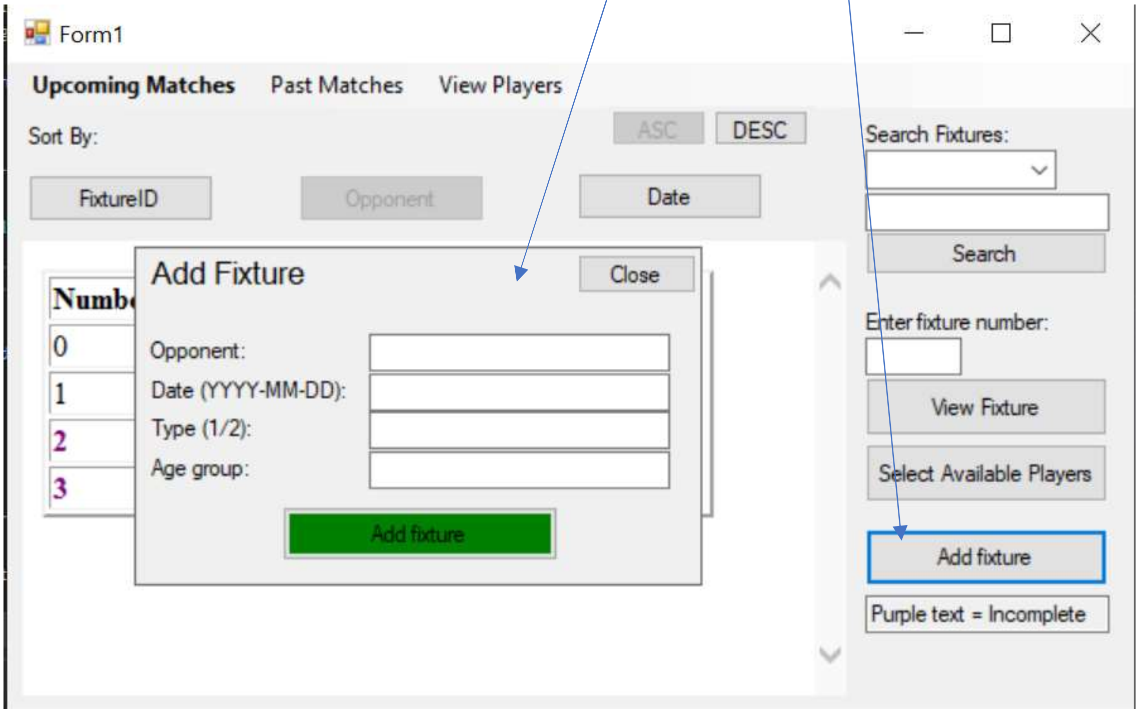


Test team 2
03/09/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
▶	111119	Player Eight	3	40	1
	111116	Player Five	2	55	0
	111115	Player Four	3	48	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	51	0
	111118	Player Seven	3	60	1
	111117	Player Six	3	45	1
	111121	Player Ten	3	50	1
	111114	Player Three	1	52	1
	111113	Player Two	1	49	1

Enables the user to enter a new fixture into the system

Add Fixture (Upcoming Fixtures):



Opponent:
Date (YYYY-MM-DD):
Type (1/2):
Age group:

Table HTML

The tables displaying fixtures seen on the ‘Upcoming matches’ and ‘Past matches’ pages are displayed by getting data from the database and then using HTML to display it as a table. I have utilised colours to indicate whether a fixture is missing player data (red) or has not been assigned players (purple). This is the function that return the HTML for the Tables:

```
    Function GenHTML() As String
        Dim HtmlSTR As String = "<table style=width:100%' border=2 align='centre'>"
        HtmlSTR &= "<tr> <th>Number</th><th>ID</th><th>Opponent</th><th>Date</th><th>"
        Dim i As Integer = 0

        For Each fixy In fixtures
            fixy.ThisTeam.FillTeam(fixy.FixtureID)
            fixy.ThisTeam.AddPlrStatsToPlrs(fixy.FixtureID)
            If Not fixy.State = 0 Then
                If fixy.ThisTeam.CheckForEmptyStats = True Then
                    HtmlSTR &= "<tr><td><b><p style='color:Tomato;'>" & i &
"</td><td><b><p style='color:Tomato;'>" & fixy.FixtureID & "</td><td><b><p
style='color:Tomato;'>" & fixy.Opponent & "</td><td><b><p style='color:Tomato;'>" &
fixy.FixtureDate & "</p></b></td></tr>"
                Else
                    HtmlSTR &= "<tr><td>" & i & "</td><td>" & fixy.FixtureID &
"</td><td>" & fixy.Opponent & "</td><td>" & fixy.FixtureDate & "</td></tr>"
                End If
            ElseIf fixy.State = 0 And fixy.ThisTeam.CountPs <
fixy.NumberOfPlayersRequired Then
                HtmlSTR &= "<tr><td><b><p style='color:Purple;'>" & i &
"</td><td><b><p style='color:Purple;'>" & fixy.FixtureID & "</td><td><b><p
style='color:Purple;'>" & fixy.Opponent & "</td><td><b><p style='color:Purple;'>" &
fixy.FixtureDate & "</p></b></td></tr>"
            Else
                HtmlSTR &= "<tr><td>" & i & "</td><td>" & fixy.FixtureID & "</td><td>"
& fixy.Opponent & "</td><td>" & fixy.FixtureDate & "</td></tr>"
            End If

            i = i + 1
        Next
        i = 0
        Return HtmlSTR
    End Function
```

Input Validation

This section contains the validation for all the inputs on the user interface

Form1 (Upcoming Matches)

TBox_SearchBox:

Invalid inputs handles by a try-catch statement in class ‘Report’

TBox_SelectFixture:

Only an integer between zero and the number of displayed fixtures should be allowed

```
If TBox_SelectFixture.Text = "" Or IsNumeric(TBox_SelectFixture.Text) = False Then

  MsgBox("Please enter a valid number")

Else

  If CInt(TBox_SelectFixture.Text) > fixReport.fixtures.Count - 1 Or CInt(TBox_SelectFixture.Text) < 0 Then

    MsgBox("Please enter a valid number")

  Else

           [CODE TO BE EXECUTED]

  End If

End If
```

TBox_Opponent, TBox_Type, TBox_Date, TBox_Age

Validation for entering a fixture

```
If Not TBox_Age.Text = Nothing And Not TBox_Date.Text = Nothing And Not TBox_Opponent.Text = Nothing And Not TBox_Type.Text = Nothing And IsDate(TBox_Date.Text) = True And IsNumeric(TBox_Type.Text) = True Then

  If Not CInt(TBox_Type.Text) > 2 And Not CInt(TBox_Type.Text) < 0 Then

    [CODE TO BE EXECUTED]

  Else

    MsgBox("Error, textboxes not filled in correctly")

  End If

Else

  MsgBox("Error, textboxes not filled in correctly")

End If
```

Form2 (Past Matches)

TBox_SearchBox2

Handled by try-catch in class ‘Report’

TBox_SelectFixture2

Only an integer between zero and the number of displayed fixtures should be allowed

```
    If TBox_SelectFixture2.Text = "" Or IsNumeric(TBox_SelectFixture2.Text) = False Then
      MsgBox("Please enter a valid number")
    Else
      If CInt(TBox_SelectFixture2.Text) > fixReport.fixtures.Count - 1 Or
CInt(TBox_SelectFixture2.Text) < 0 Then
        MsgBox("Please enter a valid number")
      Else
                [CODE TO BE EXECUTED]
      End If
    End If
```

Form 3 (View Fixture)

TBox_RunsScored, TBox_WicketsTaken, TBox_RunsCon

Input text boxes for player performance

```
    If IsNumeric(Tbox_RunsCon.Text) = False Or IsNumeric(Tbox_RunsScored.Text) = False Or
IsNumeric(Tbox_WicketsTaken.Text) = False Then
      MsgBox("Please enter valid values")
    Else
      If CInt(Tbox_RunsCon.Text) < -1 Or CInt(Tbox_RunsScored.Text) < -1 Or
CInt(Tbox_WicketsTaken.Text) < -1 Then
        MsgBox("Please enter valid values")
      Else
                [CODE TO BE EXECUTED]
      End If
    End If
```

Form 4 (View Players)

TBox_PlayerNameF, TBox_PlayerNameS, TBox_Pos, TBox_Team

When entering a new player into the database

```
 If Not TBox_PlrNameF.Text = Nothing And Not TBox_PlrNameS.Text = Nothing And Not
TBox_Pos.Text = Nothing And Not TBox_Team.Text = Nothing And TBox_Pos.Text And
IsNumeric(TBox_Pos.Text) = True Then
      If Not CInt(TBox_Pos.Text) < 0 And Not CInt(TBox_Pos.Text) > 3 Then
               [CODE TO BE EXECUTED]
        Else
          MsgBox("Error, Player ID wrong length")
        End If
      Else
        MsgBox("Error, textboxes not filled in correctly")
      End If
    Else
      MsgBox("Error, textboxes not filled in correctly")
    End If
```

Form 5 (Select available players)

N/A

Changes to system from original design/requirements

There have been a few changes from the original design of the program.

Testing Strategy

Below are the testing strategies for my program. I will be testing the inputs using TEX (Typical, Erroneous and extreme) data to ensure that the program responds correctly to any inputs it is given.

I will also include dry runs of some of the more complex procedures and compare the results with the system's outputs to ensure that they function as expected.

Inputs and controls:

Test Number	Test Description	Data type Typical Erroneous Extreme	Expected Result Typical Erroneous Extreme
Navigation			
1	Navigate to 'Past Matches' Page via the tool bar	Left Click Right Click	Past matches page opens Nothing happens
2	Navigate to 'View Teams' Page via the tool bar	Left Click Right Click	View teams page opens Nothing happens
3	Navigate to 'Upcoming matches' page via the toolbar	Left Click Right Click	Upcoming matches page opens Nothing happens
4	In 'Upcoming matches' Click all three sort buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens
5	In 'Upcoming matches' Click the ASC and DESC buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens
6	In 'Past matches' Click all three sort buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens
7	In 'Past matches' click the ASC and DESC buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens
8	In 'Upcoming matches' click 'View fixtures' with a known valid fixture number	Left Click Right Click	The view fixtures page opens Nothing happens

9	In ‘Past matches’ click ‘view fixtures’ with a known valid fixture number	Left Click Right Click	The view fixtures page opens Nothing happens
10	In ‘Upcoming matches’ click ‘Select available players’ with a known valid fixture number	Left Click Right Click	The Select available players page opens Nothing happens
11	In ‘Upcoming matches’ click ‘Add fixture’	Left Click Right Click	The add fixture panel opens Nothing happens
Form1 (Upcoming matches)			
12	Search for fixtures by FixtureID	03092001 1aB3^&*@E6$ 99999999	The fixture is correctly displayed No fixtures will be displayed No fixtures will be displayed
13	Search for fixtures by Opponent	test5 1aB3^&*@E6$ Reallyreallyreallyreally Reallyreallyreallyreally Reallyreallyreallyreally reallyreallyLongString	The fixture is correctly displayed No fixtures will be displayed No fixtures will be displayed
14	Search for fixtures by date	2021-01-02 1aB3^&*@E6$ 9999-12-31	The fixture is correctly displayed No fixtures will be displayed No fixtures will be displayed
15	Add fixture	TestTest 2021-05-06 1 U10 1aB3^&*@E6$ 1aB3^&*@E6$ 1aB3^&*@E6$ 1aB3^&*@E6$ Reallyreallyreallyreally Reallyreallyreallyreally Reallyreallyreallyreally reallyreallyLongString 9999-12-31 2 U99	The fixture is successfully added to the database An error message should display The fixture is successfully added to the database
16	Enter fixture number and click ‘view fixture’	0 1aB3^&*@E6$ [Highest index fixture number]	View fixture page opens Error message is displayed View fixture page opens
Form 2 (Past matches)			

17	Search for fixtures by FixtureID	03092001 1aB3^&*@E6$ 99999999	The fixture is correctly displayed No fixtures will be displayed No fixtures will be displayed
18	Search for fixtures by Opponent	test5 1aB3^&*@E6$ ReallyreallyreallyReallyreallyreallyReallyreallyreallyreallyreallyLongString	The fixture is correctly displayed No fixtures will be displayed No fixtures will be displayed
19	Search for fixtures by date	2021-01-02 1aB3^&*@E6$ 9999-12-31	The fixture is correctly displayed No fixtures will be displayed No fixtures will be displayed
20	Enter fixture number and click ‘view fixture’	0 1aB3^&*@E6$ [Highest index fixture number]	View fixture page opens Error message is displayed View fixture page opens
Form3 (View Fixture)			
21	Select a player from the DataGridView	Left click row header Left Click Cell other than row header	Player data is correctly displayed Nothing happens
22	For a future fixture, select player and click button ‘Swap player’ and then select a player to swap with	Select a player who isn’t in the fixture already Select a player who is already in the fixthre	Players are successfully swapped and the new team is displayed correctly Error message is displayed
23	For a future fixture, select ‘Generate team’	Perform on an empty fixture with sufficient number of available players Perform on a fixture with no players available Perform on a fixture with as sufficient number of players available but with an already assigned team	New team is generated with only available players and the correct number of players Error message Warning message with the option to override existing team
24	For a future fixture, select ‘Clear existing selections’	Perform on a full team Perform on a partially full team Perform on an empty team	All players are cleared from fixture All players are cleared from fixture Nothing happens

25	For a past fixture, select 'Generate match ratings'	Perform on a team with complete stats and no existing ratings Perform on a team with incomplete stats Perform on a team with existing ratings but changed player stats	Match ratings are successfully generated and displayed Error message Match ratings are successfully generated and displayed
26	For a past fixture, select 'unlock player stats', enter new values, click 'save new values' and then click 'lock player stats.	10, 5, 10 1aB3^&*@E6$, 1aB3^&*@E6$, 1aB3^&*@E6$ 10, 5, 10 but do not save new values	Player stats successfully change to 10, 5, 10 Error message Player stats successfully change to 99, 99, 99
Form 4 (view players)			
27	Select 'Add player', enter values and click 'add player'	Test, Tester, U10, 1 1aB3^&*@E6$, 1aB3^&*@E6$, 1aB3^&*@E6$, 1aB3^&*@E6$ Test, Tester, U10, 1 but don't confirm player	Player is successfully added to database Error message Player is not added to database
Form 5 (Select available players)			
29	On 'Upcoming matches' enter a valid fixture number and select 'Select available players. Change one player to available (green), one to unavailable (red) and the rest leave unknown (grey)	Left click the row headers Left click cells other than the row header	Player's availability will cycle with the mouse click Nothing happens
30	On 'Select available players' change one player to available and then leave and then return to the page	N/A	Availability should have saved and should be the same way it was left
Connection			
31	Open program	Database server running, correct address, no password Database server off	Program opens, fixtures are loaded Program doesn't open, error message

Processes

Process	**Test data**

CalculateMatchRating.CalculateGameRating

Fixture: TestTeam1
TotalRuns = 120
TotalWickets = 8
TotalRunsConceded = 80
Players:

PlayerID	RunsScored	Wickets	RunsCon
111115	13	1	8

```
    Sub CalculateGameRatings()
        Dim ScoreMultiplier As Integer = 50 * CurrentTeam.Count
        For Each plr In CurrentTeam
            plr.GameRating = ((plr.runsThisGame / TotalRuns) + (plr.wicketsThisGame /
(TotalWickets * 2)) + (0.5 * (1 - (plr.RunsConcededThisGame / TotalRunsConceded)))) *
ScoreMultiplier * 0.25
        Next
    End Sub
```

Player.CalculateRating

Player One
GamesPlayed:

FixtureID	GameRating	Date
05022001	60.42	2020-02-05
24022001	63.83	2020-02-24
25022001	64.70	2020-02-26

```
            While DR.Read
                GameDate = DR("Date")
                GameRating = CSng(DR("GameRating"))
                TotalScore = TotalScore + (GameRating /
Math.Sqrt(DateDiff(DateInterval.Day, GameDate.Date, Date.Today.Date)))
                Devisor = Devisor + (1 / Math.Sqrt(DateDiff(DateInterval.Day,
GameDate.Date, Date.Today.Date)))
            End While
```

GeneratedTeam.SelectPlayers

Team: Test2
Type: League
All players available

PlayerID	GamesPlayed	GamesAv	Rating
111112	10	20	65.42
111113	12	15	58.64
111114	8	9	56.39
111115	2	3	69.40
111116	5	12	69.44
111117	8	12	62.29
111118	14	15	64.90
111119	8	15	75.97
111120	10	10	50
111121	6	9	59.47
200021	0	0	50

```
        Select Case FixType
            Case 1 'friendly
                For Each plr In EligiblePlayers
                    If plr.GamesPlayed = 0 Or plr.GamesAvailable = 0 Then
                        plr.SelectionScore = 100
                    Else
                        plr.SelectionScore = (100 / (plr.GamesPlayed /
plr.GamesAvailable))
                    End If
                Next
            Case 2 'League
```

```
                For Each plr In EligiblePlayers
                    If plr.GamesPlayed = 0 Or plr.GamesAvailable = 0 Then
                        plr.SelectionScore = (plr.Rating ^ 2) / (100)
                    Else
                        plr.SelectionScore = (plr.Rating ^ 2) / (100 * plr.GamesPlayed
/ plr.GamesAvailable)
                    End If
                Next
        End Select
```

Technical Solution

GeneralInfo

```
Public Class GeneralInfo
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Property TotalGames

    Sub New()
        CalculateTotalGames()
    End Sub

    Sub CalculateTotalGames()
        Dim Cmd As MySqlCommand
        Dim SQLString As String = "SELECT COUNT(FixtureID) FROM fixtures"

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)

            Me.TotalGames = Cmd.ExecuteScalar

            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & " On General Info")
        End Try
    End Sub
End Class
```

Players

```
Public Class Player
    Dim Connection As New Connec
    Dim conn As New MySqlConnection(Connection.ConnStr)

    Property PlayerID As String
    Property FirstName As String
    Property Surname As String
    Property Team As String
    Property Position As Integer '1 - batter, 2 - bowler, 3 - both
    Property Rating As Single
    Property GamesPlayed As Integer
    Property GamesAvailable As Integer
    Property runsThisGame As Integer
    Property wicketsThisGame As Integer
    Property RunsConcededThisGame As Integer
    Property GameRating As Single
    Property SelectionScore As Single

    Sub New(ByVal PlayerID As String, ByVal FirstName As String, ByVal Surname As String, ByVal Position As Integer, ByVal Rating As
Single, ByVal Team As String, ByVal GamesPlayed As Integer, ByVal GameAvailable As Integer, ByVal RunsThisGame As Integer, ByVal
WicketsThisGame As Integer, ByVal RunsConceded As Integer)
        Me.PlayerID = PlayerID
        Me.FirstName = FirstName
        Me.Surname = Surname
        Me.Position = Position
        Me.Rating = Rating
        Me.Team = Team
        Me.GamesPlayed = GamesPlayed
        Me.GamesAvailable = GameAvailable
        Me.runsThisGame = RunsThisGame
        Me.wicketsThisGame = WicketsThisGame
        Me.RunsConcededThisGame = RunsConceded
    End Sub

    Sub AddPlayerPerformanceDB(FixtureID As String) 'given a fixture, adds player stats to instance of player for that game
        Dim Cmd As MySqlCommand
        Dim DR As MySqlDataReader
        Dim SQLString As String = "SELECT RunsScored, RunsConceded, WicketsTaken, GameRating FROM gamesplayed2 WHERE PlayerID = " &
Me.PlayerID & " AND FixtureID = " & FixtureID

        Try
            conn.Open()
            Cmd = New MySqlCommand(SQLString, conn)
            DR = Cmd.ExecuteReader()

            While DR.Read
                Me.runsThisGame = DR("RunsScored")
                Me.RunsConcededThisGame = DR("RunsConceded")
                Me.wicketsThisGame = DR("WicketsTaken")
                Me.GameRating = DR("GameRating")
            End While

            conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub

    Sub CalculateRating()
        Dim TotalScore As Single = 0
        Dim Devisor As Single = 0
        Dim GameDate As Date
        Dim GameRating As Single
        Dim Cmd As MySqlCommand
        Dim DR As MySqlDataReader
        Dim DateStr As String = Date.Today.Year & "-" & Date.Today.Month & "-" & Date.Today.Day
        Dim SQLString As String = "SELECT gamesplayed2.GameRating, fixtures.Date FROM gamesplayed2, fixtures WHERE 
gamesplayed2.FixtureID = fixtures.FixtureID AND gamesplayed2.PlayerID = " & PlayerID & " AND fixtures.Date < '" & DateStr & "'"

        Try
            conn.Open()
            Cmd = New MySqlCommand(SQLString, conn)
            DR = Cmd.ExecuteReader()

            While DR.Read
                GameDate = DR("Date")
                GameRating = CSng(DR("GameRating"))
                TotalScore = TotalScore + (GameRating / Math.Sqrt(DateDiff(DateInterval.Day, GameDate.Date, Date.Today.Date)))
                Devisor = Devisor + (1 / Math.Sqrt(DateDiff(DateInterval.Day, GameDate.Date, Date.Today.Date)))
            End While
            conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & " On calculate rating")
        End Try
        If Devisor = 0 Then
            TotalScore = 50
        Else
            TotalScore = TotalScore / Devisor
        End If

        Rating = TotalScore
    End Sub

End Class
```

Team

```
Public Class team 'list of player that have been picked and then play a fixture
    Dim Connection As Connec
    Dim Conn As MySqlConnection
    Property squad As New List(Of Player)
    Property currentTeam As New List(Of Player)

    Sub New()
        Connection = New Connec
        Conn = New MySqlConnection(Connection.ConnStr)
        GetAllPlayers()
        'For Each plr In currentTeam
        '    plr.CalculateRating()
        'Next
    End Sub

    Sub FillTeam(ByVal FixtureID As String) 'also removes any players that are already in it (but they will be re added if they are
in the DB)
        Me.currentTeam.Clear()
        Dim Cmd As MySqlCommand
        Dim DR As MySqlDataReader
        Dim SQLString As String = "SELECT * FROM players INNER JOIN gamesplayed2 ON players.PlayerID = gamesplayed2.PlayerID WHERE 
gamesplayed2.FixtureID = " & FixtureID & ""

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)
            DR = Cmd.ExecuteReader()

            While DR.Read
                currentTeam.Add(New Player(DR("PlayerID"), DR("FirstName"), DR("Surname"), DR("Position"), DR("Rating"), DR("Team"), 
DR("GamesPlayed"), DR("GamesAvailable"), 0, 0, 0))
            End While

            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & " On Fill Squad")
        End Try
    End Sub

    Function ReturnTeam()
        Return currentTeam
    End Function

    Sub GetAllPlayers()
        squad.Clear()
        Dim Cmd As MySqlCommand
        Dim DR As MySqlDataReader
        Dim SQLString As String = "SELECT * FROM players ORDER BY Surname ASC"

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)
            DR = Cmd.ExecuteReader()

            While DR.Read
                squad.Add(New Player(DR("PlayerID"), DR("FirstName"), DR("Surname"), DR("Position"), DR("Rating"), DR("Team"), 
DR("GamesPlayed"), DR("GamesAvailable"), 0, 0, 0))
            End While
            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & "On Get All Players")
        End Try
    End Sub

    Public Function CountPs()
        Return currentTeam.Count
    End Function

    Function CheckForEmptyStats() 'checks if any player data from the game hasn't been entered. Returns true if empty data found
        Dim Found As Boolean = False
        If currentTeam.Count = 0 Then
            Found = True
        End If
        For Each plr In currentTeam
            If plr.runsThisGame = -1 Or plr.wicketsThisGame = -1 Or plr.RunsConcededThisGame = -1 Then
                Found = True
            End If
        Next
        Return Found
    End Function

    Sub AddPlrStatsToPlrs(FixtureID As String) 'adds player stats to each player in a game
        For Each plr In currentTeam
            plr.AddPlayerPerformanceDB(FixtureID)
        Next
    End Sub

    Public Function ReturnPlayer(PlayerID As String)
        Dim SearchedPlayer As Player = Nothing
        For Each plr In squad
            If plr.PlayerID = PlayerID Then
                SearchedPlayer = plr
            End If
        Next
        Return SearchedPlayer
    End Function

    Public Function ReturnSquad()
```

```
        Return squad
    End Function

    Sub RefreshPlrRatings()
        For Each plr In squad
            plr.CalculateRating()
        Next

        Dim Cmd As MySqlCommand
        Dim SQLString As String = ""

        Try
            Conn.Open()
            For Each plr In squad
                SQLString = "UPDATE players SET Rating = " & plr.Rating & " WHERE PlayerID = " & plr.PlayerID
                Cmd = New MySqlCommand(SQLString, Conn)
                Cmd.ExecuteNonQuery()
            Next
            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & "On Get Fixtures")
        End Try
    End Sub
End Class
```

Fixtures

```
Public Class Fixture
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)

    Property ThisTeam As New team
    Property AgeGroup As Integer 'this is a string in the db and gets converted to an integer
    Property FixtureID As String
    Property Opponent As String
    Property FixtureDate As Date
    Property State As Integer '0 - not played, 1 - win, 2 - loss, 3 draw
    Property TotalRunsScored As Integer
    Property TotalRunsConceded As Integer
    Property NumberOfPlayersRequired As Integer
    Property TotalWicketsTaken As Integer
    Property Type As Integer '0 - Default, 1 - Friendly, 2 - League

    Sub New(ByVal fixID As String, ByVal opp As String, ByVal fixDate As Date, ByVal Type As Integer, ByVal State As Integer, ByVal
TotalRuns As Integer, ByVal TotalWickets As Integer, ByVal NumberOfPlayers As Integer, ByVal AgeGroupString As String, ByVal
TotalRunsConceded As Integer)
        Me.FixtureID = fixID
        Me.Opponent = opp
        Me.FixtureDate = fixDate
        Me.Type = Type
        Me.State = State
        Me.TotalRunsScored = TotalRuns
        Me.TotalWicketsTaken = TotalWickets
        Me.NumberOfPlayersRequired = NumberOfPlayers
        Me.TotalRunsConceded = TotalRunsConceded
        Dim NewAgeString As String
        NewAgeString = Mid(AgeGroupString, 2)
        Me.AgeGroup = CInt(NewAgeString)
        ThisTeam.FillTeam(Me.FixtureID)
    End Sub
End Class
```

Report

```
Public Class Report
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Property SortByString As String
    Property SortOrderString As String
    Property fixtures As New List(Of Fixture)

    Sub New(ByVal SortByString As String, ByVal SortOrderString As String, ByVal WhereClause As String) '1 past, 2 future
        Dim SQLString As String
        Dim CurrentDate As String
        Me.SortOrderString = SortOrderString
        Me.SortByString = SortByString

        CurrentDate = Date.Today.Year & "-" & Date.Today.Month & "-" & Date.Today.Day

        SQLString = "SELECT * FROM fixtures "
        SQLString &= WhereClause & " "
        SQLString &= "ORDER BY " & Me.SortByString & " " & Me.SortOrderString & ""

        fixtures.Clear()
        GetFixtures(SQLString)
    End Sub

    Function GenHTML() As String
        Dim HtmlSTR As String = "<table style=width:100%' border=2 align='centre'>"
        HtmlSTR &= "<tr> <th>Number</th><th>ID</th><th>Opponent</th><th>Date</th><th>"
        Dim i As Integer = 0

        For Each fixy In fixtures
            fixy.ThisTeam.FillTeam(fixy.FixtureID)
            fixy.ThisTeam.AddPlrStatsToPlrs(fixy.FixtureID)
            If Not fixy.State = 0 Then
                If fixy.ThisTeam.CheckForEmptyStats = True Then
                    HtmlSTR &= "<tr><td><b><p style='color:Tomato;'>" & i & "</td><td><b><p style='color:Tomato;'>" & fixy.FixtureID
& "</td><td><b><p style='color:Tomato;'>" & fixy.Opponent & "</td><td><b><p style='color:Tomato;'>" & fixy.FixtureDate &
"</p></b></td></tr>"
                Else
                    HtmlSTR &= "<tr><td>" & i & "</td><td>" & fixy.FixtureID & "</td><td>" & fixy.Opponent & "</td><td>" &
fixy.FixtureDate & "</td></tr>"
                End If
            ElseIf fixy.State = 0 And fixy.ThisTeam.CountPs < fixy.NumberOfPlayersRequired Then
                HtmlSTR &= "<tr><td><b><p style='color:Purple;'>" & i & "</td><td><b><p style='color:Purple;'>" & fixy.FixtureID &
"</td><td><b><p style='color:Purple;'>" & fixy.Opponent & "</td><td><b><p style='color:Purple;'>" & fixy.FixtureDate &
"</p></b></td></tr>"
            Else
                HtmlSTR &= "<tr><td>" & i & "</td><td>" & fixy.FixtureID & "</td><td>" & fixy.Opponent & "</td><td>" &
fixy.FixtureDate & "</td></tr>"
            End If

            i = i + 1
        Next
        i = 0
        Return HtmlSTR
    End Function

    Sub GetFixtures(ByVal SQLString As String)
        Dim Cmd As MySqlCommand
        Dim DR As MySqlDataReader

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)
            DR = Cmd.ExecuteReader()

            While DR.Read
                fixtures.Add(New Fixture(DR("FixtureID"), DR("Opponent"), DR("Date"), DR("Type"), DR("State"), DR("TotalRuns"),
DR("TotalWickets"), DR("NumberOfPlayers"), DR("AgeGroup"), DR("TotalRunsConceded")))
            End While

            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message & "On Get Fixtures")
        End Try
    End Sub
End Class
```

GeneratedTeam

```
Class GeneratedTeam 'fill team must be used ater this otherwise the team wont be added to the fixture in the program
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Property ThisTeam As New team
    Property FixtureID As String
    Property EligiblePlayers As New List(Of Player)
    Property SortList As New List(Of Player)
    Property ExistingTeam As New List(Of Player)
    Sub New(FixtureID As String, AgeGroup As Integer, Type As Integer, NoPlayers As Integer)
        SortList = Nothing
        Me.FixtureID = FixtureID
        ThisTeam.FillTeam(Me.FixtureID)
        CheckExistingTeam(AgeGroup, Type, NoPlayers)
    End Sub

    Sub CreateNewTeam(AgeGroup As Integer, Type As Integer, NoPlayers As Integer) 'groups the procedures that are needed to gen a new
team
        GetEligiblePlayers()
        SelectPlayers(Type)
        RemoveExcessPlayers(NoPlayers)
    End Sub

    Sub CheckExistingTeam(AgeGroup As Integer, Type As Integer, NoPlayers As Integer)
        If ThisTeam.CountPs = NoPlayers Then
            Dim Answer As Integer
            Answer = MsgBox("A Team has already been assigned. Would you Like To delete the existing selections And generate a New
team?", vbQuestion + vbYesNo + vbDefaultButton2)
            If Answer = vbYes Then
                DeleteExistingTeamDB()
                CreateNewTeam(AgeGroup, Type, NoPlayers)
                WriteLineupToDB()
            Else
                SortList = ThisTeam.ReturnTeam
            End If
        ElseIf ThisTeam.CountPs = 0 Then
            MsgBox("No existing team found. New team will be generated")
            DeleteExistingTeamDB()
            CreateNewTeam(AgeGroup, Type, NoPlayers)
            WriteLineupToDB()
        ElseIf ThisTeam.CountPs > 0 And ThisTeam.CountPs < NoPlayers Then
            Dim Answer As Integer
            Answer = MsgBox("Existing selections were found but too few players are assigned For the fixture. Would you Like a New
team To be generated? Select 'No' if you want to keep the current selections.", vbQuestion + vbYesNo + vbDefaultButton2)
            If Answer = vbYes Then
                DeleteExistingTeamDB()
                CreateNewTeam(AgeGroup, Type, NoPlayers)
                WriteLineupToDB()
            Else
                SortList = ThisTeam.ReturnTeam
            End If
        End If
    End Sub

    Sub DeleteExistingTeamDB()
        Dim Cmd As MySqlCommand
        Dim SQLString As String = "DELETE FROM gamesplayed2 WHERE FixtureID = " & FixtureID & ""

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)
            Cmd.ExecuteNonQuery()
            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Sub GetEligiblePlayers() 'gets players that are available
        EligiblePlayers.Clear()
        Dim Cmd As MySqlCommand
        Dim DR As MySqlDataReader
        Dim SQLString As String = "SELECT * FROM players INNER JOIN availability2 ON players.PlayerID = availability2.PlayerID WHERE 
availability2.Availability = 2 AND FixtureID = " & FixtureID & " ORDER BY players.Surname ASC"

        Try
            Conn.Open()
            Cmd = New MySqlCommand(SQLString, Conn)
            DR = Cmd.ExecuteReader()

            While DR.Read()
                EligiblePlayers.Add(New Player(DR("PlayerID"), DR("FirstName"), DR("Surname"), DR("Position"), DR("Rating"), 
DR("Team"), DR("GamesPlayed"), DR("GamesAvailable"), 0, 0, 0))
            End While

            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Sub SelectPlayers(FixType As Integer) 'calculates a team and writes it to SortList
```

```
        Select Case FixType
            Case 1 'friendly
                For Each plr In EligiblePlayers
                    If plr.GamesPlayed = 0 Or plr.GamesAvailable = 0 Then
                        plr.SelectionScore = 100
                    Else
                        plr.SelectionScore = (100 / (plr.GamesPlayed / plr.GamesAvailable))
                    End If
                Next
            Case 2 'League
                For Each plr In EligiblePlayers
                    If plr.GamesPlayed = 0 Or plr.GamesAvailable = 0 Then
                        plr.SelectionScore = (plr.Rating ^ 2) / (100)
                    Else
                        plr.SelectionScore = (plr.Rating ^ 2) / (100 * plr.GamesPlayed / plr.GamesAvailable)
                    End If
                Next
        End Select

        Dim TestStr As String = ""
        For Each plr In EligiblePlayers
            TestStr &= plr.SelectionScore & vbCrLf
        Next
        MsgBox(TestStr)

        SortList = EligiblePlayers
        QuickSortScores(0, EligiblePlayers.Count - 1)
        SortList.Reverse()
    End Sub

    Sub QuickSortScores(ByVal min As Integer, max As Integer)
        Dim RandNum As New Random
        Dim mid As Integer
        Dim top As Integer
        Dim bot As Integer
        Dim i As Integer
        Dim MidP As Player

        If min >= max Then Exit Sub
        i = RandNum.Next(min, max + 1)
        MidP = SortList(i)
        mid = SortList(i).SelectionScore

        SortList(i) = SortList(min)

        bot = min
        top = max

        Do
            Do While SortList(top).SelectionScore >= mid
                top = top - 1
                If top <= bot Then Exit Do
            Loop
            If top <= bot Then
                SortList(bot) = MidP
                Exit Do
            End If

            SortList(bot) = SortList(top)
            bot = bot + 1

            Do While SortList(bot).SelectionScore < mid
                bot = bot + 1
                If bot >= top Then Exit Do
            Loop
            If bot >= top Then
                bot = top
                SortList(top) = MidP
                Exit Do
            End If
            SortList(top) = SortList(bot)
        Loop
        QuickSortScores(min, bot - 1)
        QuickSortScores(bot + 1, max)
    End Sub

    Function ReturnSortList()
        Return SortList
    End Function

    Sub RemoveExcessPlayers(NoPlayers As Integer) 'removes players with lowest selectionsscore from sortlist
        Dim TempSortList As New List(Of Player)
        If EligiblePlayers.Count < NoPlayers Then
            MsgBox("Error, not enough eligible players")
        Else
            For i = 0 To NoPlayers - 1
                TempSortList.Add(SortList(i))
            Next
        End If
        SortList.Clear()
        SortList = TempSortList
    End Sub


    Public Sub WriteLineupToDB() 'uploads new lineup to gamesplayed table
        Dim Cmd As MySqlCommand
        Dim SQLString As String = ""

        For Each plr In SortList
            SQLString = "INSERT INTO gamesplayed2 (PlayerID, FixtureID) VALUES (" & plr.PlayerID & ", " & FixtureID & ")"

            Try
                Conn.Open()
                Cmd = New MySqlCommand(SQLString, Conn)
                Cmd.ExecuteNonQuery()
                Conn.Close()
            Catch ex As Exception
                MsgBox(ex.Message)
            End Try
        Next
    End Sub

    Function ReturnEligiblePlayers()
        Return EligiblePlayers
    End Function

End Class
```

CalculateMatchRatings

```
Class CalculateMatchRatings
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Private CurrentTeam As List(Of Player)
    Private TotalRuns As Integer
    Private TotalWickets As Integer
    Private TotalRunsConceded As Integer
    Private FixtureID As String

    Sub New(CurrentTeam As List(Of Player), TotalRuns As Integer, TotalWickets As Integer, TotalRunsConceded As Integer, FixtureID As
String)
        Me.CurrentTeam = CurrentTeam
        Me.TotalRuns = TotalRuns
        Me.TotalWickets = TotalWickets
        Me.TotalRunsConceded = TotalRunsConceded
        Me.FixtureID = FixtureID

        CalculateGameRatings()
        UploadMRatingDB()
    End Sub

    Sub CalculateGameRatings()
        Dim ScoreMultiplier As Integer = 50 * CurrentTeam.Count
        For Each plr In CurrentTeam
            plr.GameRating = ((plr.runsThisGame / TotalRuns) + (plr.wicketsThisGame / (TotalWickets * 2)) + (0.5 * (1 -
(plr.RunsConcededThisGame / TotalRunsConceded)))) * ScoreMultiplier * 0.25
        Next
    End Sub

    Sub UploadMRatingDB()
        Dim Cmd As MySqlCommand
        Dim SQLString As String

        For Each plr In CurrentTeam
            If Not plr.GamesPlayed = 0 Then
```

```
                SQLString = "UPDATE gamesplayed2 SET GameRating = " & plr.GameRating & " WHERE PlayerID = " & plr.PlayerID & " AND 
FixtureID = " & FixtureID
                Try
                    Conn.Open()
                    Cmd = New MySqlCommand(SQLString, Conn)
                    Cmd.ExecuteNonQuery()
                    Conn.Close()
                Catch ex As Exception
                    MsgBox(ex.Message)
                End Try
            End If

        Next
    End Sub

    Function ReturnTeam()
        Return CurrentTeam
    End Function
End Class
```

Availability

```
Public Class Availability
    Property ThisPlayer As Player
    Property Available As Integer

    Sub New(ThisPlayer As Player, Available As Integer)
        Me.ThisPlayer = ThisPlayer
        Me.Available = Available
    End Sub
End Class
```

Form1

```
Imports MySql.Data.MySqlClient
```

```
Public Class Form1
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Dim GeneralInfo1 As GeneralInfo
    Public InstanceForm2 As Form2
    Public InstanceForm4 As Form4
    Public InstanceForm5 As Form5
    Dim fixReport As Report
    Dim FormLocation As Point
    Dim SortByString As String
    Dim SortOrderString As String
    Dim SquadTeam As New team

    Sub New()
        ' This call is required by the designer.
        InitializeComponent()

        FormLocation.X = 500
        FormLocation.Y = 200

        SquadTeam.GetAllPlayers()
        SquadTeam.RefreshPlrRatings()

        GeneralInfo1 = New GeneralInfo
        InstanceForm2 = New Form2()
        InstanceForm4 = New Form4()
        Panel_AddFix.Hide()
        Me.Show()
        Me.Location = FormLocation

        DefaultSetup()
        RefreshTable()
    End Sub


    Private Sub DefaultSetup()
        SortByString = "FixtureID"
        SortOrderString = "ASC"
        Btn_Sort1.Enabled = 0
        Btn_SortOrderASC.Enabled = 0
    End Sub

    Private Sub RefreshWebBrowser()
        WebBrowser1.DocumentText = Nothing
        WebBrowser1.DocumentText = fixReport.GenHTML()
    End Sub


    Private Sub PastMatchesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles PastMatchesToolStripMenuItem.Click
        InstanceForm2.Show()
        InstanceForm2.Location = Me.Location
        Me.Hide()
    End Sub

    Private Sub RefreshTable()
        Dim WhereClause As String = "WHERE State = 0"
        fixReport = Nothing
        fixReport = New Report(SortByString, SortOrderString, WhereClause)
        RefreshWebBrowser()
    End Sub

    Private Sub Btn_Sort1_Click(sender As Object, e As EventArgs) Handles Btn_Sort1.Click
        SortByString = Btn_Sort1.Text
        RefreshTable()
        Btn_Sort1.Enabled = False
        Btn_Sort2.Enabled = True
        Btn_Sort3.Enabled = True
    End Sub

    Private Sub Btn_Sort2_Click(sender As Object, e As EventArgs) Handles Btn_Sort2.Click
        SortByString = Btn_Sort2.Text
        RefreshTable()
        Btn_Sort1.Enabled = True
        Btn_Sort2.Enabled = False
        Btn_Sort3.Enabled = True
    End Sub

    Private Sub Btn_Sort3_Click(sender As Object, e As EventArgs) Handles Btn_Sort3.Click
        SortByString = Btn_Sort3.Text
        RefreshTable()
        Btn_Sort1.Enabled = True
        Btn_Sort2.Enabled = True
        Btn_Sort3.Enabled = False
    End Sub

    Private Sub Btn_SortOrderASC_Click(sender As Object, e As EventArgs) Handles Btn_SortOrderASC.Click
        SortOrderString = "ASC"
        Btn_SortOrderASC.Enabled = False
        Btn_SortOrderDESC.Enabled = True
        RefreshTable()
    End Sub

    Private Sub Btn_SortOrderDESC_Click(sender As Object, e As EventArgs) Handles Btn_SortOrderDESC.Click
        SortOrderString = "DESC"
        Btn_SortOrderDESC.Enabled = False
        Btn_SortOrderASC.Enabled = True
        RefreshTable()
    End Sub

    Private Sub Btn_Search_Click(sender As Object, e As EventArgs) Handles Btn_Search.Click
        Dim SelectedField As String = CBox_ChooseSearch.SelectedItem.ToString
        Dim SearchString As String = TBox_SearchBox1.Text
        Dim WhereClause As String = "WHERE " & SelectedField & " = '" & SearchString & "'"
        fixReport = Nothing
        fixReport = New Report(SortByString, SortOrderString, WhereClause)
        RefreshWebBrowser()
    End Sub

    Private Sub Btn_ViewFixture_Click(sender As Object, e As EventArgs) Handles Btn_ViewFixture.Click
        If TBox_SelectFixture.Text = "" Or IsNumeric(TBox_SelectFixture.Text) = False Then
            MsgBox("Please enter a valid number")
        Else
            If CInt(TBox_SelectFixture.Text) > fixReport.fixtures.Count - 1 Or CInt(TBox_SelectFixture.Text) < 0 Then
                MsgBox("Please enter a valid number")
            Else
                Dim SelectedFixIndex As Integer = CInt(TBox_SelectFixture.Text)
                Dim CurrentFixture As Fixture = fixReport.fixtures(SelectedFixIndex)

                Dim InstanceForm3 As New Form3(CurrentFixture)
                InstanceForm3.Location = Me.Location
            End If
        End If
    End Sub

    Private Sub Btn_AddFix_Click(sender As Object, e As EventArgs) Handles Btn_AddFix.Click
        Panel_AddFix.Show()
    End Sub

    Private Sub Btn_EnterFix_Click(sender As Object, e As EventArgs) Handles Btn_EnterFix.Click
        Dim Cmd As MySqlCommand
        Dim SQLString As String = ""

        If Not TBox_Age.Text = Nothing And Not TBox_Date.Text = Nothing And Not TBox_Opponent.Text = Nothing And Not TBox_Type.Text =
Nothing And IsDate(TBox_Date.Text) = True And IsNumeric(TBox_Type.Text) = True Then
            If Not CInt(TBox_Type.Text) > 2 And Not CInt(TBox_Type.Text) < 0 Then
                Dim NewFixID As String = ""
                Dim NewFixDate As Date = CDate(TBox_Date.Text)

                NewFixID &= NewFixDate.Day.ToString("00")
                NewFixID &= NewFixDate.Month.ToString("00")
                NewFixID &= Mid(NewFixDate.Year.ToString, 3)
                Dim Count As Integer = 0
                For Each fixy In fixReport.fixtures
                    If fixy.FixtureDate = NewFixDate Then
                        Count = Count + 1
                    End If
                Next
                Count = Count + 1
                NewFixID &= Count.ToString("00")

                MsgBox(NewFixID)
                If NewFixID.Length = 8 Then
                    Dim DateStr As String = NewFixDate.Year.ToString & "-" & NewFixDate.Month.ToString & "-" &
NewFixDate.Day.ToString
                    SQLString = "INSERT INTO fixtures (FixtureID, Opponent, Date, Type, agegroup) VALUES ("
                    SQLString &= "'" & NewFixID & "', '" & TBox_Opponent.Text & "', '" & DateStr & "', " & CInt(TBox_Type.Text) & ",
'" & TBox_Age.Text & "')"
                    Try
                        Conn.Open()
                        Cmd = New MySqlCommand(SQLString, Conn)
                        Cmd.ExecuteNonQuery()
                        Conn.Close()
                        MsgBox("Fixture added " & SQLString)
                        TBox_Age.Text = Nothing
                        TBox_Date.Text = Nothing
                        TBox_Opponent.Text = Nothing
                        TBox_Type.Text = Nothing
                    Catch ex As Exception
                        MsgBox(ex.Message)
                    End Try
                Else
                    MsgBox("Error, fixID wrong length")
                End If
            Else
                MsgBox("Error, textboxes not filled in correctly")
            End If
        Else
            MsgBox("Error, textboxes not filled in correctly")
        End If
    End Sub

    Private Sub Btn_ClosePanel_Click(sender As Object, e As EventArgs) Handles Btn_ClosePanel.Click
        Panel_AddFix.Hide()
    End Sub

    Private Sub ViewTeamsToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ViewTeamsToolStripMenuItem.Click
        InstanceForm4.Show()
        InstanceForm4.Location = Me.Location
        Me.Hide()
    End Sub

    Private Sub Btn_SelAvails_Click(sender As Object, e As EventArgs) Handles Btn_SelAvails.Click
        If TBox_SelectFixture.Text = "" Or IsNumeric(TBox_SelectFixture.Text) = False Then
            MsgBox("Please enter a valid number")
        Else
            If CInt(TBox_SelectFixture.Text) > fixReport.fixtures.Count - 1 Or CInt(TBox_SelectFixture.Text) < 0 Then
                MsgBox("Please enter a valid number")
            Else
                Dim Index As Integer = CInt(TBox_SelectFixture.Text)
                Dim CurrentFix As Fixture = fixReport.fixtures(Index)
                InstanceForm5 = New Form5(CurrentFix)
                InstanceForm5.Show()
            End If
        End If
    End Sub
End Class
```

Form2

```
Public Class Form2
    Dim GeneralInfo1 As GeneralInfo
    Dim FormLocation As Point
    Dim fixReport As Report
    Dim SortByString As String
    Dim SortOrderString As String
    Sub New()
        ' This call is required by the designer.
        InitializeComponent()

        Me.GeneralInfo1 = New GeneralInfo
        DefaultSetup()
        RefreshTable()
        Me.Hide()
    End Sub

    Sub DefaultSetup()
        SortByString = "FixtureID"
        SortOrderString = "ASC"
        Btn_SortOrderASC2.Enabled = False
        Btn_Sort12.Enabled = False
    End Sub

    Sub RefreshWebBrowser()
        WebBrowser1.DocumentText = Nothing
        WebBrowser1.DocumentText = fixReport.GenHTML
    End Sub
    Private Sub RefreshTable()
        Dim WhereClause As String = "WHERE NOT State = 0"
        fixReport = Nothing
        fixReport = New Report(SortByString, SortOrderString, WhereClause)
        RefreshWebBrowser()
    End Sub

    Private Sub UpcomingMatchesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles
UpcomingMatchesToolStripMenuItem.Click
        Form1.Show()
        Form1.Location = Me.Location
        Me.Hide()
    End Sub

    Private Sub Btn_Sort12_Click(sender As Object, e As EventArgs) Handles Btn_Sort12.Click
        SortByString = Btn_Sort12.Text
        RefreshTable()
        Btn_Sort12.Enabled = False
        Btn_Sort22.Enabled = True
        Btn_Sort32.Enabled = True
    End Sub

    Private Sub Btn_Sort22_Click(sender As Object, e As EventArgs) Handles Btn_Sort22.Click
        SortByString = Btn_Sort22.Text
        RefreshTable()
        Btn_Sort12.Enabled = True
        Btn_Sort22.Enabled = False
        Btn_Sort32.Enabled = True
    End Sub

    Private Sub Btn_Sort32_Click(sender As Object, e As EventArgs) Handles Btn_Sort32.Click
        SortByString = Btn_Sort32.Text
        RefreshTable()
        Btn_Sort12.Enabled = True
        Btn_Sort22.Enabled = True
        Btn_Sort32.Enabled = False
    End Sub

    Private Sub Btn_SortOrderASC2_Click(sender As Object, e As EventArgs) Handles Btn_SortOrderASC2.Click
        SortOrderString = "ASC"
        Btn_SortOrderASC2.Enabled = False
        Btn_SortOrderDESC2.Enabled = True
        RefreshTable()
    End Sub

    Private Sub Btn_SortOrderDESC2_Click(sender As Object, e As EventArgs) Handles Btn_SortOrderDESC2.Click
        SortOrderString = "DESC"
        Btn_SortOrderDESC2.Enabled = False
        Btn_SortOrderASC2.Enabled = True
        RefreshTable()
    End Sub

    Private Sub Btn_Search2_Click(sender As Object, e As EventArgs) Handles Btn_Search2.Click
        Dim SelectedField As String = CBox_ChooseSearch2.SelectedItem.ToString
        Dim SearchString As String = TBox_SearchBox2.Text
        Dim WhereClause As String = "WHERE " & SelectedField & " = '" & SearchString & "'"
        fixReport = Nothing
        fixReport = New Report(SortByString, SortOrderString, WhereClause)
        RefreshWebBrowser()
    End Sub

    Private Sub Btn_ViewFixture2_Click(sender As Object, e As EventArgs) Handles Btn_ViewFixture2.Click
        If TBox_SelectFixture2.Text = "" Or IsNumeric(TBox_SelectFixture2.Text) = False Then
            MsgBox("Please enter a valid number")
        Else
            If CInt(TBox_SelectFixture2.Text) > fixReport.fixtures.Count - 1 Or CInt(TBox_SelectFixture2.Text) < 0 Then
                MsgBox("Please enter a valid number")
            Else
                Dim SelectedFixIndex As Integer = CInt(TBox_SelectFixture2.Text)
                Dim LastWindowOpen As Integer = 1
                Dim CurrentFixture As Fixture = fixReport.fixtures(SelectedFixIndex)

                Dim InstanceForm3 As New Form3(CurrentFixture)
            End If
        End If
    End Sub

    Private Sub PastMatchesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles PastMatchesToolStripMenuItem.Click
        Form4.Show()
        Me.Hide()
        Form4.Location = Me.Location
    End Sub
    Private Sub ViewTeamsToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ViewTeamsToolStripMenuItem.Click
        Form4.Show()
        Form4.Location = Me.Location
        Me.Hide()
    End Sub
End Class
```

Form3

```
Imports MySql.Data.MySqlClient
Public Class Form3
    Dim Connection As New Connec
    Dim Conn As MySqlConnection
    Dim DataComplete As Boolean
    Dim PlrSwap(2) As String 'holds the id of the players to be swapped
    Property SelectedFix As Fixture
    Property CurrentTeam As List(Of Player)
    Property Squad As List(Of Player)
    Property Avails As New List(Of Availability)
    Sub New(ByVal CurrentFixture As Fixture)
        InitializeComponent()

        Conn = New MySqlConnection(Connection.ConnStr)
        Me.SelectedFix = CurrentFixture
        Me.SelectedFix.ThisTeam = CurrentFixture.ThisTeam
        Me.SelectedFix.ThisTeam.FillTeam(SelectedFix.FixtureID)
        Me.SelectedFix.ThisTeam.AddPlrStatsToPlrs(SelectedFix.FixtureID)
        Me.SelectedFix.ThisTeam.GetAllPlayers()
        CurrentTeam = Me.SelectedFix.ThisTeam.ReturnTeam
        Squad = Me.SelectedFix.ThisTeam.ReturnSquad
        Me.Show()
        DataGridView_SwapPlr.Hide()

        GetAvails()
        DisplaySetup()
        DisplayPlayers()
        RefreshCells()
        DataGridView_Players.SelectionMode = DataGridViewSelectionMode.FullRowSelect

    End Sub

    Sub DisplaySetup()
        Lbl_FixtureTitle.Text = SelectedFix.AgeGroup & " VS " & SelectedFix.Opponent
        Lbl_FixDate.Text = "Date: " & SelectedFix.FixtureDate
        Lbl_FixtureID.Text = "Fixture ID: " & SelectedFix.FixtureID
        Lbl_FixState.Text = "State: "
        Select Case SelectedFix.State
            Case 0
                Lbl_FixState.Text &= "Not Played"
            Case 1
                Lbl_FixState.Text &= "Won"
            Case 2
                Lbl_FixState.Text &= "Loss"
            Case 3
                Lbl_FixState.Text &= "Draw"
        End Select
        Lbl_FixType.Text = "Type: "
        Select Case SelectedFix.Type
            Case 1
                Lbl_FixType.Text &= "Friendly"
            Case 2
                Lbl_FixType.Text &= "League"
        End Select

        Tbox_RunsScored.ReadOnly = True
        Tbox_RunsCon.ReadOnly = True
        Tbox_WicketsTaken.ReadOnly = True

        If SelectedFix.FixtureDate < Date.Today Then
            Btn_ClearSels.Enabled = False
            Btn_ClearSels.Hide()
            Btn_GenerateTeam.Enabled = False
            Btn_GenerateTeam.Hide()
            Btn_SwapPlayers.Enabled = False
            Btn_SwapPlayers.Hide()
            DataComplete = True
            For Each plr In CurrentTeam 'checks if any player match data is missing
                If plr.runsThisGame = -1 Or plr.wicketsThisGame = -1 Or plr.RunsConcededThisGame = -1 Then
                    DataComplete = False
                End If
            Next
        Else
            Btn_SwapPlayers.Hide()
            DataGridView_SwapPlr.Hide()
            Btn_CalcMRatings.Enabled = False
            Btn_CalcMRatings.Hide()
            Btn_UnlockPP.Enabled = 0
            Btn_UnlockPP.Hide()
        End If
        Btn_UploadPStats.Enabled = False
        Btn_UploadPStats.Hide()
    End Sub

    Private Sub Btn_CloseForm3_Click(sender As Object, e As EventArgs) Handles Btn_CloseForm3.Click
        Dim GenMRatings As New CalculateMatchRatings(CurrentTeam, SelectedFix.TotalRunsScored, SelectedFix.TotalWicketsTaken,
SelectedFix.TotalRunsConceded, SelectedFix.FixtureID)
        CurrentTeam.Clear()
        CurrentTeam = GenMRatings.ReturnTeam

        Me.Dispose()
    End Sub


    Sub DisplayPlayers()
        DataGridView_Players.Rows.Clear()
        CurrentTeam = SelectedFix.ThisTeam.ReturnTeam

        If Not CurrentTeam.Count = 0 Then
            Dim PositionString As String = ""
            For Each plr In CurrentTeam
                Select Case plr.Position
                    Case 1
                        PositionString = "Batter"
                    Case 2
                        PositionString = "Bowler"
                    Case 3
```

```
                        PositionString = "Batter/Bowler"
                End Select
                If SelectedFix.FixtureDate < Date.Today Then
                    DataGridView_Players.Rows.Add(plr.FirstName & " " & plr.Surname, PositionString, plr.Rating, 
Math.Round(plr.GameRating, 3), 2)
                Else
                    DataGridView_Players.Rows.Add(plr.FirstName & " " & plr.Surname, PositionString, plr.Rating, 
Math.Round(plr.GamesPlayed / plr.GamesAvailable, 3), 2)
                End If
            Next
        End If
        If SelectedFix.FixtureDate < Date.Today Then
            DataGridView_Players.Columns(3).HeaderText = "Game Rating"
        End If
    End Sub

    Sub GetAvails() 'gets available players
        Dim Cmd As MySqlCommand
        Dim SQLString As String
        Dim AvInt As Integer
        Try
            Conn.Open()
            For Each plr In Squad
                SQLString = "SELECT Availability FROM availability2 WHERE FixtureID = " & SelectedFix.FixtureID & " AND PlayerID =" & 
plr.PlayerID
                Cmd = New MySqlCommand(SQLString, Conn)
                AvInt = Cmd.ExecuteScalar()
                If AvInt = Nothing Then
                    Avails.Add(New Availability(plr, 0))
                Else
                    Avails.Add(New Availability(plr, AvInt))
                End If
            Next
            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

    Sub RefreshCells()
```

```
      For Each row As DataGridViewRow In DataGridView_Players.Rows
          If row.Cells("Availability2").Value = 1 Then
              row.DefaultCellStyle.BackColor = Color.Red
          ElseIf row.Cells("Availability2").Value = 2 Then
              row.DefaultCellStyle.BackColor = Color.Green
          ElseIf row.Cells("Availability2").Value = 0 Then
              row.DefaultCellStyle.BackColor = Color.Gray
          End If
      Next

      For Each row As DataGridViewRow In DataGridView_SwapPlr.Rows
          If row.Cells("Availability").Value = 1 Then
              row.DefaultCellStyle.BackColor = Color.Red
          ElseIf row.Cells("Availability").Value = 2 Then
              row.DefaultCellStyle.BackColor = Color.Green
          ElseIf row.Cells("Availability").Value = 0 Then
              row.DefaultCellStyle.BackColor = Color.Gray
          End If
      Next
  End Sub
Private Sub DataGridView_Players_RowHeaderMouseClick() Handles DataGridView_Players.RowHeaderMouseClick
      Btn_SwapPlayers.Show()
      PlrSwap(1) = CurrentTeam(DataGridView_Players.SelectedRows(0).Index).PlayerID
      Dim PlrIndex As Integer = DataGridView_Players.SelectedRows.Item(0).Index
      Lbl_PlrName.Text = DataGridView_Players.SelectedRows.Item(0).Cells(0).Value.ToString
      Lbl_PlrRating.Text = CurrentTeam(PlrIndex).Rating
      Lbl_PlrTeam.Text = CurrentTeam(PlrIndex).Team
      Lbl_GamesAvailable.Text = CurrentTeam(PlrIndex).GamesAvailable
      Lbl_GamesPlayed.Text = CurrentTeam(PlrIndex).GamesPlayed
      Lbl_AGScore.Text = 100 * Math.Round((CurrentTeam(PlrIndex).GamesPlayed / CurrentTeam(PlrIndex).GamesAvailable), 3) & "%"

      Tbox_RunsScored.Text = CurrentTeam(PlrIndex).runsThisGame
      Tbox_RunsCon.Text = CurrentTeam(PlrIndex).RunsConcededThisGame
      Tbox_WicketsTaken.Text = CurrentTeam(PlrIndex).wicketsThisGame

      Btn_UploadPStats.Enabled = False
      Btn_UploadPStats.Hide()
      Tbox_RunsScored.ReadOnly = True
      Tbox_RunsCon.ReadOnly = True
      Tbox_WicketsTaken.ReadOnly = True
```

```
        Btn_UnlockPP.Text = "Unlock player performance stats"
        Btn_UnlockPP.BackColor = Color.Orange
    End Sub

    Private Sub Btn_GenerateTeam_Click(sender As Object, e As EventArgs) Handles Btn_GenerateTeam.Click
        DataGridView_Players.Rows.Clear()
        Dim NewTeam As New GeneratedTeam(SelectedFix.FixtureID, SelectedFix.AgeGroup, SelectedFix.Type, 
SelectedFix.NumberOfPlayersRequired)
        SelectedFix.ThisTeam.FillTeam(SelectedFix.FixtureID)
        DisplayPlayers()
        MsgBox("Complete")
    End Sub

    Private Sub Btn_ClearSels_Click(sender As Object, e As EventArgs) Handles Btn_ClearSels.Click 'deletes all players assigned to a 
fixture (unless it has been played)
        Dim Answer As Integer
        Answer = MsgBox("Are you sure you want to clear the current player selections for this fixture?", vbQuestion + vbYesNo + 
vbDefaultButton2)

        If Answer = vbYes Then
            If SelectedFix.State <> 0 Then
                MsgBox("Error, fixture has already been played")
            ElseIf SelectedFix.State = 0 Then
                Dim Cmd As MySqlCommand
                Dim SQLString As String = "DELETE FROM gamesplayed2 WHERE FixtureID = " & SelectedFix.FixtureID & ""
                Try
                    Conn.Open()
                    Cmd = New MySqlCommand(SQLString, Conn)
                    Cmd.ExecuteNonQuery()
                    Conn.Close()
                Catch ex As Exception
                    MsgBox(ex.Message)
                End Try
            End If
            DisplayPlayers()
            MsgBox("Players successfully removed")
        End If
        SelectedFix.ThisTeam.FillTeam(SelectedFix.FixtureID)
        CurrentTeam = SelectedFix.ThisTeam.ReturnTeam
        DisplayPlayers()
    End Sub


    Private Sub Btn_UnlockPP_Click(sender As Object, e As EventArgs) Handles Btn_UnlockPP.Click
        Dim PlrIndex As Integer = DataGridView_Players.SelectedRows.Item(0).Index
        If SelectedFix.FixtureDate < Date.Today Then
            If Tbox_RunsScored.ReadOnly = True Then
                Tbox_RunsScored.ReadOnly = False
                Tbox_RunsCon.ReadOnly = False
                Tbox_WicketsTaken.ReadOnly = False
                Btn_UnlockPP.Text = "Lock player performance stats"
                Btn_UnlockPP.BackColor = Color.Red
                Btn_UploadPStats.Enabled = True
                Btn_UploadPStats.Show()
            ElseIf Tbox_RunsScored.ReadOnly = False Then
                Tbox_RunsScored.ReadOnly = True
                Tbox_RunsCon.ReadOnly = True
                Tbox_WicketsTaken.ReadOnly = True
                Btn_UnlockPP.Text = "Unlock player performance stats"
                Btn_UnlockPP.BackColor = Color.Orange
                Btn_UploadPStats.Enabled = False
                Btn_UploadPStats.Hide()
                Tbox_RunsScored.Text = CurrentTeam(PlrIndex).runsThisGame
                Tbox_RunsCon.Text = CurrentTeam(PlrIndex).RunsConcededThisGame
                Tbox_WicketsTaken.Text = CurrentTeam(PlrIndex).wicketsThisGame
            End If
        Else
            MsgBox("Error, cannot alter player performance stats for a fixture that hasn't been played")
        End If

    End Sub

    Private Sub Btn_UploadPStats_Click(sender As Object, e As EventArgs) Handles Btn_UploadPStats.Click
        Dim PlrIndex As Integer = DataGridView_Players.SelectedRows.Item(0).Index
        Dim Cmd As MySqlCommand

        If IsNumeric(Tbox_RunsCon.Text) = False Or IsNumeric(Tbox_RunsScored.Text) = False Or IsNumeric(Tbox_WicketsTaken.Text) = 
False Then
            MsgBox("Please enter valid values")
        Else
```

```
            If CInt(Tbox_RunsCon.Text) < -1 Or CInt(Tbox_RunsScored.Text) < -1 Or CInt(Tbox_WicketsTaken.Text) < -1 Then
                MsgBox("Please enter valid values")
            Else
                Dim SQLString As String = "UPDATE gamesplayed2 SET RunsScored = " & CInt(Tbox_RunsScored.Text) & ", RunsConceded = "
& CInt(Tbox_RunsCon.Text) & ", WicketsTaken = " & CInt(Tbox_WicketsTaken.Text)
                SQLString &= " WHERE PlayerID = '" & CurrentTeam(PlrIndex).PlayerID & "' AND FixtureID = '" & SelectedFix.FixtureID &
"'"

                Try
                    Conn.Open()
                    Cmd = New MySqlCommand(SQLString, Conn)
                    Cmd.ExecuteNonQuery()
                    Conn.Close()
                    MsgBox("New values uploaded")
                Catch ex As Exception
                    MsgBox(ex.Message)
                End Try
            End If
        End If
    End Sub

    Private Sub Btn_CalcMRatings_Click(sender As Object, e As EventArgs) Handles Btn_CalcMRatings.Click
        Dim PlrString As String = ""

        If DataComplete = False Then
            Dim MissingPList As List(Of Player) = ReturnMissingPData()
            For Each plr In MissingPList
                PlrString &= plr.PlayerID & " " & plr.FirstName & " " & plr.Surname & vbCrLf
            Next
            MsgBox("Player data missing for: " & vbCrLf & PlrString & "Match ratings cannot be calculated without complete data")
        Else
            Dim CalcRatings As New CalculateMatchRatings(CurrentTeam, SelectedFix.TotalRunsScored, SelectedFix.TotalWicketsTaken,
SelectedFix.TotalRunsConceded, SelectedFix.FixtureID)
            CurrentTeam = Nothing
            CurrentTeam = CalcRatings.ReturnTeam
            For Each plr In CurrentTeam
                PlrString &= plr.FirstName & " " & plr.Surname & " " & plr.GameRating & vbCrLf
            Next
            MsgBox("Match Ratings:" & vbCrLf & PlrString)
        End If
        DisplayPlayers()
    End Sub

    Private Function ReturnMissingPData() 'returns a list of players whose data is missing
        Dim MissingPData As New List(Of Player)
        For Each plr In CurrentTeam
            If plr.runsThisGame = -1 Or plr.wicketsThisGame = -1 Or plr.RunsConcededThisGame = -1 Then
                MissingPData.Add(plr)
            End If
        Next
        Return MissingPData
    End Function

    Private Sub Btn_SwapPlayer_Click(sender As Object, e As EventArgs) Handles Btn_SwapPlayers.Click
        DataGridView_SwapPlr.Show()
        For Each av In Avails
            DataGridView_SwapPlr.Rows.Add(av.ThisPlayer.FirstName & " " & av.ThisPlayer.Surname, av.ThisPlayer.Position,
av.ThisPlayer.Rating, Math.Round(av.ThisPlayer.GamesPlayed / av.ThisPlayer.GamesAvailable, 3), av.Available)
        Next
        MsgBox("Selet player to swap '" & DataGridView_Players.SelectedRows(0).Cells(0).Value.ToString & "' with")
        RefreshCells()
    End Sub

    Private Sub DataGridView_SwapPlr_RowHeaderMouseClick() Handles DataGridView_SwapPlr.RowHeaderMouseClick
        PlrSwap(2) = Squad(DataGridView_SwapPlr.SelectedRows(0).Index).PlayerID

        Dim Found As Boolean = False
        For Each plr In CurrentTeam
            If PlrSwap(2) = plr.PlayerID Then
                Found = True
            End If
        Next

        If Found = False Then
            Dim answer As Integer
            answer = MsgBox("Are you sure you want to remove " & DataGridView_Players.SelectedRows(0).Cells(0).Value.ToString & "
with " & DataGridView_SwapPlr.SelectedRows(0).Cells(0).Value.ToString & " ?", vbQuestion + vbYesNo + vbDefaultButton2)

            If answer = vbYes Then
                Dim Cmd1 As MySqlCommand
                Dim Cmd2 As MySqlCommand
                Dim SQLstring1 As String = "DELETE FROM gamesplayed2 WHERE FixtureID = " & SelectedFix.FixtureID & " AND PlayerID = " 
& PlrSwap(1)
                Dim SQLstring2 As String = "INSERT INTO gamesplayed2 (PlayerID, FixtureID) VALUES (" & PlrSwap(2) & ", " & 
SelectedFix.FixtureID & ")"
                Try
                    Conn.Open()
                    Cmd1 = New MySqlCommand(SQLstring1, Conn)
                    Cmd2 = New MySqlCommand(SQLstring2, Conn)
                    Cmd1.ExecuteNonQuery()
                    Cmd2.ExecuteNonQuery()
                    Conn.Close()
                    MsgBox("Complete")
                Catch ex As Exception
                    MsgBox(ex.Message)
                End Try
                Me.Hide()
            End If
        Else
            MsgBox("Player is already in fixture")
        End If

    End Sub

End Class
```

Form4

```
Imports MySql.Data.MySqlClient
Public Class Form4
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Dim Squad As New team
    Dim CurrentSquad As List(Of Player)
    Sub New()
        InitializeComponent()
        Squad.GetAllPlayers()
        CurrentSquad = Squad.ReturnSquad

        For Each plr In CurrentSquad
            DataGridView_AllPlayers.Rows.Add(plr.PlayerID, plr.FirstName & " " & plr.Surname, plr.Position, Math.Round(plr.Rating,
1), Math.Round(plr.GamesPlayed / plr.GamesAvailable, 3))
        Next
        Panel_AddPlr.Hide()
    End Sub

    Private Sub UpcomingMatchesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles
UpcomingMatchesToolStripMenuItem.Click
        Form1.Show()
        Me.Hide()
        Form1.Location = Me.Location
    End Sub

    Private Sub PastMatchesToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles PastMatchesToolStripMenuItem.Click
        Form2.Show()
        Me.Hide()
        Form2.Location = Me.Location
    End Sub

    Private Sub Btn_AddPlayer_Click(sender As Object, e As EventArgs) Handles Btn_AddPlayer.Click
        Panel_AddPlr.Show()
    End Sub

    Private Sub Btn_EnterPlr_Click(sender As Object, e As EventArgs) Handles Btn_EnterPlr.Click
        Dim Cmd As MySqlCommand
        Dim SQLString As String = ""

        If Not TBox_PlrNameF.Text = Nothing And Not TBox_PlrNameS.Text = Nothing And Not TBox_Pos.Text = Nothing And Not
TBox_Team.Text = Nothing And IsNumeric(TBox_Pos.Text.ToString) = True Then
            If Not CInt(TBox_Pos.Text) < 0 And Not CInt(TBox_Pos.Text) > 3 Then
                Dim NewPlrID As String = ""
                'these next ifs generate a player id for the new fixture
                NewPlrID &= Mid(Date.Today.Year, 3)
                Dim count As Integer = 0
                For Each plr In CurrentSquad
                    count = count + 1
                Next
                count = count + 1
                Dim IndexStr As String = count.ToString("0000")
                NewPlrID &= IndexStr

                MsgBox(NewPlrID)
                If NewPlrID.Length = 6 Then
                    SQLString = "INSERT INTO players (PlayerID, FirstName, Surname, Team, Position) VALUES "
                    SQLString &= "('" & NewPlrID & "', '" & TBox_PlrNameF.Text & "', '" & TBox_PlrNameS.Text & "', '" &
TBox_Team.Text & "', " & CInt(TBox_Pos.Text) & ")"
                    Try
                        Conn.Open()
                        Cmd = New MySqlCommand(SQLString, Conn)
                        Cmd.ExecuteNonQuery()
                        Conn.Close()
                        MsgBox("Fixture added " & SQLString)
                        TBox_PlrNameF.Text = Nothing
                        TBox_PlrNameS.Text = Nothing
                        TBox_Pos.Text = Nothing
                        TBox_Team.Text = Nothing
                    Catch ex As Exception
                        MsgBox(ex.Message)
                    End Try
                Else
                    MsgBox("Error, Player ID wrong length")
                End If
            Else
                MsgBox("Error, textboxes not filled in correctly")
            End If
        Else
            MsgBox("Error, textboxes not filled in correctly")
        End If
    End Sub

    Private Sub Btn_ClosePanel_Click(sender As Object, e As EventArgs) Handles Btn_ClosePanel.Click
        Panel_AddPlr.Hide()
        TBox_PlrNameF.Text = Nothing
        TBox_PlrNameS.Text = Nothing
        TBox_Pos.Text = Nothing
        TBox_Team.Text = Nothing
    End Sub

End Class
```

Form5

```
Imports MySql.Data.MySqlClient


Public Class Form5
    Dim Connection As New Connec
    Dim Conn As New MySqlConnection(Connection.ConnStr)
    Dim CurrentSquad As List(Of Player)
    Property SelectedFix As Fixture

    Sub New(ThisFixture As Fixture)
        InitializeComponent()
        Me.SelectedFix = ThisFixture
        SelectedFix.ThisTeam.GetAllPlayers()
        CurrentSquad = SelectedFix.ThisTeam.ReturnSquad
        DisplayAllPlrs()
        Lbl_FixName.Text = SelectedFix.Opponent
        Lbl_Date.Text = SelectedFix.FixtureDate
    End Sub
    Private Sub Btn_CloseForm3_Click(sender As Object, e As EventArgs) Handles Btn_CloseForm3.Click
        SaveAvailability()
        Form1.Show()
        Form1.Location = Me.Location
        Me.Dispose()
    End Sub
    Sub DisplayAllPlrs()
        Dim Avs As New List(Of Availability)
        Dim Cmd As MySqlCommand
        Dim SQLString As String
        Dim AvInt As Integer
        Try
            Conn.Open()
            For Each plr In CurrentSquad
                SQLString = "SELECT Availability FROM availability2 WHERE FixtureID = " & SelectedFix.FixtureID & " AND PlayerID =" & 
plr.PlayerID
                Cmd = New MySqlCommand(SQLString, Conn)
                AvInt = Cmd.ExecuteScalar()
                If AvInt = Nothing Then
                    Avs.Add(New Availability(plr, 0))
                Else
                    Avs.Add(New Availability(plr, AvInt))
                End If
            Next
            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

        For Each av In Avs
            DataGridView_AllPlayers.Rows.Add(av.ThisPlayer.PlayerID, av.ThisPlayer.FirstName & " " & av.ThisPlayer.Surname, 
av.ThisPlayer.Position, av.ThisPlayer.Rating, Math.Round(av.ThisPlayer.GamesPlayed / av.ThisPlayer.GamesAvailable), av.Available)
        Next
        RefreshCells()
    End Sub

    Sub RefreshCells()
        For Each row As DataGridViewRow In DataGridView_AllPlayers.Rows
            If row.Cells("Availability").Value = 1 Then
                row.DefaultCellStyle.BackColor = Color.Red
            ElseIf row.Cells("Availability").Value = 2 Then
                row.DefaultCellStyle.BackColor = Color.Green
            ElseIf row.Cells("Availability").Value = 0 Then
                row.DefaultCellStyle.BackColor = Color.Gray
            End If
        Next
    End Sub

    Private Sub DataGridView_AllPlayers_RowHeaderMouseClick() Handles DataGridView_AllPlayers.RowHeaderMouseClick
        Select Case DataGridView_AllPlayers.SelectedRows(0).Cells("Availability").Value
            Case 0
                DataGridView_AllPlayers.SelectedRows(0).Cells("Availability").Value = 1
            Case 1
                DataGridView_AllPlayers.SelectedRows(0).Cells("Availability").Value = 2
            Case 2
                DataGridView_AllPlayers.SelectedRows(0).Cells("Availability").Value = 0
        End Select
        RefreshCells()
    End Sub

    Sub SaveAvailability()
        Dim Avails As New List(Of Availability)
        Avails.Clear()
        For Each row As DataGridViewRow In DataGridView_AllPlayers.Rows
            If Not row.Index >= DataGridView_AllPlayers.Rows.Count - 1 Then
                Avails.Add(New Availability(CurrentSquad(row.Index), row.Cells("Availability").Value))
            End If
        Next

        UploadAvail(Avails)
    End Sub

    Sub UploadAvail(Avails As List(Of Availability)) 'clears existing, uploads new
        Dim Cmd As MySqlCommand
        Dim SQLString As String = ""

        Try
            Conn.Open()
            SQLString = "DELETE FROM availability2 WHERE FixtureID = " & SelectedFix.FixtureID
            Cmd = New MySqlCommand(SQLString, Conn)
            Cmd.ExecuteNonQuery()
            For Each av In Avails
                SQLString = "INSERT INTO availability2 VALUES (" & av.ThisPlayer.PlayerID & ", " & SelectedFix.FixtureID & ", " &
av.Available & ")"
                Cmd = New MySqlCommand(SQLString, Conn)
                Cmd.ExecuteNonQuery()
            Next
            Conn.Close()
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

End Class
```

Testing

Below are the test results for my program. I will be testing the inputs using TEX (Typical, Erroneous and extreme) data to ensure that the program responds correctly to any inputs it is given

I will also include dry runs of some of the more complex procedures and compare the results with the system's outputs to ensure that they function as expected

Inputs and Controls

The brackets contain the reference to the screenshot for that test at the end of this document under 'Testing Screenshots

Test Number	Test Description	Data type Typical Erroneous Extreme	Expected Result Typical Erroneous Extreme	Actual Result	Pass/Fail
Navigation					
1	Navigate to 'Past Matches' Page via the tool bar	Left Click Right Click	Past matches page opens Nothing happens	Past matches page opens(1.1) Nothing happens(1.2)	Pass Pass
2	Navigate to 'View Players' Page via the tool bar	Left Click Right Click	View players page opens Nothing happens	View players page opens(2.1) Nothing happens(2.2)	Pass Pass
3	Navigate to 'Upcoming matches' page via the toolbar	Left Click Right Click	Upcoming matches page opens Nothing happens	Upcoming matches page opens(3.1 Nothing happens(3.2)	Pass Pass
4	In 'Upcoming matches' Click all three sort buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens	A new table is displayed with the fixtures sorted correctly(4.1) Nothing happens(4.2)	Pass Pass
5	In 'Upcoming matches' Click the ASC and DESC buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly	A new table is displayed with the fixtures sorted correctly(5.1)	Pass Pass

			Nothing happens	Nothing happens(5.2)	
6	In 'Past matches' Click all three sort buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens	A new table is displayed with the fixtures sorted correctly(6.1) Nothing happens(6.2)	Pass Pass
7	In 'Past matches' click the ASC and DESC buttons	Left Click Right Click	A new table is displayed with the fixtures sorted correctly Nothing happens	A new table is displayed with the fixtures sorted correctly(7.1) Nothing happens(7.2)	Pass Pass
8	In 'Upcoming matches' click 'View fixtures' with a known valid fixture number	Left Click, 2 Right Click, 2	The view fixtures page open Nothing happens	The view fixtures page open(8.1) Nothing happens(8.2)	Pass Pass
9	In 'Past matches' click 'view fixtures' with a known valid fixture number	Left Click, 1 Right Click, 1	The view fixtures page opens Nothing happens	The view fixtures page opens(9.1) Nothing happens(9.2)	Pass Pass
10	In 'Upcoming matches' click 'Select available players' with a known valid fixture number	Left Click Right Click	The Select available players page opens Nothing happens	The Select available players page opens(10.1) Nothing happens(10.2)	Pass Pass
11	In 'Upcoming matches' click 'Add fixture'	Left Click Right Click	The add fixture panel opens Nothing happens	The add fixture panel opens(11.1) Nothing happens(11.2)	Pass Pass
Form1 (Upcoming matches)					
12	Search for fixtures by FixtureID	03092001 1aB3^&*@E6$ 99999999	The fixture is correctly displayed No fixtures will be displayed	The fixture is correctly displayed(12.1) No fixtures are displayed(12.2) No fixtures will is displayed(12.3)	Pass Pass Pass

			No fixture will be displayed		
13	Search for fixtures by Opponent	test5 1aB3^&*@E6$ ReallyreallyreallyreallyReallyreallyreallyreallyReallyreallyreallyreallyreallyreallyLongString	The fixture is correctly displayed No fixtures will be displayed No fixture will be displayed	The fixture is correctly displayed(13.1) No fixtures are displayed(13.2) No fixtures are displayed(13.3)	Pass Pass Pass
14	Search for fixtures by date	2021-01-02 1aB3^&*@E6$ 9999-12-31	The fixture is correctly displayed No fixtures will be displayed No fixture will be displayed	The fixture is correctly displayed(14.1) No fixtures are displayed(14.2) No fixtures are displayed(14.3)	Pass Pass Pass
15	Add fixture	TestTest 2021-05-06 1 U10 1aB3^&*@E6$ 1aB3^&*@E6$ 1aB3^&*@E6$ 1aB3^&*@E6$ ReallyreallyreallyreallyReallyreallyreallyreallyReallyreallyreallyreallyreallyreallyLongString 9999-12-31 2 U99	The fixture is successfully added to the database An error message should display The fixture is successfully added to the database	The fixture is successfully added to the database(15.1) An error message displays (15.2) The fixture is successfully added to the database(15.3)	Pass Pass Pass
16	Enter fixture number and click 'view fixture'	2 1aB3^&*@E6$ 5	View fixture page opens Error message is displayed View fixture page opens	View fixture page opens(16.1) Error message is displayed(16.2) View fixture page opens(16.3)	Pass Pass Pass
Form 2 (Past matches)					
17	Search for fixtures by FixtureID	03092001 1aB3^&*@E6$ 99999999	The fixture is correctly displayed	The fixture is correctly displayed(17.1) No fixtures are displayed(17.2)	Pass Pass Pass

			No fixtures will be displayed No fixture will be displayed	No fixtures are displayed(17.3)	
18	Search for fixtures by Opponent	test5 1aB3^&*@E6$ ReallyreallyreallyreallyReallyreallyreallyreallyReallyreallyreallyreallyreallyreallyLongString	The fixture is correctly displayed No fixtures will be displayed No fixture will be displayed	The fixture is correctly displayed(18.1) No fixtures are displayed(18.2) The fixture is correctly displayed because it was generated in a previous test(18.3)	Pass Pass Pass
19	Search for fixtures by date	2021-01-02 1aB3^&*@E6$ 9999-12-31	The fixture is correctly displayed No fixtures will be displayed An error message will display and no fixture will be displayed	The fixture is correctly displayed(19.1) No fixtures are displayed(19.2) The fixture is correctly displayed because it was generated in a previous test(19.3)	Pass Pass Pass
20	Enter fixture number and click 'view fixture'	0 1aB3^&*@E6$ 2	View fixture page opens Error message is displayed View fixture page opens	View fixture page opens(20.1) Error message is displayed(20.2) View fixture page opens(20.3)	Pass Pass Pass
			Form3 (View Fixture)		
21	Select a player from the DataGridView	Left click row header Left Click Cell other than row header	Player data is correctly displayed Nothing happens	Player data is correctly displayed(21.1) Nothing happens(21.2)	Pass Pass
22	For a future fixture, select player and click button 'Swap player' and then select a	Select a player who isn't in the fixture already (player 9) Select a player who is already in the fixture (player 5)	Players are successfully swapped and the new team is	Players are successfully swapped and the new team is displayed correctly(22.1)	Pass Pass

	player to swap with		displayed correctly Error message is displayed	Error message is displayed(22.2)	
23	For a future fixture, select 'Generate team'	Perform on an empty fixture with sufficient number of available players (test5) Perform on a fixture with no players available(TestTest) Perform on a fixture with as sufficient number of players available but with an already assigned team	New team is generated with only available players and the correct number of players Error message Warning message with the option to override existing team	New team is generated with only available players and the correct number of players(23.1) Error message(23.2) Warning message with the option to override existing team(23.3)	Pass Pass Pass
24	For a future fixture, select 'Clear existing selections'	Perform on a full team Perform on an empty team	All players are cleared from fixture Nothing happens	All players are cleared from fixture(24.1) Nothing happens(24.2)	Pass Pass
25	For a past fixture, select 'Generate match ratings'	Perform on a team with complete stats and no existing ratings Perform on a team with incomplete stats Perform on a team with existing ratings but changed player stats	Match ratings are successfully generated and displayed Error message Match ratings are successfully generated and displayed	Match ratings are successfully generated and displayed(25.1) Error message(25.2) Match ratings are successfully generated and displayed(25.3)	Pass Pass Pass
26	For a past fixture, select 'unlock player stats', enter new values, click 'save new values' and then click 'lock player stats.	10, 5, 10 1aB3^&*@E6$, 1aB3^&*@E6$, 1aB3^&*@E6$ 99, 99, 99	Player stats successfully change to 10, 5, 10 Error message Player stats successfully change to 99, 99, 99	Player stats successfully change to 10, 5, 10(26.1) Error message(26.2) Player stats successfully change to 99, 99, 99(26.3)	Pass Pass Pass

Form 4 (view players)					
27	Select 'Add player', enter values and click 'add player'	Test, Tester, U10, 1 1aB3^&*@E6$, 1aB3^&*@E6$, 1aB3^&*@E6$, 1aB3^&*@E6$ Test, Tester, U10, 1 but don't confirm player	Player is successfully added to database Error message Player is not added to database	Player is successfully added to database(27.1) Error message(27.2) Player is not added to database(27.3)	Pass Pass Pass
Form 5 (Select available players)					
28	On 'Upcoming matches' enter a valid fixture number and select 'Select available players. Change one player to available (green), one to unavailable (red) and the rest leave unknown (grey)	Left click the row headers Left click cells other than the row header	Player's availability will cycle with the mouse click Nothing happens	Player's availability cycles with the mouse click(28.1) Nothing happens(28.2)	Pass Pass
29	On 'Select available players' change one player to available and then leave and then return to the page	Use a middle entry Use the first entry Use the last entry	Availability should have saved and should be the same way it was left Availability should have saved and should be the same way it was left Availability should have saved and should be the same way it was left	Availability is correctly saved(29.1) Availability is correctly saved(29.2) Availability is correctly saved(29.3)	Pass Pass Pass
Connection					
30	Open program	Database server running, correct address Database server off	Program opens, fixtures are loaded	Program opens, fixtures are loaded(30.1)	Pass Pass

			Program doesn’t open, error message	Program doesn’t open, error message(30.2)	

Processes

Here I will test the processes as planned in the testing strategy. I will compare the calculated results with the programs output.

Below is a repeat of the table in the design, the results are one the next page.

Process	**Test data**

CalculateMatchRating.CalculateGameRating	Fixture: TestTeam1
	TotalRuns = 120
	TotalWickets = 8
	TotalRunsConceded = 80
	Players:

PlayerID	RunsScored	Wickets	RunsCon
111115	13	1	8

```
    Sub CalculateGameRatings()
        Dim ScoreMultiplier As Integer = 50 * CurrentTeam.Count
        For Each plr In CurrentTeam
            plr.GameRating = ((plr.runsThisGame / TotalRuns) + (plr.wicketsThisGame / 
(TotalWickets * 2)) + (0.5 * (1 - (plr.RunsConcededThisGame / TotalRunsConceded)))) * 
ScoreMultiplier * 0.25
        Next
    End Sub
```

Player.CalculateRating	Player One GamesPlayed:

FixtureID	GameRating	Date
05022001	60.42	2020-02-05
24022001	63.83	2020-02-24
25022001	64.70	2020-02-26

```
            While DR.Read
                GameDate = DR("Date")
                GameRating = CSng(DR("GameRating"))
                TotalScore = TotalScore + (GameRating / 
Math.Sqrt(DateDiff(DateInterval.Day, GameDate.Date, Date.Today.Date)))
                Devisor = Devisor + (1 / Math.Sqrt(DateDiff(DateInterval.Day, 
GameDate.Date, Date.Today.Date)))
            End While
```

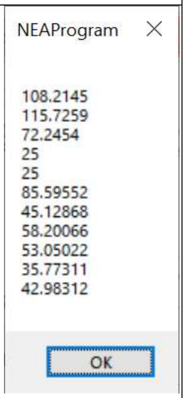
GeneratedTeam.SelectPlayers	Team: Test2 Type: League All players available

PlayerID	GamesPlayed	GamesAv	Rating
111112	10	20	65.42
111113	12	15	58.64
111114	8	9	56.39
111115	2	3	69.40
111116	5	12	69.44
111117	8	12	62.29
111118	14	15	64.90
111119	8	15	75.97
111120	10	10	50
111121	6	9	59.47
200021	0	0	50

```
        Select Case FixType
            Case 1 'friendly
                For Each plr In EligiblePlayers
                    If plr.GamesPlayed = 0 Or plr.GamesAvailable = 0 Then
                        plr.SelectionScore = 100
                    Else
                        plr.SelectionScore = (100 / (plr.GamesPlayed /
plr.GamesAvailable))
                    End If
                Next
            Case 2 'League
                For Each plr In EligiblePlayers
                    If plr.GamesPlayed = 0 Or plr.GamesAvailable = 0 Then
                        plr.SelectionScore = (plr.Rating ^ 2) / (100)
                    Else
                        plr.SelectionScore = (plr.Rating ^ 2) / (100 * plr.GamesPlayed
/ plr.GamesAvailable)
                    End If
                Next
        End Select
```

Results:

Numbers in brackets are references to the testing screenshots at the end of the documnet

<table>
<tr><th>Process</th><th>Calculation</th><th>Expected Output</th><th>Program output</th></tr>
<tr><td>CalculateMatchRating. CalculateGameRating</td><td>((13 / 120) + (1 / (2 * 8)) + (0.5 * (1 - (8 / 80)))) * (50 * 8) * 0.25</td><td>62.083</td><td>62.083 (31) Pass</td></tr>
<tr><td>Player. Calculaterating</td><td>
<table>
<tr><td>0 + (60.42 / SQRT(51))</td><td>8.460492268</td></tr>
<tr><td>8.46 + (63.83 / SQRT(32))</td><td>19.74414873</td></tr>
<tr><td>20.85 + (64.70 / SQRT(30))</td><td>31.55669855</td></tr>
<tr><td></td><td></td></tr>
<tr><td>0 + (1 / SQRT(51))</td><td>0.140028008</td></tr>
<tr><td>0.14 + (1 / SQRT(32))</td><td>0.316804704</td></tr>
<tr><td>0.32 + (1 / SQRT(30))</td><td>0.49937889</td></tr>
<tr><td></td><td></td></tr>
<tr><td>32.7 / 0.500</td><td>63.1918954</td></tr>
</table>
</td><td>63.19</td><td>63.2 (33) Pass</td></tr>
<tr><td>GeneratedTeam. SelectPlayers</td><td colspan="2">
<table>
<tr><td></td><td></td><td></td></tr>
<tr><td>111112</td><td>(65.42 ^ 2) / ((100 * 10) / 20))</td><td>85.595528</td></tr>
<tr><td>111113</td><td>(58.64 ^ 2) / ((100 * 12) / 15)</td><td>42.98312</td></tr>
<tr><td>111114</td><td>(56.39 ^ 2) / ((100 * 8) / 9)</td><td>35.77311113</td></tr>
<tr><td>111115</td><td>(69.40 ^ 2) / ((100 * 2) / 3)</td><td>72.2454</td></tr>
<tr><td>111116</td><td>(69.44 ^ 2) / ((100 * 5) / 12)</td><td>115.7259264</td></tr>
<tr><td>111117</td><td>(62.29 ^ 2) / ((100 * 8) / 12)</td><td>58.2006615</td></tr>
<tr><td>111118</td><td>(64.90 ^ 2) / ((100 * 14) / 15)</td><td>45.12867857</td></tr>
<tr><td>111119</td><td>(75.97 ^ 2) / ((100 * 8) / 15)</td><td>108.2145169</td></tr>
<tr><td>111120</td><td>(50 ^ 2) / ((100 * 10) / 10)</td><td>25</td></tr>
<tr><td>111121</td><td>(59.47 ^ 2) / ((100 * 6) / 9)</td><td>53.0502135</td></tr>
<tr><td>200021</td><td>(50 ^ 2) / (100)</td><td>25</td></tr>
</table>
</td><td>

108.2145
115.7259
72.2454
25
25
85.59552
45.12868
58.20066
53.05022
35.77311
42.98312
(32)Pass. Values match up. Small errors accounted for by rounding.</td></tr>
</table>

Evaluation

Comparison of system and original requirements

1. To be able to store player and match data
   The database stores all necessary player and match data
   1.1. All data to be stored on a database
        All data is stored on a server based database
   1.2. Stores relevant personal player data, such as player name and team
        Player name, team and position are stored on the database
   1.3. Store player match statistics
        The database stores the runs scored, runs conceded and wickets taken for each fixture the player is in
        1.3.1. Stores past and future availability of players
               The 'Availability' table in the database stores which players are available for past and future fixtures
        1.3.2. Stores players' past match performance
               The 'GamesPlayed' table in the database stores the performance of players for every fixture
        1.3.3. Store number of games played and number of games available
               Both the number of games played and the number of games available are stored in the 'Players' table in the database
   1.4. Store Availability of players for future fixtures
        This information is stored in the 'Availability' table of the databse

2. Collect and store player availability
   I decided not to use an email based system for my project, however the user still has the ability to enter player availability within the program, and player availability is stored in the database.
   2.1. Automatically send an email to players requesting availability at the start of season
   2.2. The email can be easily responded to by clicking a link
   2.3. The email links to the database and automatically stores the availability data for each player for each game
   2.4. Acceptable responses to email: 'Available', 'Not available', 'Maybe' and 'No answer'

3. Select players for games
   The system has the ability to automatically generate player selections for games
   3.1. System can generate an ideal selection of players for each upcoming game
        The system selects the ideal players for each fixture based on multiple parameters
   3.2. The selection process aims to make fair picks – each player plays a number of games approximately proportional to the amount of games they have been available for
        The algorithm is more likely to select players that have a low ratio of games played to games available, which allows each player to play a fair amount of games

3.3. The system should have a means of storing, producing and accounting for differences in player ability

The algorithm can calculate a value that represents how a player is performing over time. It is a good measure of each players ability at any given time because it considers the players' performances across games – while being biased towards more recent fixtures.

3.4. As well as making fair picks, the system should also select players according to their ability – so that better players play more difficult/significant games.

The algorithm will select higher ability players for competitive games, favouring rating over ratio of games played when selecting players for the fixture.

3.5. The type of fixture (friendly, league, etc) should be accounted for when selecting players so that friendly games prioritise fair picks and competitive games prioritise high performing players

The algorithm aims to generate as fair picks as possible for non-competitive games (friendly matches), disregarding player ability, whereas for competitive games the algorithm favours higher performing players over fair picks (it still considers the ratio of games played for either type of fixture).

3.6. When making selections, the system takes into account multiple factors:

3.6.1. Ratio of number of games available to number of games played

The algorithm aims to make as fair picks as possible by using the ratio of games played to games available when calculating a team

3.6.2. The type of game (League, friendly, etc)

The algorithm has separate formulae for calculating League and friendly matches; with the intention of generating suitable selections for the type of game.

3.6.3. The team that the player is in

The players team is not directly used in the algorithm because its is an unusual situation for a player to play in a different age group, so its best for the coach to make decisions regarding this factor.

3.6.4. The performance of the player over time (This should be calculated using the past performances of the player, with recent games having a more significant impact than older games)

The algorithm does exactly this by taking all the past performances of the player from the database and calculating a rating that incorporates all of them, however the more recent games have a stronger impact on the rating than the older ones.

3.7. Fair picks should generally be prioritised over a strong team

For friendly games, fair picks are prioritised. For competitive games a strong team is prioritised

3.8. Generated selections are changeable and not fixed

The user has the ability to swap any player out of a line-up generated by the system. No selections are locked to the user

3.9. Data on future fixtures should be automatically retrieved via an API

It was my idea originally to use an API to get fixture information, however this did not turn out to be a feasible solution. In the new system the user enters the fixtures within the application.

3.10. System should be able to pick players based on multiple upcoming games

If players have already been assigned to a future fixture the system can account for this, however the system cannot consider multiple upcoming fixtures simultaneously when generating teams

4. There should be a user interface
   The user communicates with the system via a user interface in the form of a windows forms application
   - 4.1. The interface should be easy and intuitive to use
     It is very straightforward to navigate the application. The user can switch between past fixtures, future fixtures and view the players in the system with one button click.
   - 4.2. The interface should not require the user to enter SQL or VB.Net commands so that it is accessible and convenient to use
     All VB and SQL commands are hidden from the user. The user only has to enter plain text and click buttons for the system to operate.
   - 4.3. The application should be accessible to only the coach
     The application is local so only the user can access it. The database does not currently have a password set but it is possible to set a password on phpMyAdmin so that only the user can access the database
   - 4.4. The user should be able to view player data through the interface – including availability and player performance
     The user has the availability to view player data on the 'View Players' page and they can view the match performance and availability of players via the 'View Fixture' and 'Select Availability' pages.
   - 4.5. The program should present relevant data about fixtures to the user, including the ID, date, type and state
     The fixture DI, date, opponent, type, state and list of players is presented to the user when viewing a fixture
   - 4.6. For a future fixture: The program should present the user with a list of players playing in each fixture (or players assigned to play in a future fixture)
     The user is presented with a list of players either assigned to play in a fixture or that have already played in a fixture
   - 4.7. For a future fixture: The program should present the user with a list of available players to select from
     When the user wishes to alter an assigned team, they can select the 'Swap player' button in 'View fixture' which presents a list of all the players and shows their availability for the fixture
   - 4.8. For a past fixture: The program should present the user with the list of players that played in the game
     The user can view the list of players that played in a past fixture the same way they view the team for a future fixture
   - 4.9. The program should tell the user what fixtures need data to be input
     The program colour codes the fixtures on the past and upcoming fixtures pages. On the upcoming fixture page, a purple fixture means that there is not a valid team assigned to the fixture yet. A red fixture in past fixtures means that there is player performance data missing from that fixture.

4.10. The user should be able to enter player performances through the interface – and this data should be stored on the database
The user can enter player performance for past matches through the view fixtures page. These values are indeed uploaded and stored in the database

4.11. The program should allow the user to search for fixtures
The user can search for fixtures by FixtureID, Date or opponent for past and future fixtures

4.12. The user should be presented with the picks generated by the system
The user can view the list of picks generated by the system in the 'View fixtures' page

4.13. The user should be able to change the suggested picks through the interface
The user can use the 'Swap player' button in the view fixtures tab to alter the players assigned to a fixture.

4.13.1. The user should be given flexibility when adjusting selections – including the ability to assign players to older age-group games
The user has full flexibility to add any player to a fixture that they want, however the algorithm will only make selections within the usual rules of the team

4.13.2. The changes made by the user should not be restricted and the system should update data accordingly when changes are made
The system updates the database when the user changes the selections for a game.

4.13.3. Relevant information should be presented to the user when they are adjusting selections, to inform their decisions
When the user is selecting players they have the ability to click on each player and view all their statistics; including the number of games played and the rating of the player

4.14. Any data that is required to be entered by the user should be able to be entered through the interface
The user is not required to edit data from the database directly. Database functions can be performed in the interface

4.15. The user should be able to add new players to the system through the interface
The user has the ability to add new players to the system on the 'View players' page

4.16. The user should be able to add new fixtures to the system through the interface
The user can add new fixtures to the system from the 'Upcoming matches' page

Conclusion

I believe I was able to generate the system I set out to create and I am satisfied that the user interface, database and algorithms perform their roles well.

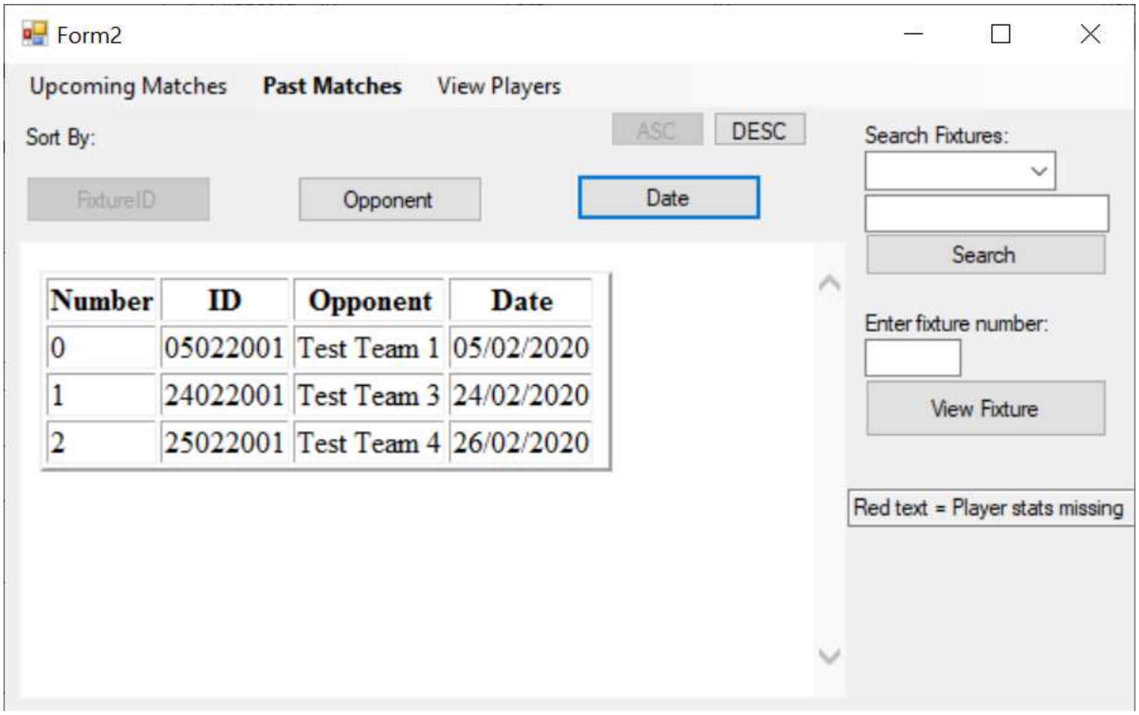
Nevertheless, I think that if I were to revisit the problem, there are a number of ways I could improve the system. Firstly I think I could make the user interface simpler and quicker to use. The method for manually changing the players assigned to a fixture could certainly be more streamlined – and a I think a ‘drag and drop’ type control would be easier to use and more intuitive for the user.

Secondly I think a more general page for manipulating the database directly would add another level of flexibility to the system. At the moment the system makes many communications with the database, however providing the user with a simple way of manually editing records in the database would add to the system.

Finally I think an email based availability system could add to the system, however it would take a significant amount of time to implement and potentially add unnecessary complexity to the system.

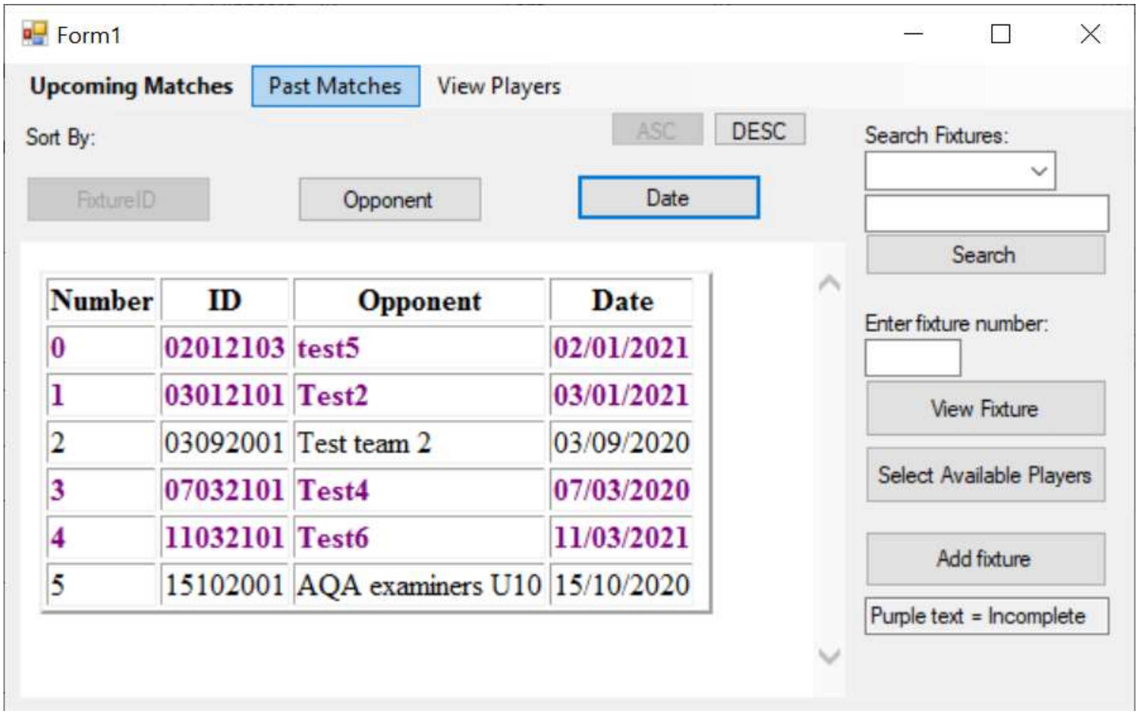
Testing Screenshots

1.1



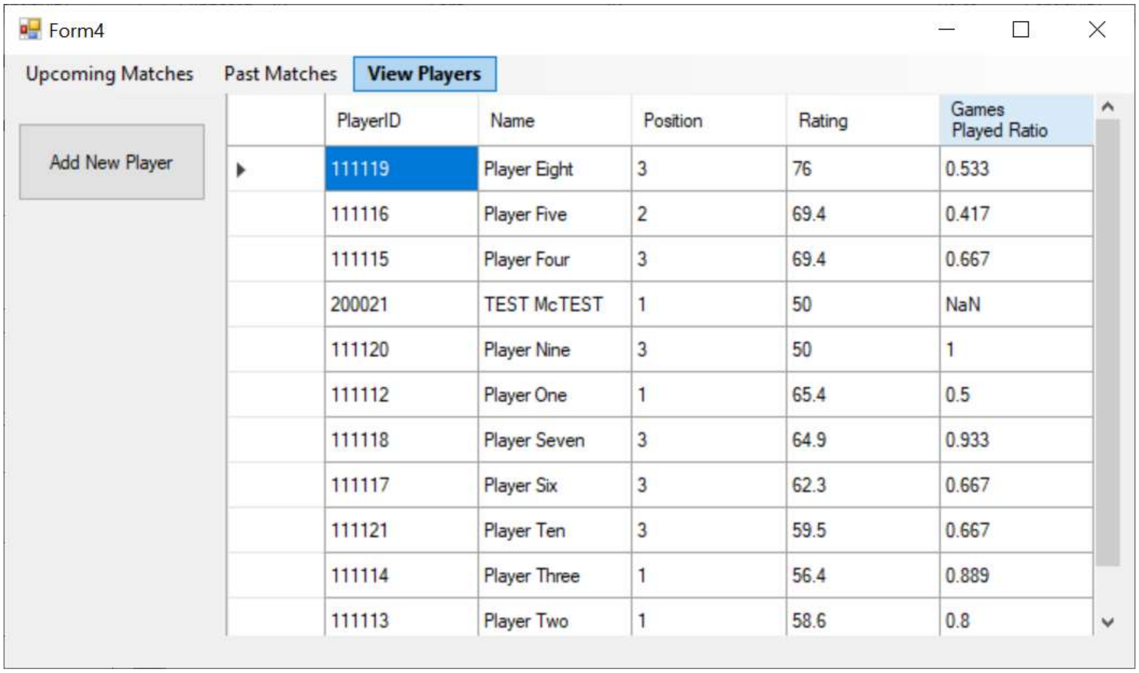
Number	ID	Opponent	Date
0	05022001	Test Team 1	05/02/2020
1	24022001	Test Team 3	24/02/2020
2	25022001	Test Team 4	26/02/2020

1.2



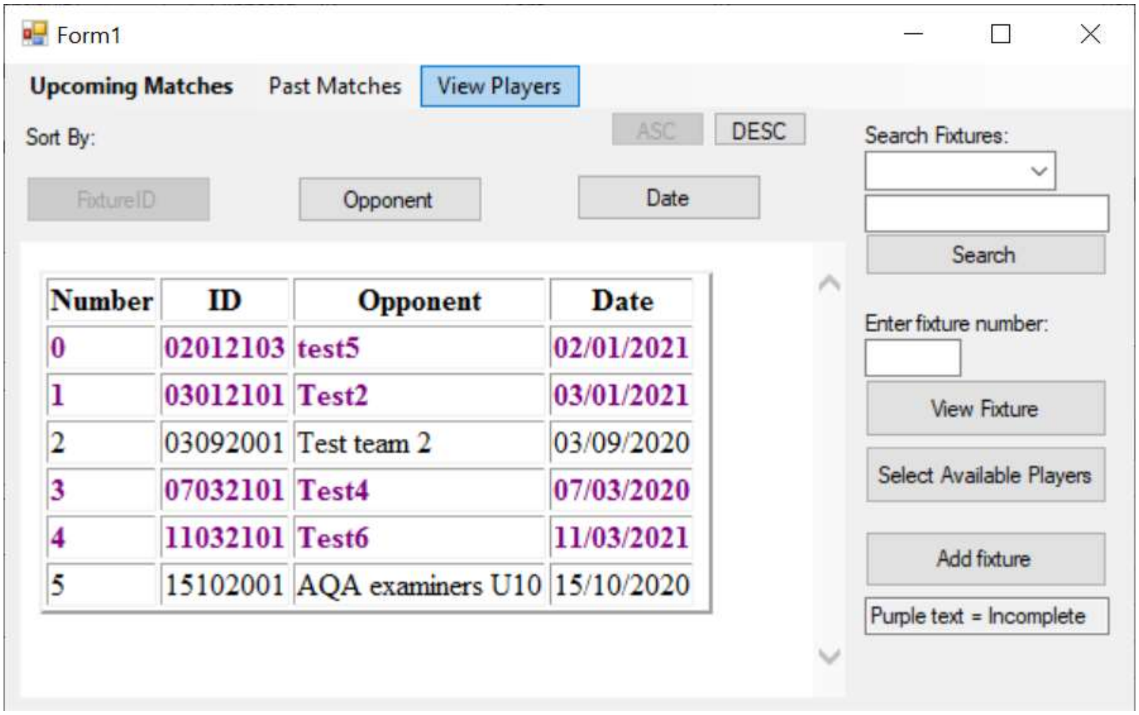
Number	ID	Opponent	Date
0	**02012103**	**test5**	**02/01/2021**
1	**03012101**	**Test2**	**03/01/2021**
2	03092001	Test team 2	03/09/2020
3	**07032101**	**Test4**	**07/03/2020**
4	**11032101**	**Test6**	**11/03/2021**
5	15102001	AQA examiners U10	15/10/2020

2.1



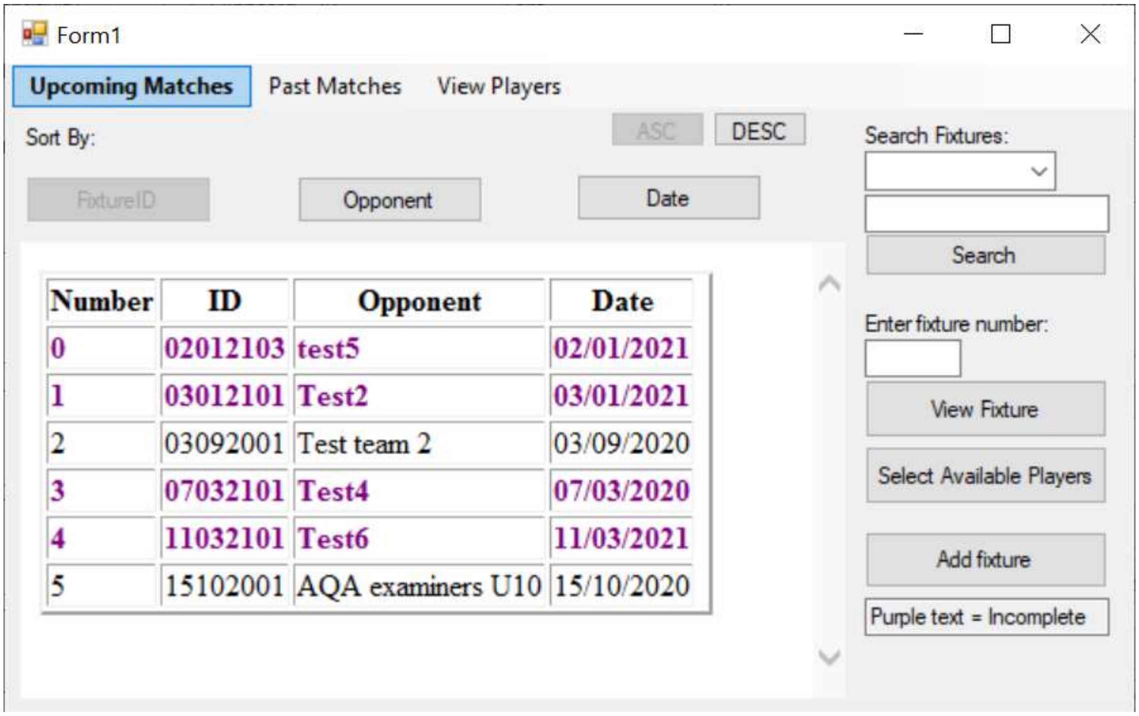
	PlayerID	Name	Position	Rating	Games Played Ratio
▶	111119	Player Eight	3	76	0.533
	111116	Player Five	2	69.4	0.417
	111115	Player Four	3	69.4	0.667
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	65.4	0.5
	111118	Player Seven	3	64.9	0.933
	111117	Player Six	3	62.3	0.667
	111121	Player Ten	3	59.5	0.667
	111114	Player Three	1	56.4	0.889
	111113	Player Two	1	58.6	0.8

2.2



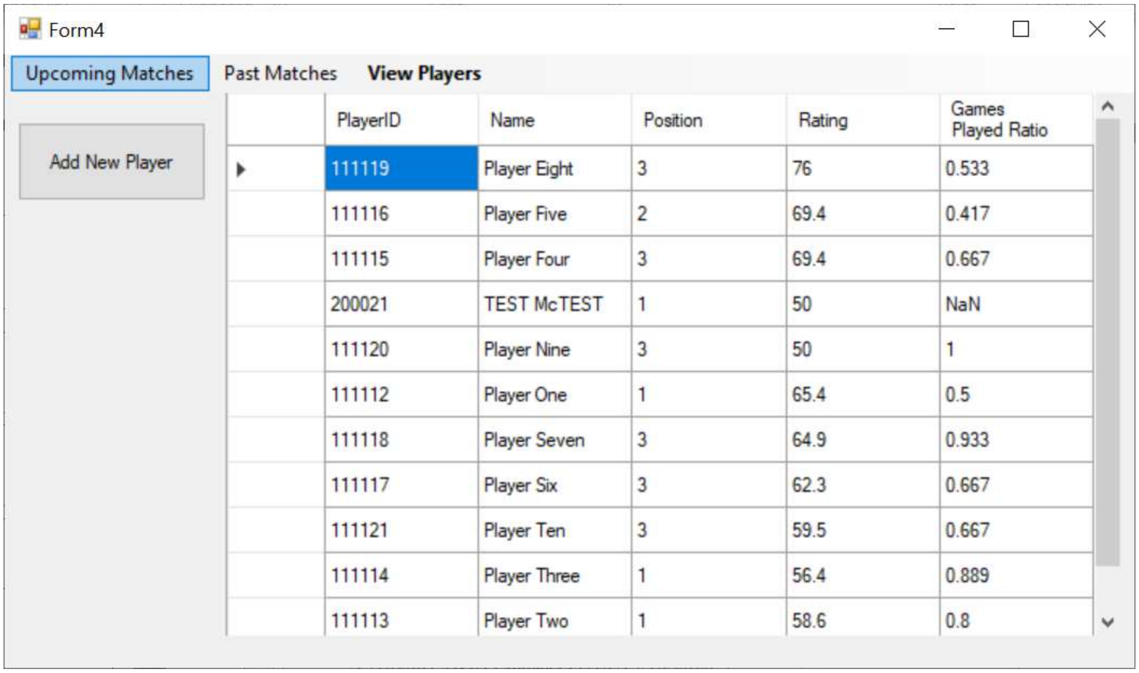
Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

3.1



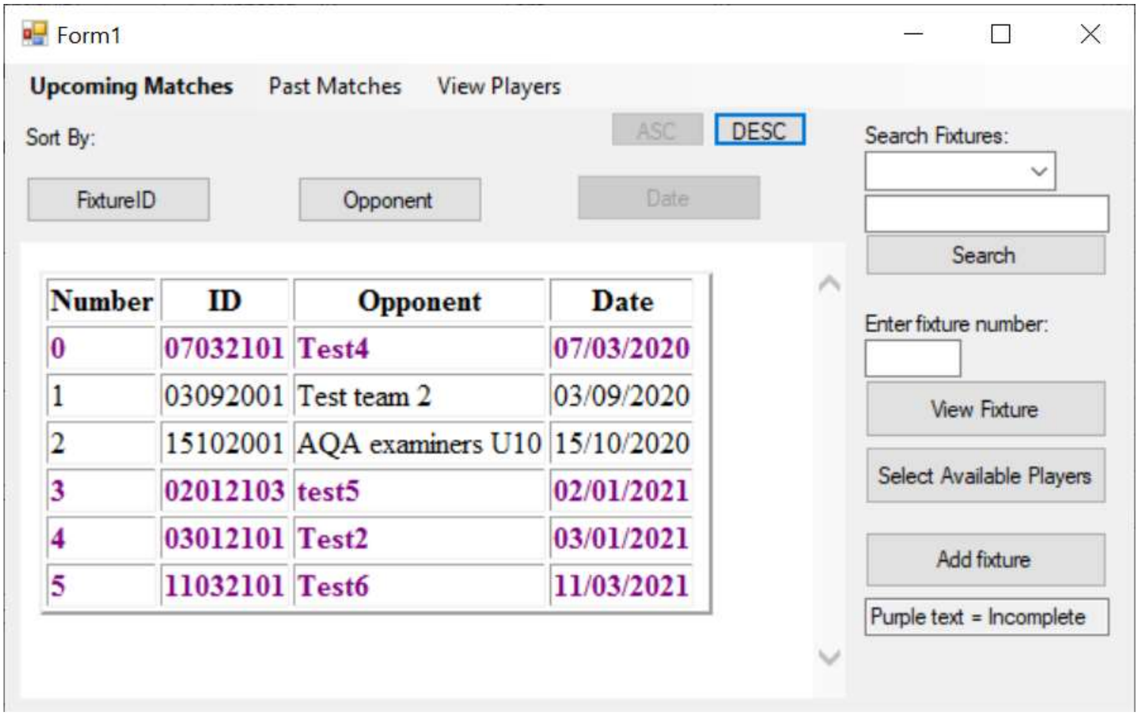
Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

3.2



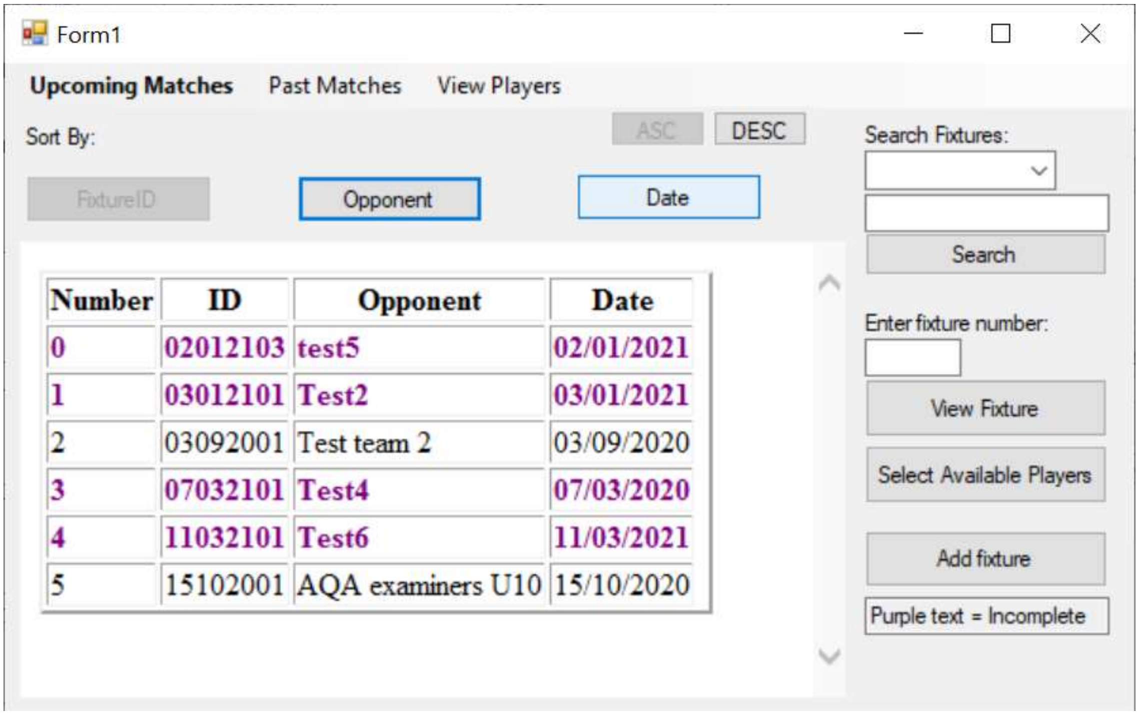
PlayerID	Name	Position	Rating	Games Played Ratio
111119	Player Eight	3	76	0.533
111116	Player Five	2	69.4	0.417
111115	Player Four	3	69.4	0.667
200021	TEST McTEST	1	50	NaN
111120	Player Nine	3	50	1
111112	Player One	1	65.4	0.5
111118	Player Seven	3	64.9	0.933
111117	Player Six	3	62.3	0.667
111121	Player Ten	3	59.5	0.667
111114	Player Three	1	56.4	0.889
111113	Player Two	1	58.6	0.8

4.1



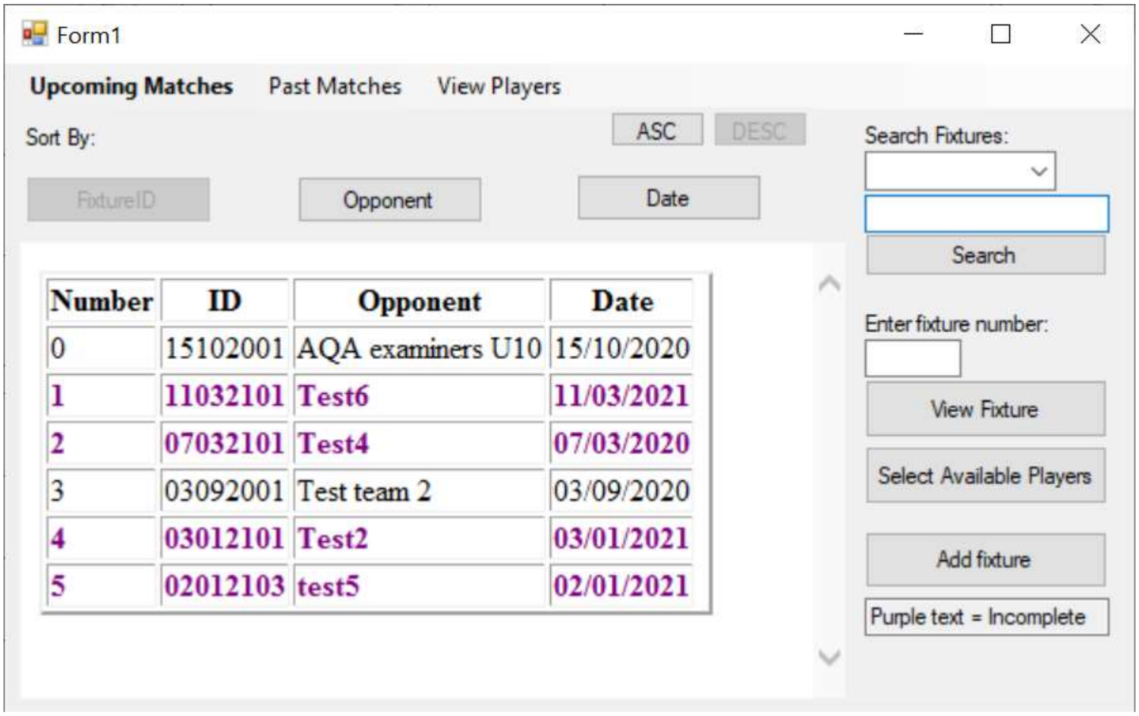
Number	ID	Opponent	Date
0	07032101	Test4	07/03/2020
1	03092001	Test team 2	03/09/2020
2	15102001	AQA examiners U10	15/10/2020
3	02012103	test5	02/01/2021
4	03012101	Test2	03/01/2021
5	11032101	Test6	11/03/2021

4.2



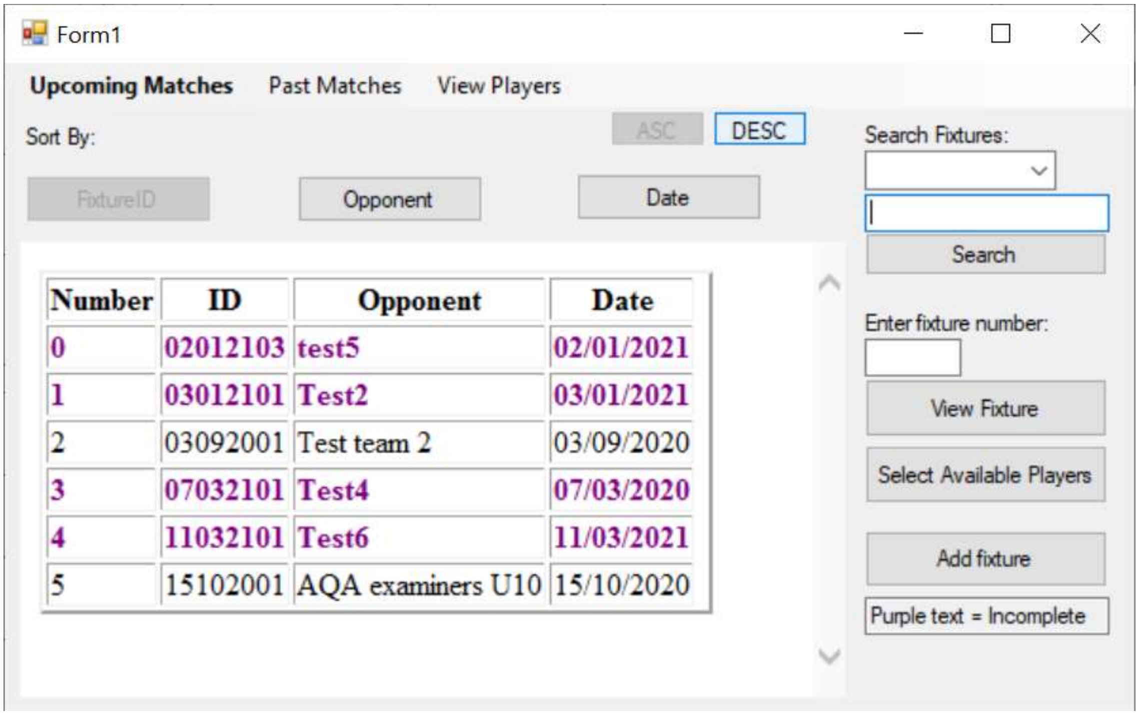
Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

5.1



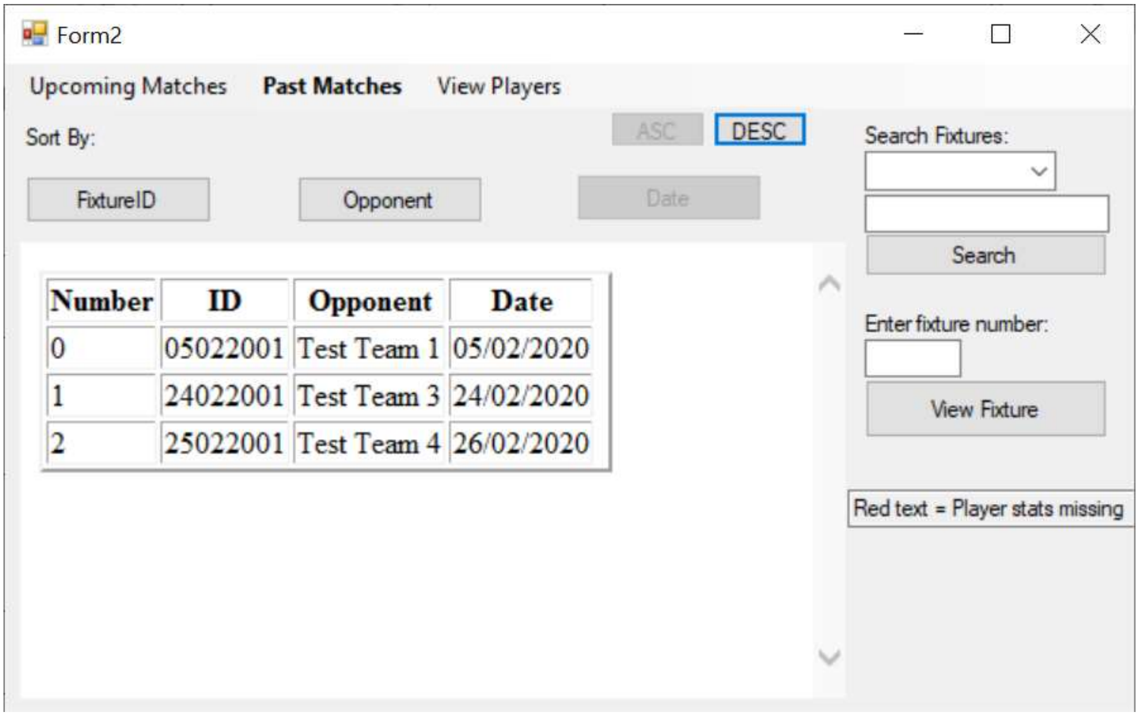
Number	ID	Opponent	Date
0	15102001	AQA examiners U10	15/10/2020
1	11032101	Test6	11/03/2021
2	07032101	Test4	07/03/2020
3	03092001	Test team 2	03/09/2020
4	03012101	Test2	03/01/2021
5	02012103	test5	02/01/2021

5.2



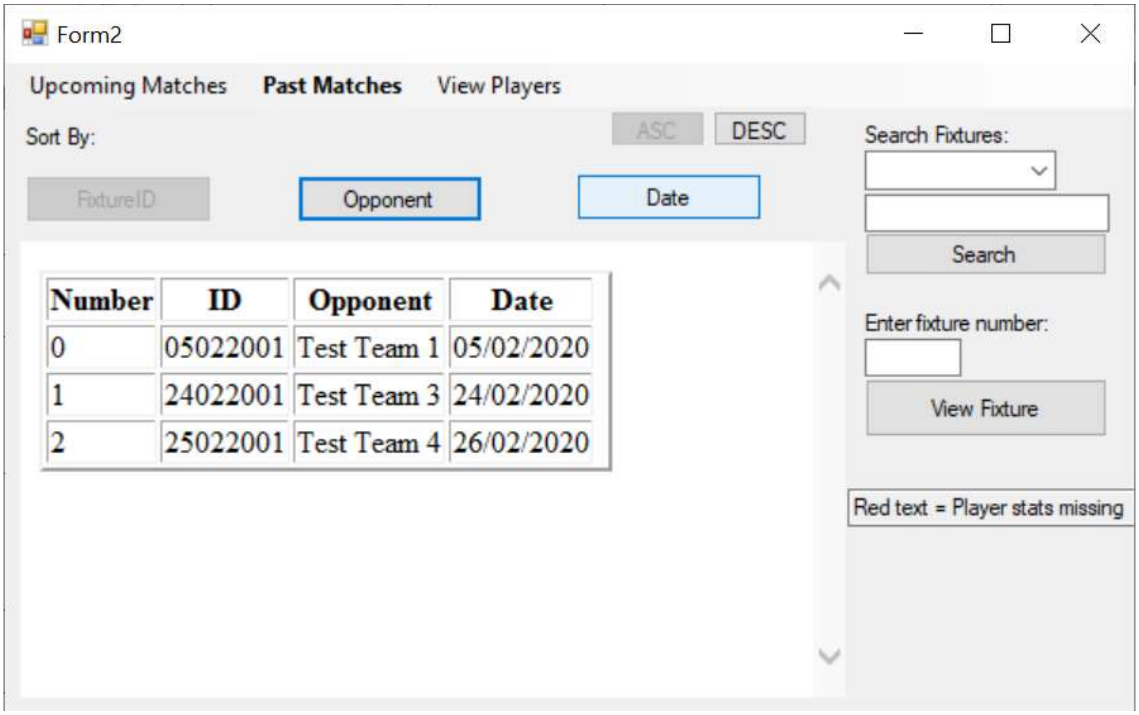
Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

6.1



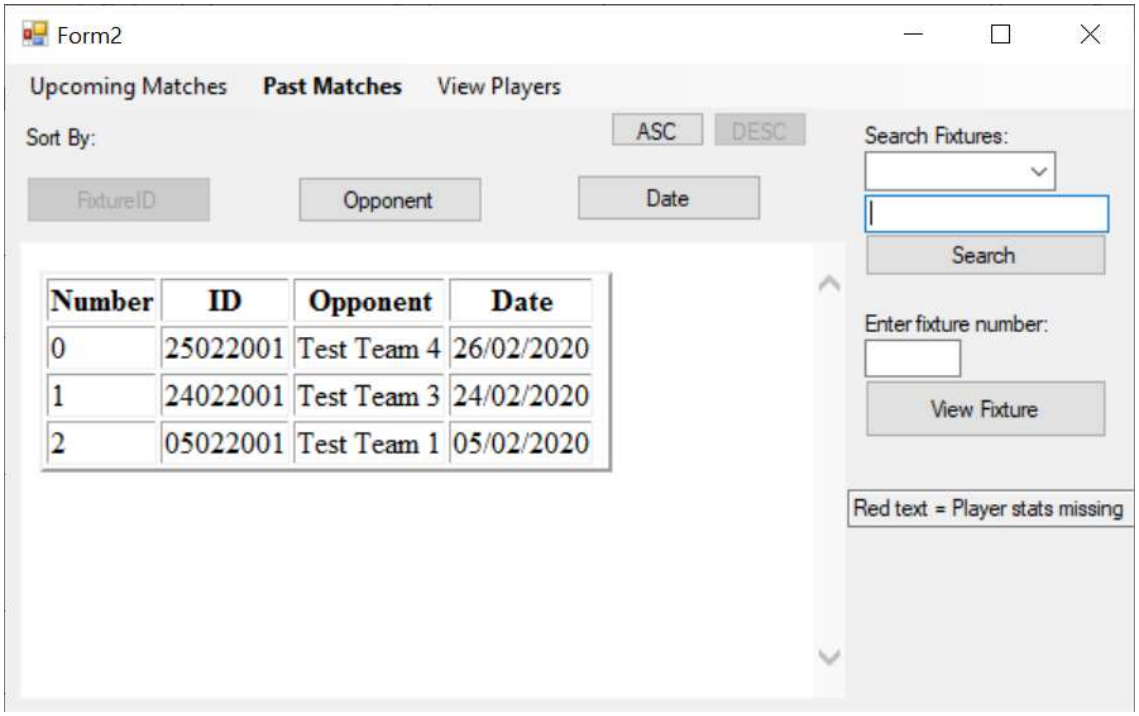
Number	ID	Opponent	Date
0	05022001	Test Team 1	05/02/2020
1	24022001	Test Team 3	24/02/2020
2	25022001	Test Team 4	26/02/2020

6.2



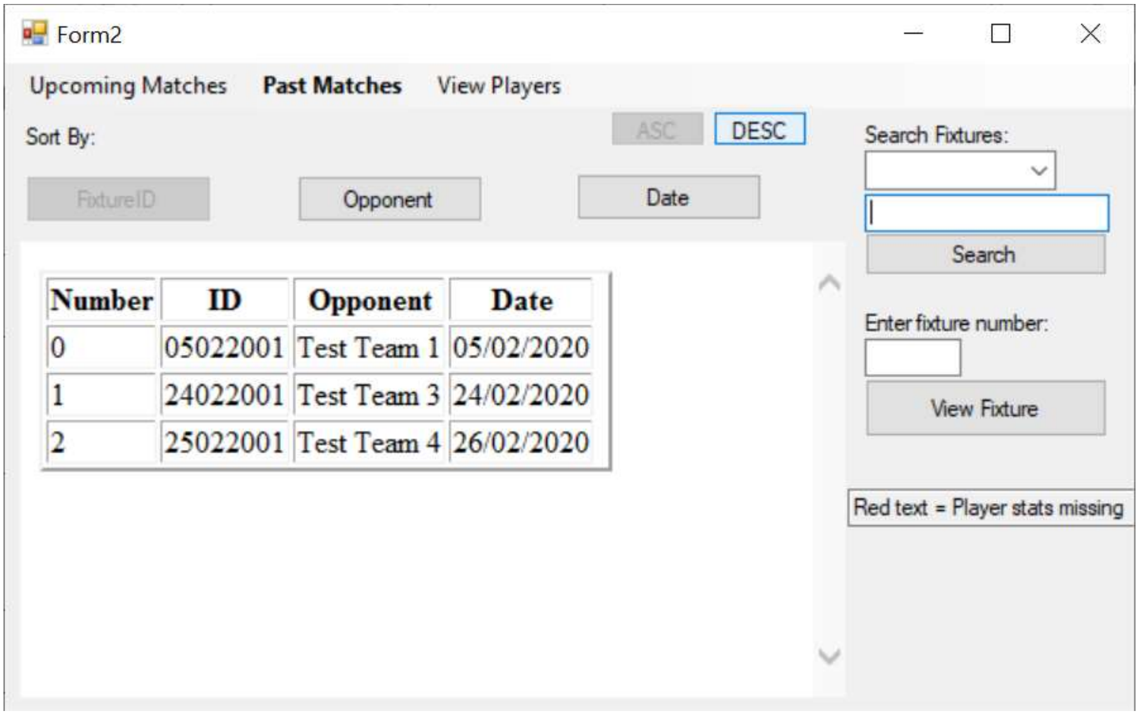
Number	ID	Opponent	Date
0	05022001	Test Team 1	05/02/2020
1	24022001	Test Team 3	24/02/2020
2	25022001	Test Team 4	26/02/2020

7.1



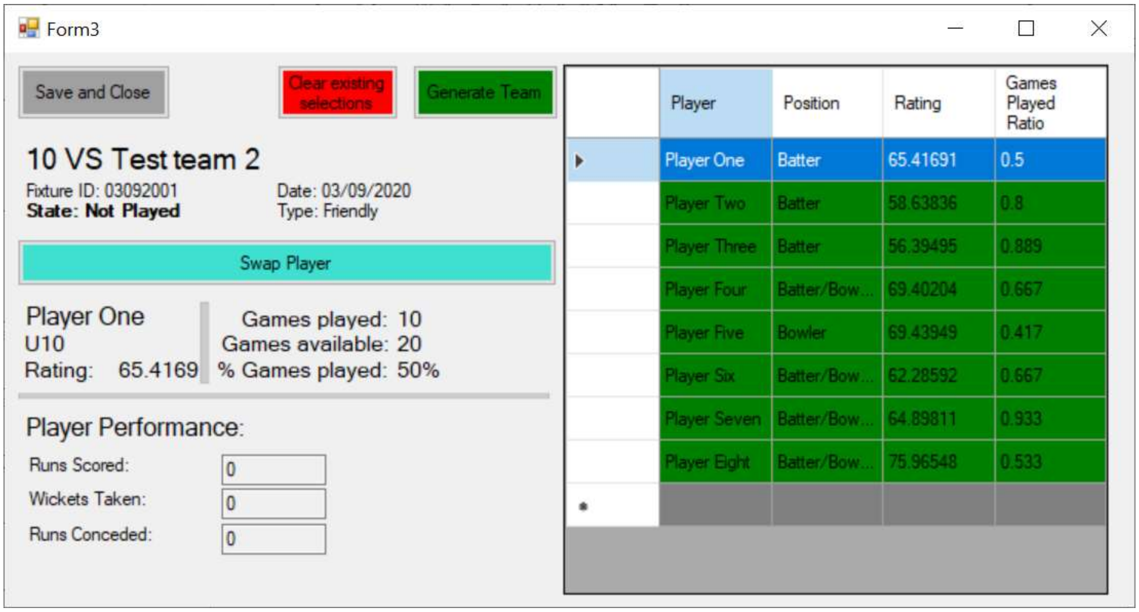
Number	ID	Opponent	Date
0	25022001	Test Team 4	26/02/2020
1	24022001	Test Team 3	24/02/2020
2	05022001	Test Team 1	05/02/2020

7.2



Number	ID	Opponent	Date
0	05022001	Test Team 1	05/02/2020
1	24022001	Test Team 3	24/02/2020
2	25022001	Test Team 4	26/02/2020

8.1



10 VS Test team 2

Fixture ID: 03092001
State: Not Played
Date: 03/09/2020
Type: Friendly

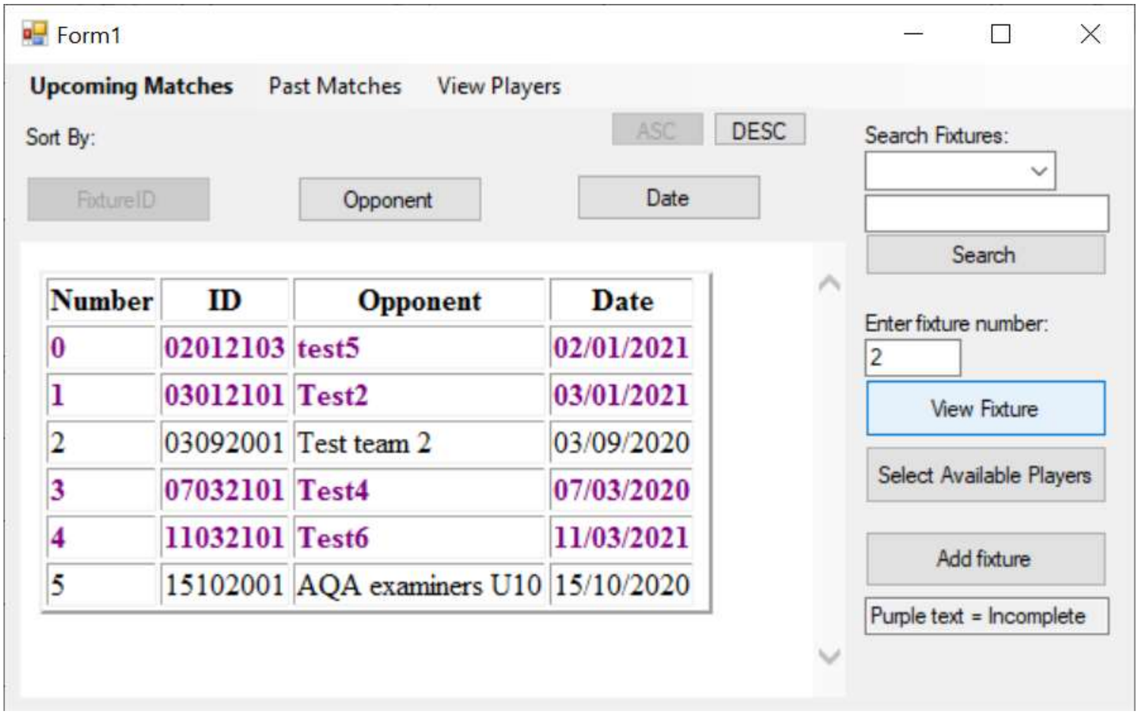
Player One
U10
Rating: 65.4169
Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

Runs Scored: 0
Wickets Taken: 0
Runs Conceded: 0

	Player	Position	Rating	Games Played Ratio
▶	Player One	Batter	65.41691	0.5
	Player Two	Batter	58.63836	0.8
	Player Three	Batter	56.39495	0.889
	Player Four	Batter/Bow...	69.40204	0.667
	Player Five	Bowler	69.43949	0.417
	Player Six	Batter/Bow...	62.28592	0.667
	Player Seven	Batter/Bow...	64.89811	0.933
	Player Eight	Batter/Bow...	75.96548	0.533
*				

8.2

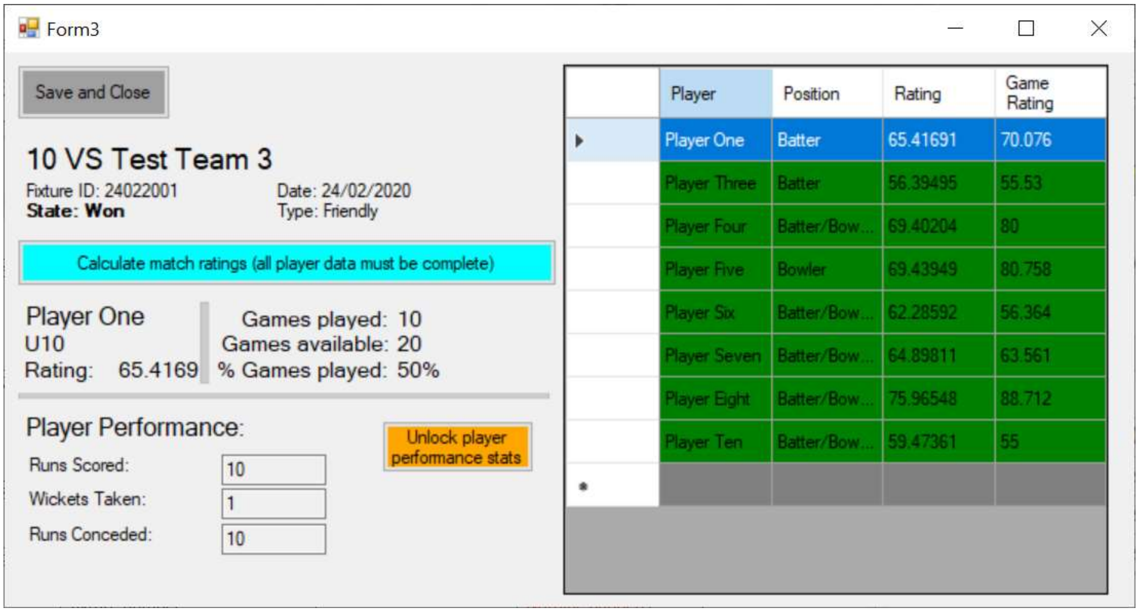


Enter fixture number: 2

Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

Purple text = Incomplete

9.1


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 3

Fixture ID: 24022001
Date: 24/02/2020
State: Won
Type: Friendly

Player One
U10
Rating: 65.4169

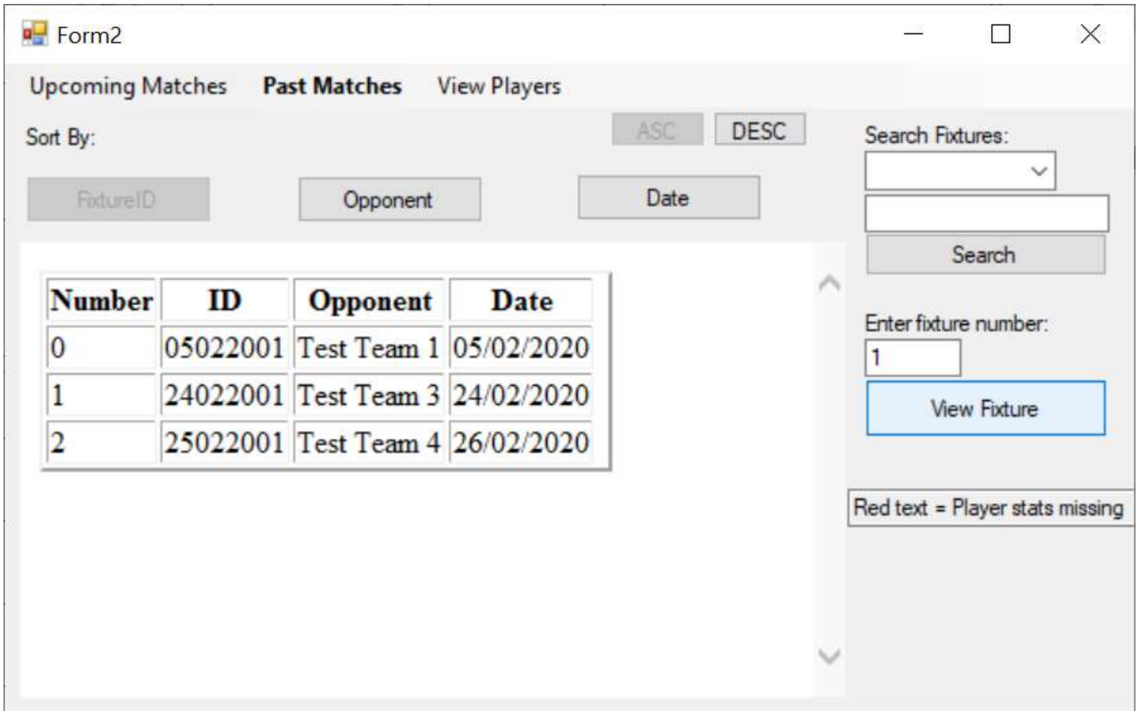
Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

Runs Scored: 10
Wickets Taken: 1
Runs Conceded: 10

Player	Position	Rating	Game Rating
Player One	Batter	65.41691	70.076
Player Three	Batter	56.39495	55.53
Player Four	Batter/Bow...	69.40204	80
Player Five	Bowler	69.43949	80.758
Player Six	Batter/Bow...	62.28592	56.364
Player Seven	Batter/Bow...	64.89811	63.561
Player Eight	Batter/Bow...	75.96548	88.712
Player Ten	Batter/Bow...	59.47361	55

9.2

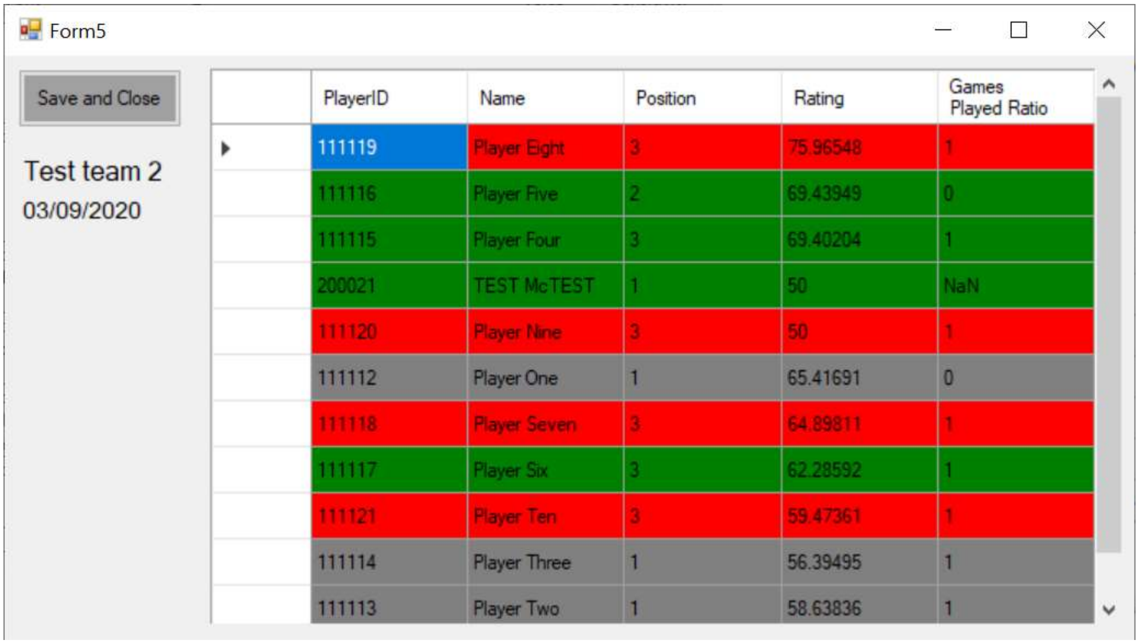


Number	ID	Opponent	Date
0	05022001	Test Team 1	05/02/2020
1	24022001	Test Team 3	24/02/2020
2	25022001	Test Team 4	26/02/2020

Enter fixture number: 1

Red text = Player stats missing

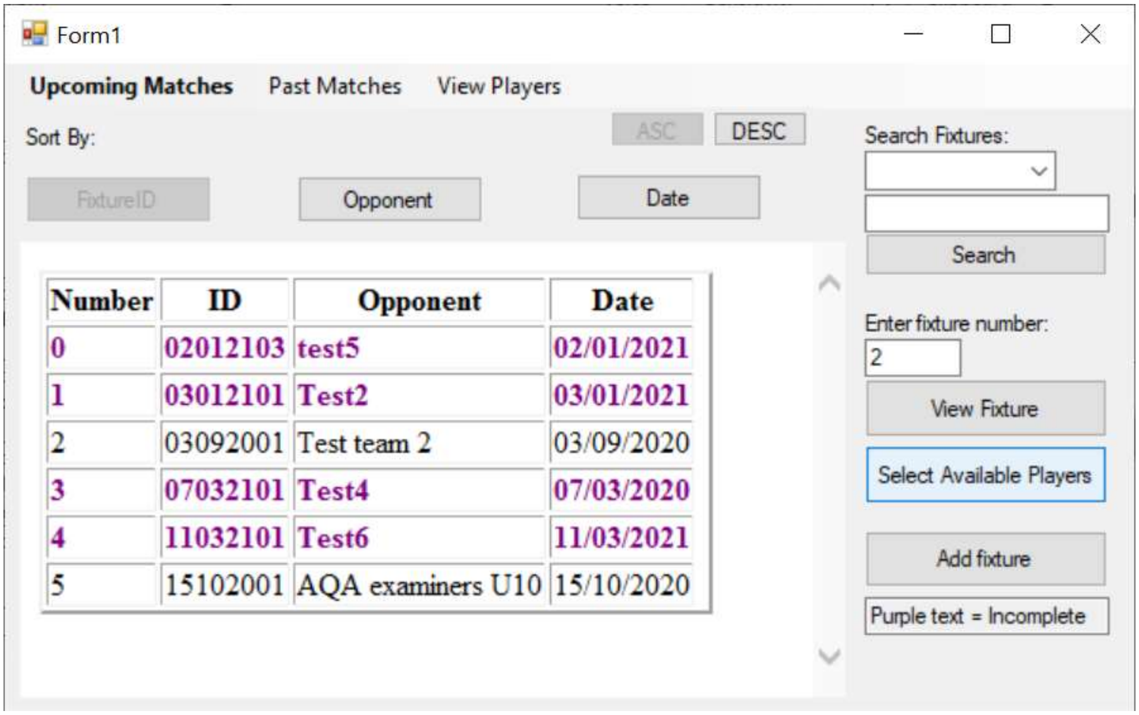
10.1



Test team 2
03/09/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
▸	111119	Player Eight	3	75.96548	1
	111116	Player Five	2	69.43949	0
	111115	Player Four	3	69.40204	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	65.41691	0
	111118	Player Seven	3	64.89811	1
	111117	Player Six	3	62.28592	1
	111121	Player Ten	3	59.47361	1
	111114	Player Three	1	56.39495	1
	111113	Player Two	1	58.63836	1

10.2

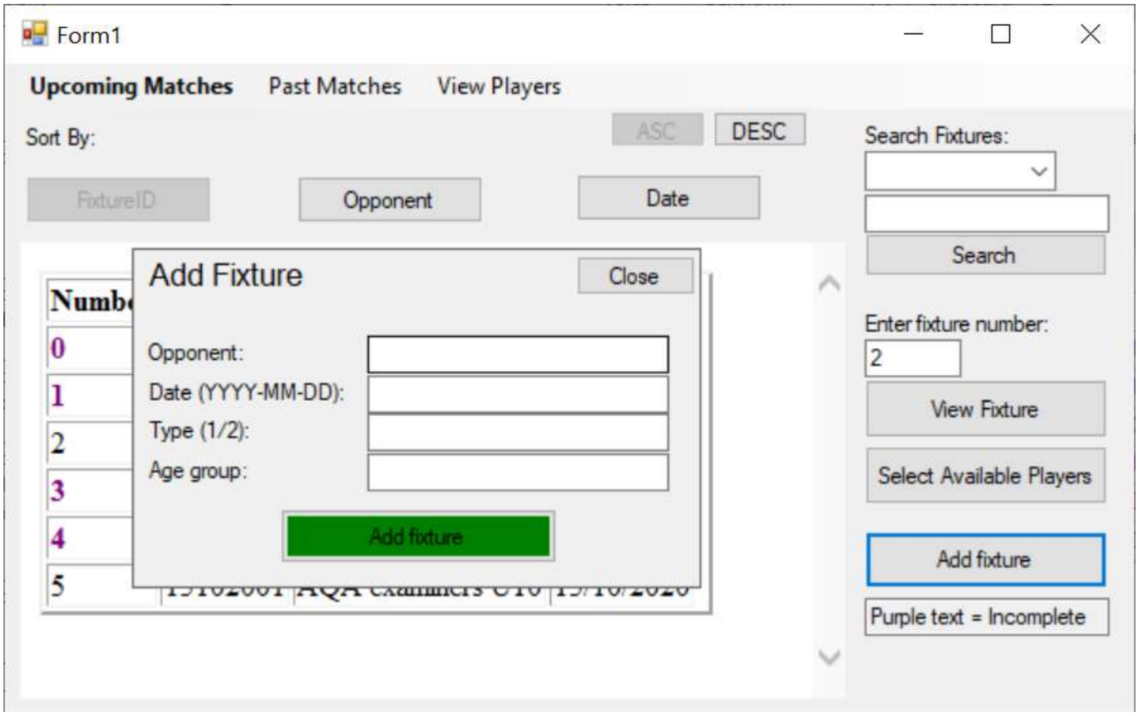


Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

Enter fixture number: 2

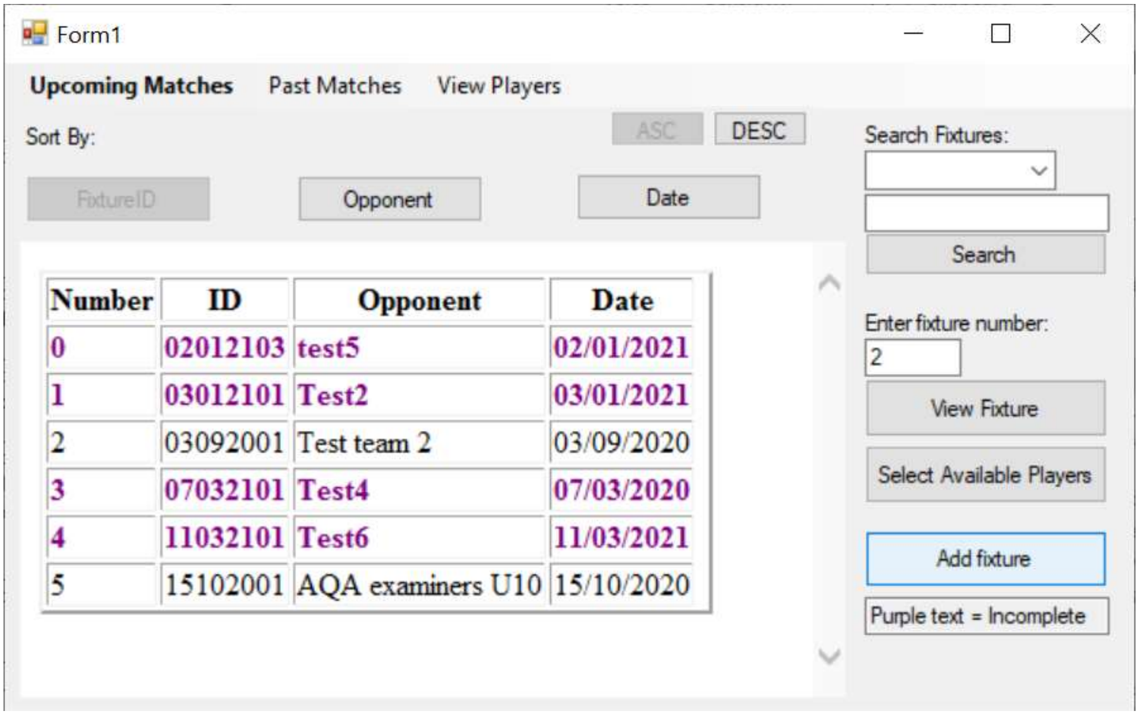
Purple text = Incomplete

11.1



Enter fixture number: 2

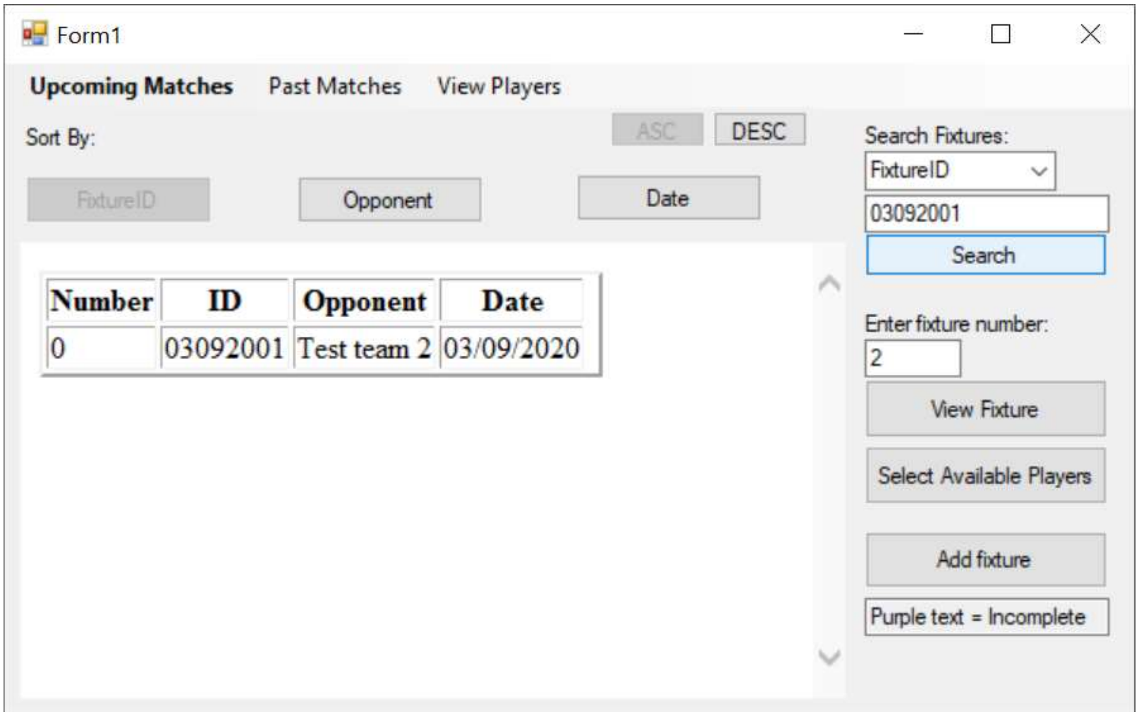
11.2



Enter fixture number: 2

Number	ID	Opponent	Date
0	02012103	test5	02/01/2021
1	03012101	Test2	03/01/2021
2	03092001	Test team 2	03/09/2020
3	07032101	Test4	07/03/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

12.1



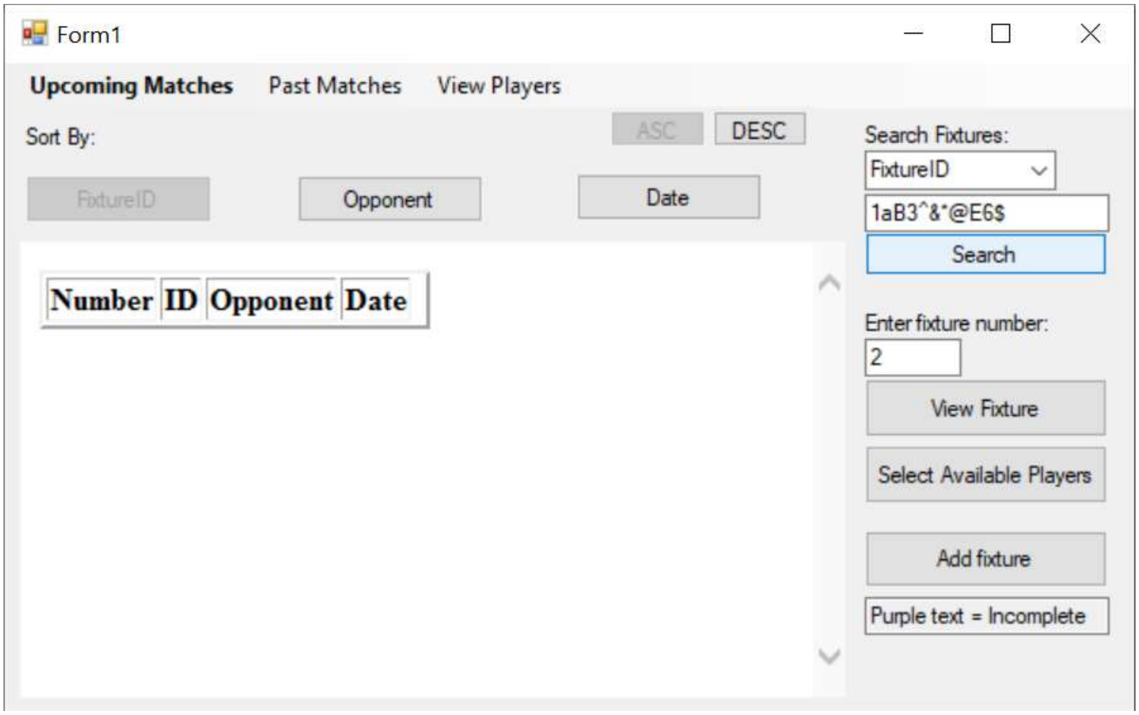
Search Fixtures: FixtureID

03092001

Enter fixture number: 2

Number	ID	Opponent	Date
0	03092001	Test team 2	03/09/2020

12.2



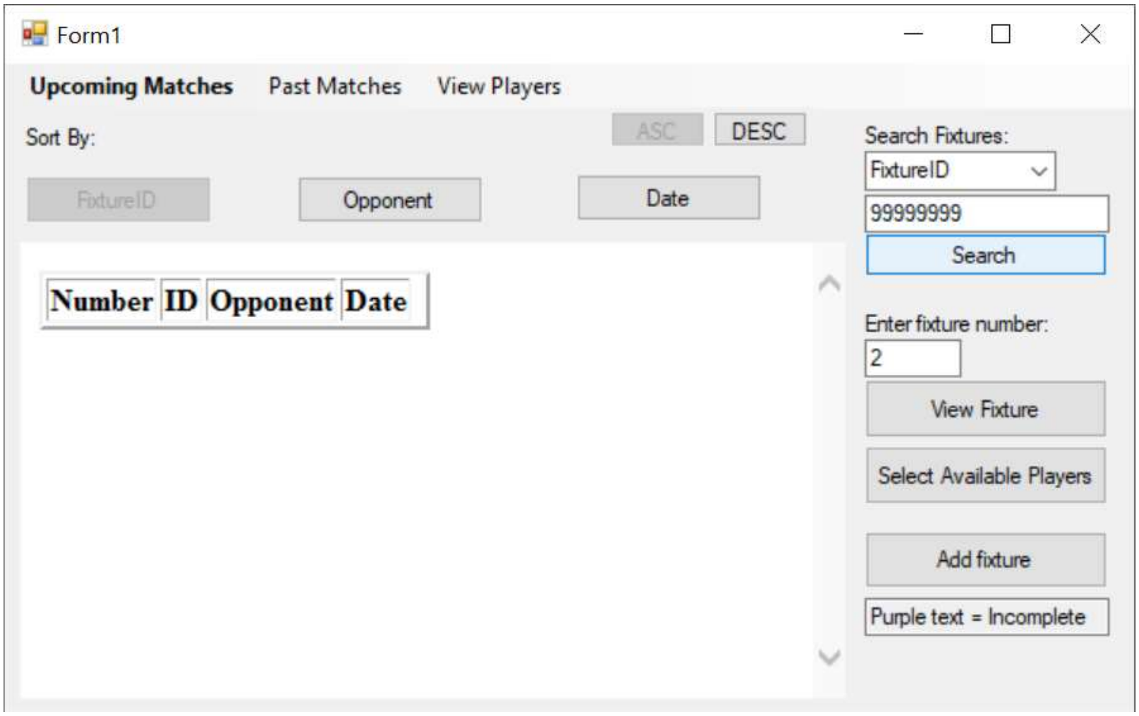
Search Fixtures: FixtureID

1aB3^&*@E6$

Enter fixture number: 2

Number	ID	Opponent	Date

12.3



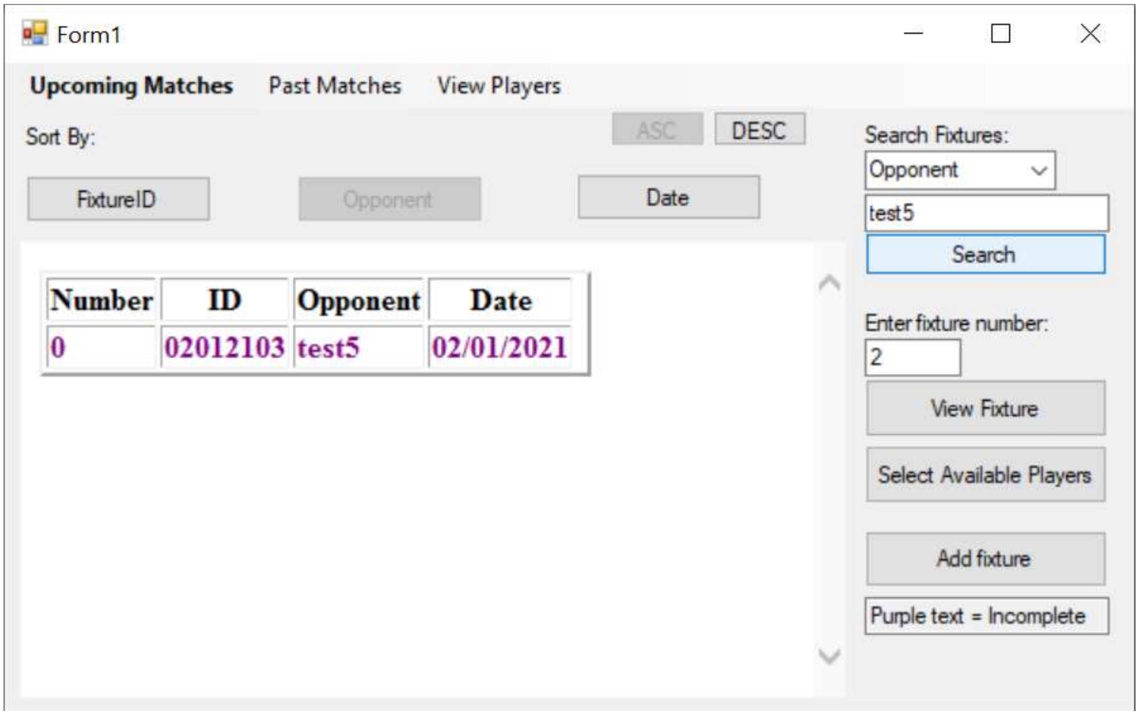
Search Fixtures: FixtureID

99999999

Enter fixture number: 2

Number	ID	Opponent	Date

13.1



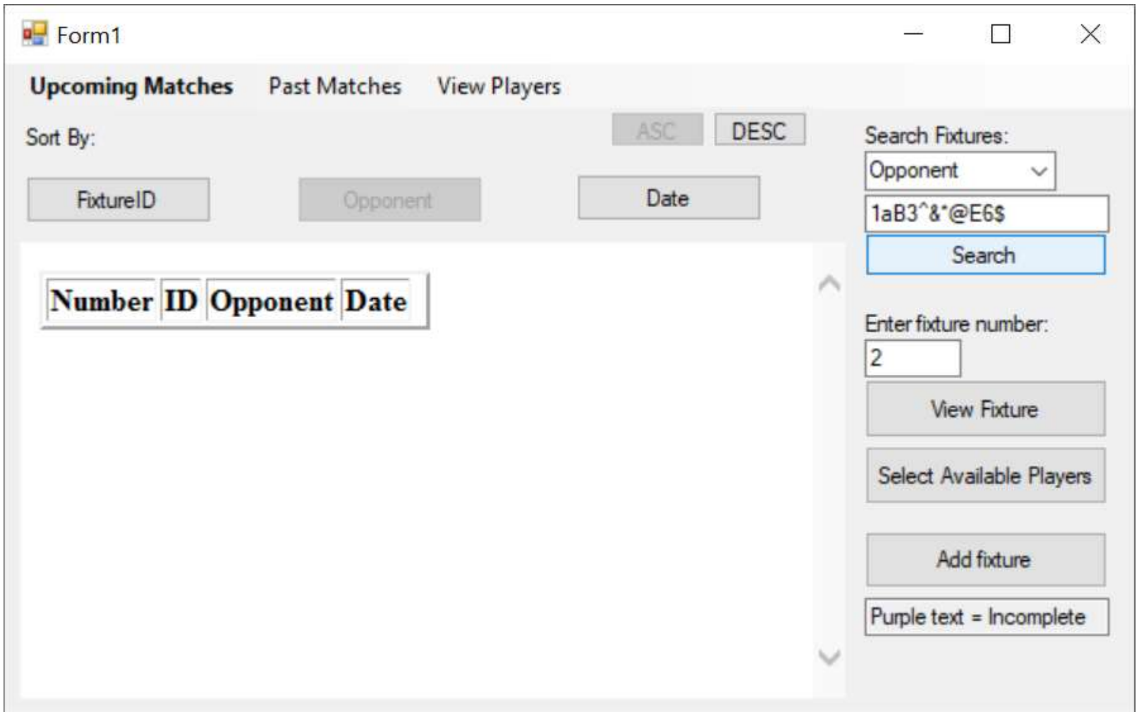
Search Fixtures: Opponent

test5

Enter fixture number: 2

Number	ID	Opponent	Date
0	02012103	test5	02/01/2021

13.2



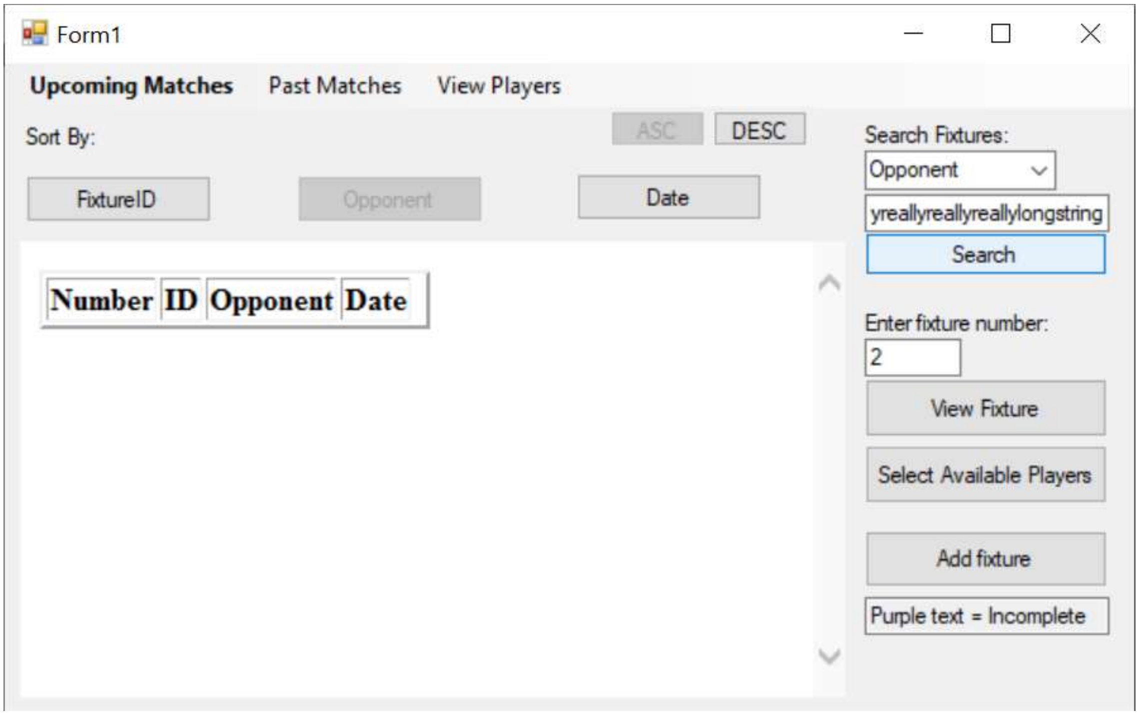
Search Fixtures: Opponent

1aB3^&*@E6$

Enter fixture number: 2

Number	ID	Opponent	Date

13.3



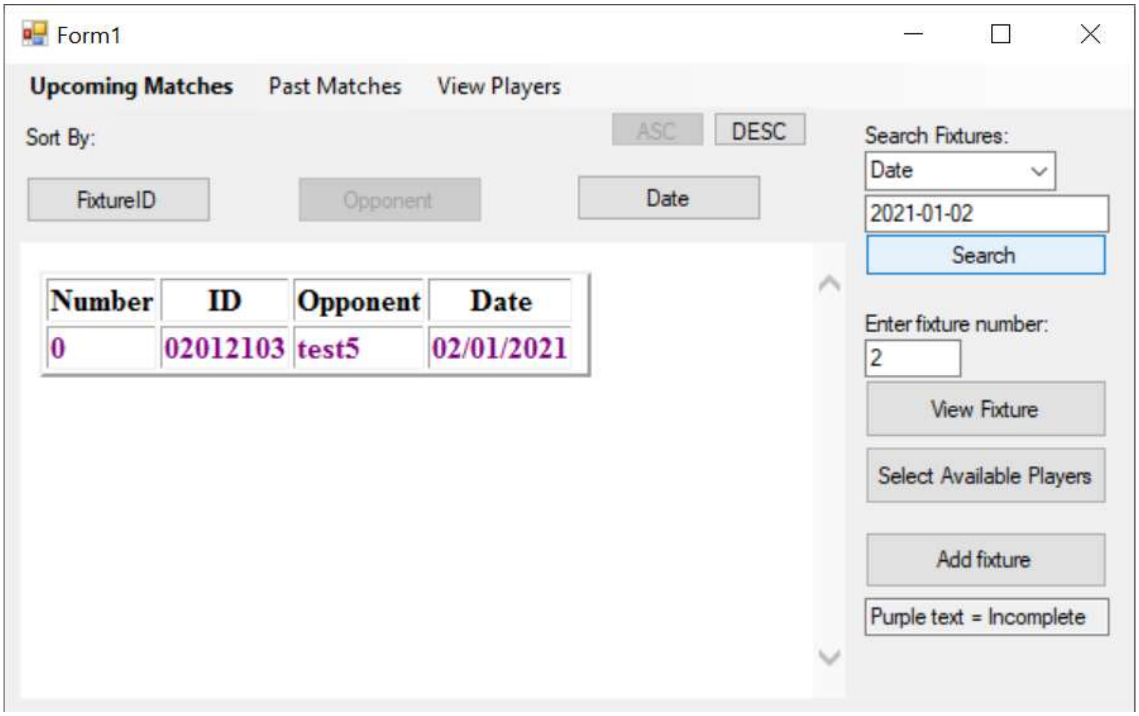
Search Fixtures: Opponent

yreallyreallyreallylongstring

Enter fixture number: 2

Number	ID	Opponent	Date

14.1



Search Fixtures: Date

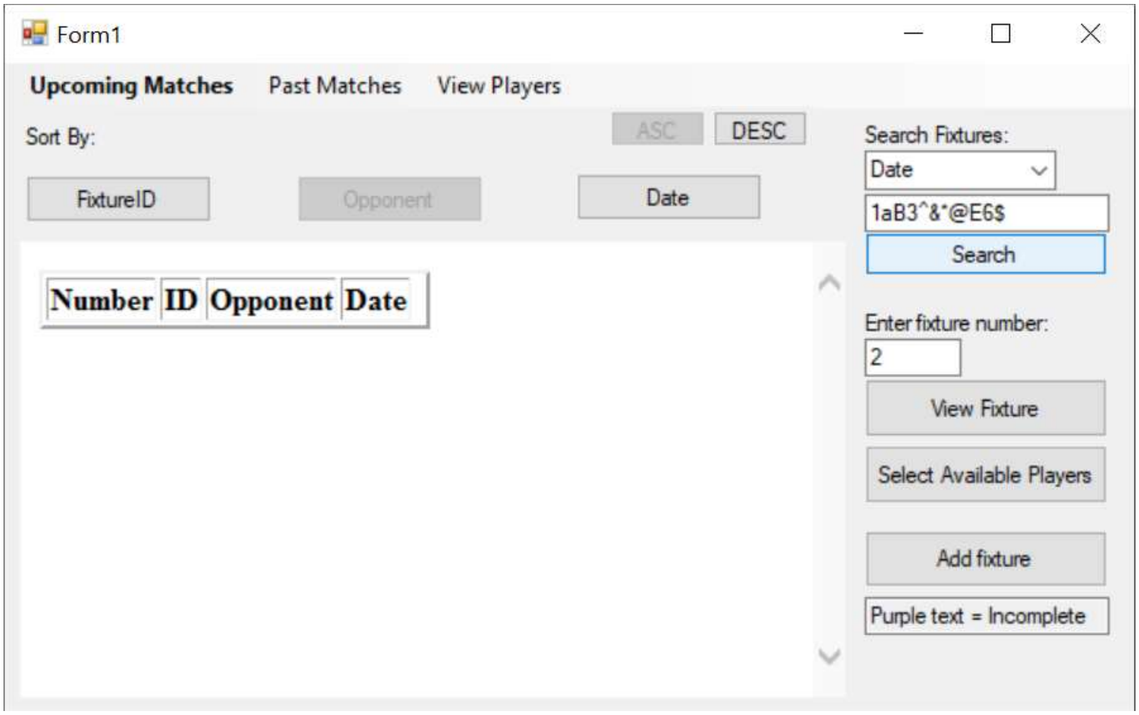
2021-01-02

Enter fixture number: 2

Purple text = Incomplete

Number	ID	Opponent	Date
0	02012103	test5	02/01/2021

14.2



Search Fixtures: Date

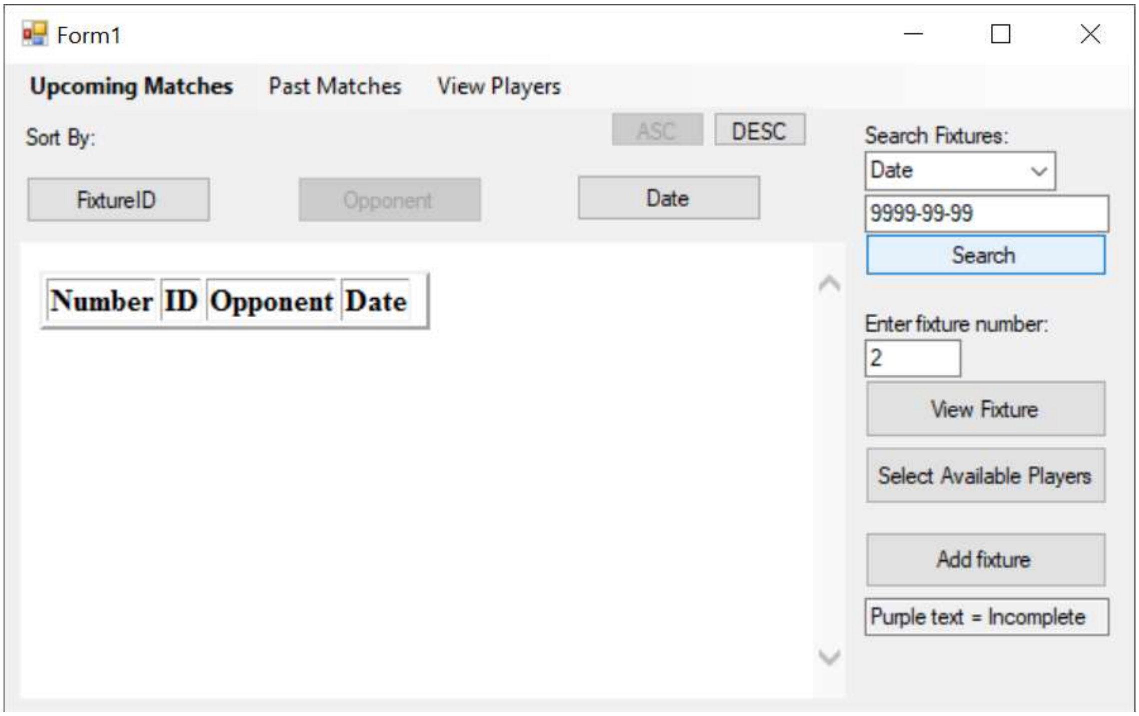
1aB3^&*@E6$

Enter fixture number: 2

Purple text = Incomplete

Number	ID	Opponent	Date

14.3



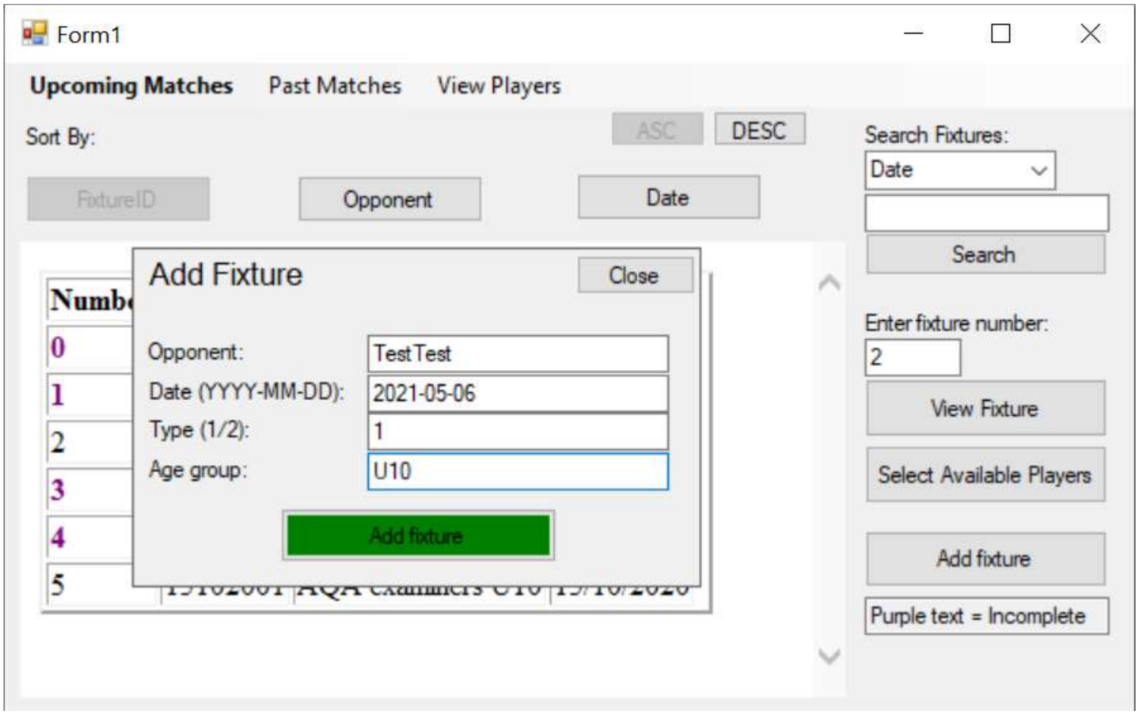
Search Fixtures: Date

9999-99-99

Enter fixture number: 2

Number	ID	Opponent	Date

15.1



Search Fixtures: Date

Enter fixture number: 2

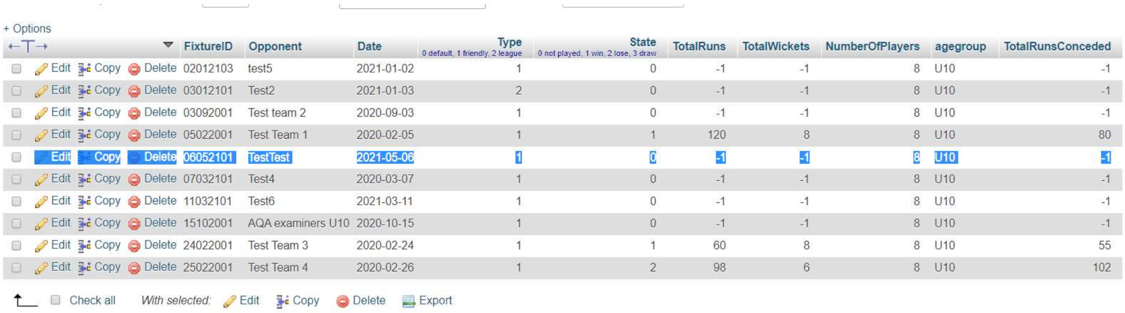
Opponent: TestTest

Date (YYYY-MM-DD): 2021-05-06

Type (1/2): 1

Age group: U10

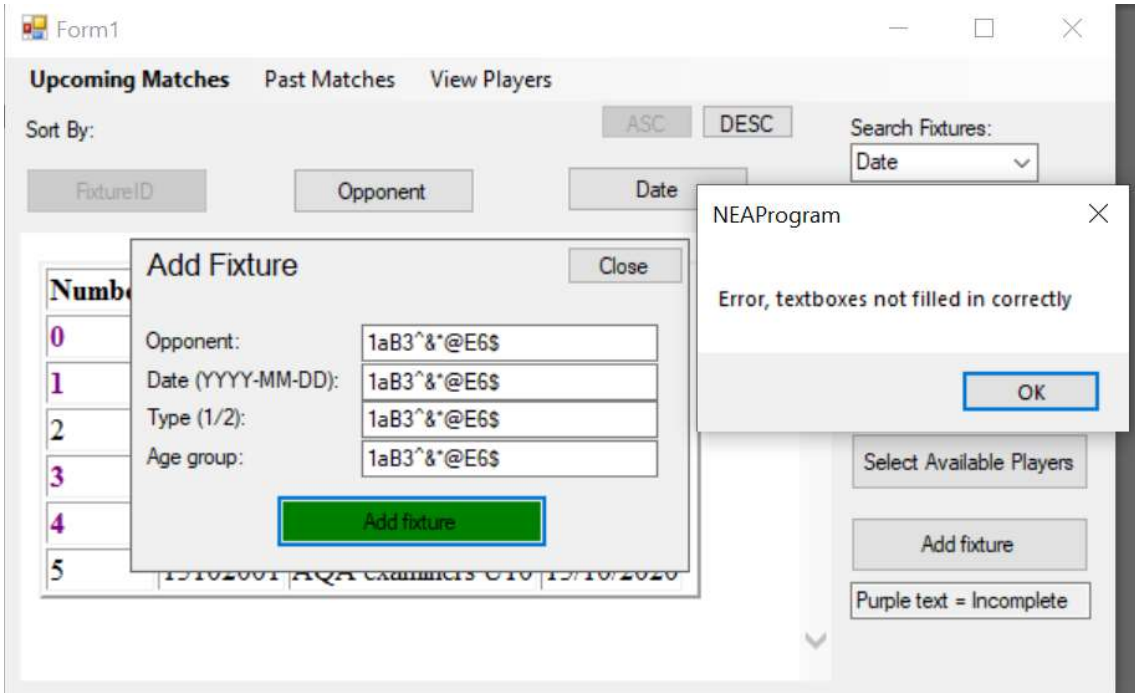
Numb			
0			
1			
2			
3			
4			
5	15102001	AQA examiners U10	15/10/2020



+ Options

	FixtureID	Opponent	Date	Type 0 default, 1 friendly, 2 league	State 0 not played, 1 win, 2 lose, 3 draw	TotalRuns	TotalWickets	NumberOfPlayers	agegroup	TotalRunsConceded
□ Edit Copy Delete	02012103	test5	2021-01-02	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	03012101	Test2	2021-01-03	2	0	-1	-1	8	U10	-1
□ Edit Copy Delete	03092001	Test team 2	2020-09-03	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	05022001	Test Team 1	2020-02-05	1	1	120	8	8	U10	80
□ Edit Copy Delete	06052101	TestTest	2021-05-06	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	07032101	Test4	2020-03-07	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	11032101	Test6	2021-03-11	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	15102001	AQA examiners U10	2020-10-15	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	24022001	Test Team 3	2020-02-24	1	1	60	8	8	U10	55
□ Edit Copy Delete	25022001	Test Team 4	2020-02-26	1	2	98	6	8	U10	102

15.2



Add Fixture

Opponent: 1aB3^&*@E6$

Date (YYYY-MM-DD): 1aB3^&*@E6$

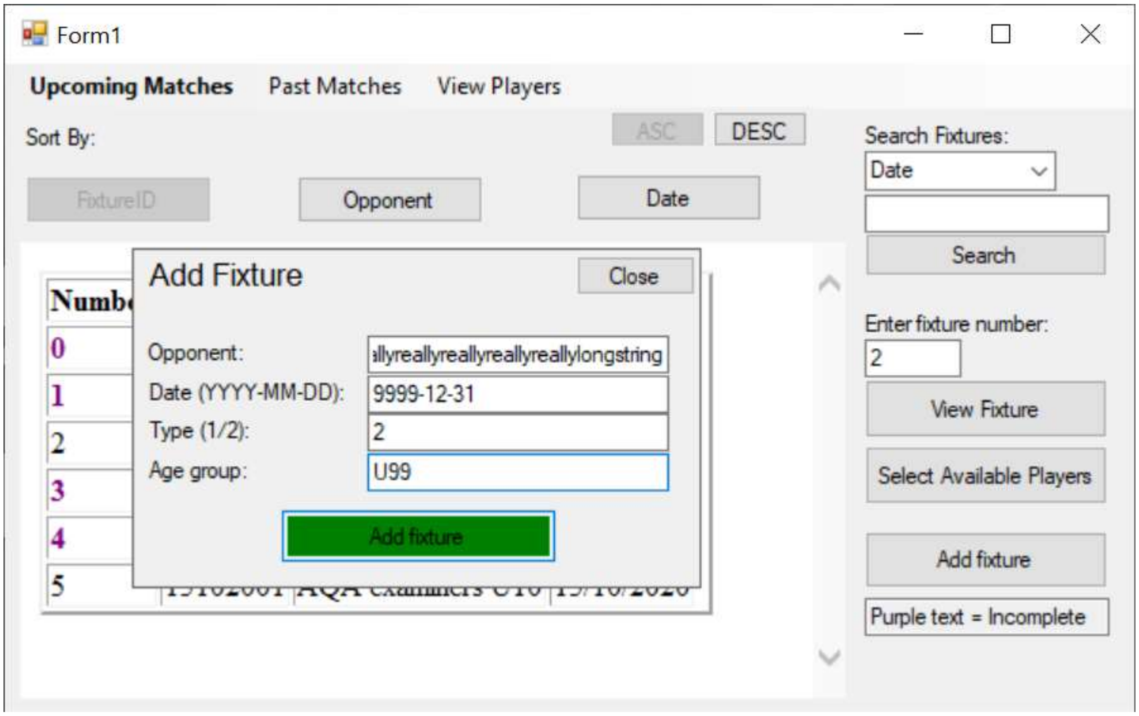
Type (1/2): 1aB3^&*@E6$

Age group: 1aB3^&*@E6$

NEAProgram

Error, textboxes not filled in correctly

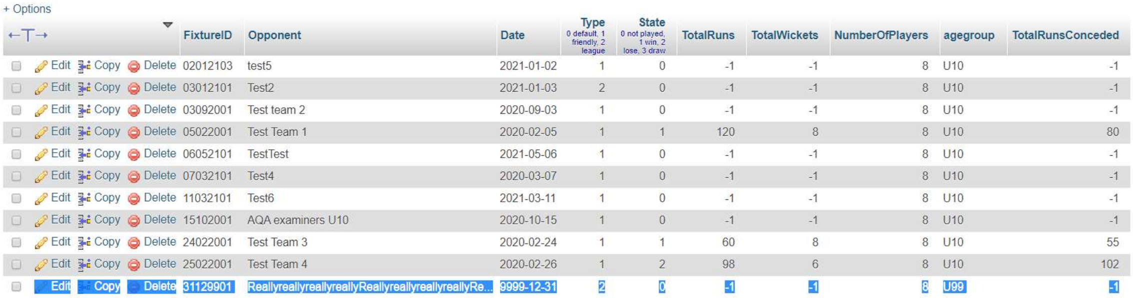
15.3



Search Fixtures: Date

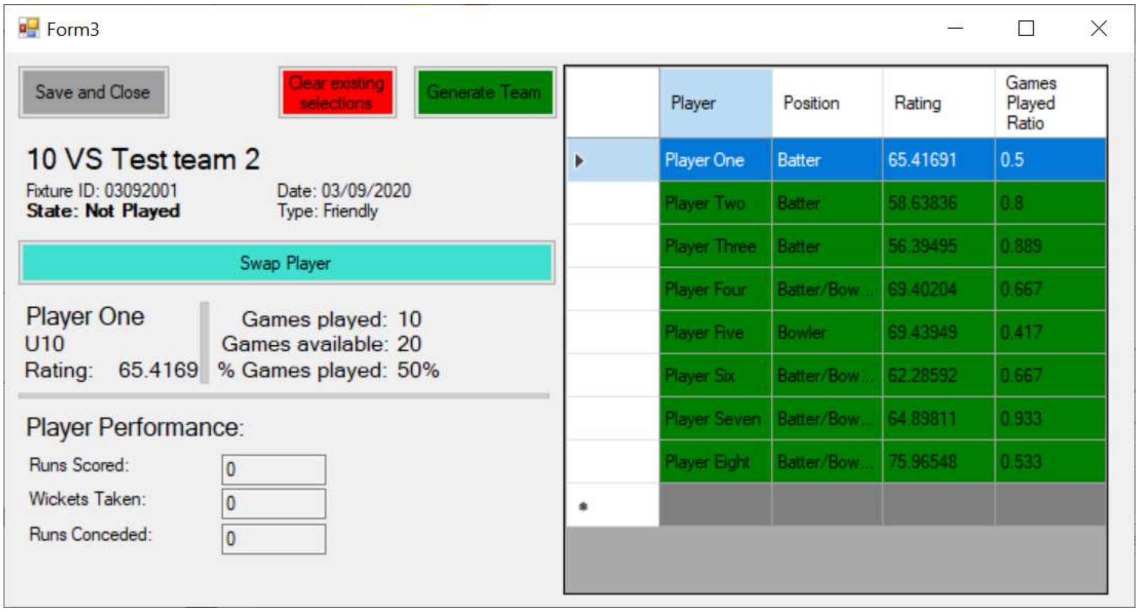
Enter fixture number: 2

Opponent:	allyreallyreallyreallyreallylongstring
Date (YYYY-MM-DD):	9999-12-31
Type (1/2):	2
Age group:	U99



	FixtureID	Opponent	Date	Type 0 default, 1 friendly, 2 league	State 0 not played, 1 win, 2 lose, 3 draw	TotalRuns	TotalWickets	NumberOfPlayers	agegroup	TotalRunsConceded
□ Edit Copy Delete	02012103	test5	2021-01-02	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	03012101	Test2	2021-01-03	2	0	-1	-1	8	U10	-1
□ Edit Copy Delete	03092001	Test team 2	2020-09-03	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	05022001	Test Team 1	2020-02-05	1	1	120	8	8	U10	80
□ Edit Copy Delete	06052101	TestTest	2021-05-06	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	07032101	Test4	2020-03-07	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	11032101	Test6	2021-03-11	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	15102001	AQA examiners U10	2020-10-15	1	0	-1	-1	8	U10	-1
□ Edit Copy Delete	24022001	Test Team 3	2020-02-24	1	1	60	8	8	U10	55
□ Edit Copy Delete	25022001	Test Team 4	2020-02-26	1	2	98	6	8	U10	102
□ Edit Copy Delete	31129901	ReallyreallyreallyreallyReallyreallyreallyreallyRe...	9999-12-31	2	0	-1	-1	8	U99	-1

16.1



10 VS Test team 2

Fixture ID: 03092001
State: Not Played

Date: 03/09/2020
Type: Friendly

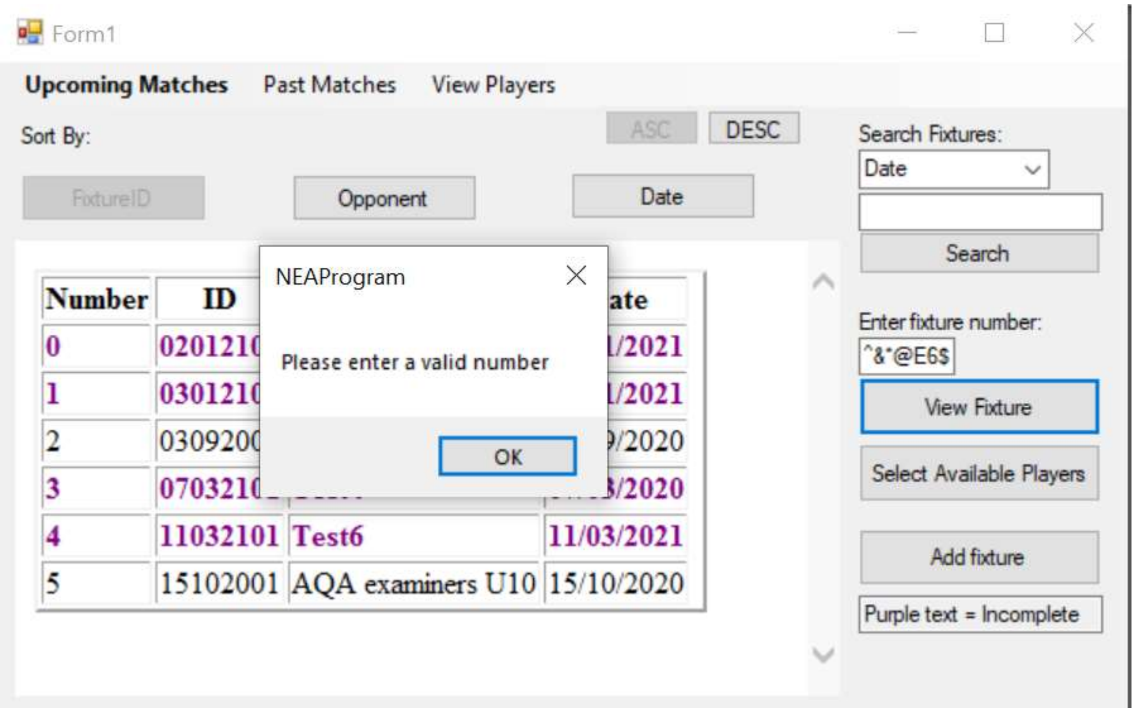
Player One
U10
Rating: 65.4169

Games played: 10
Games available: 20
% Games played: 50%

Runs Scored:	0
Wickets Taken:	0
Runs Conceded:	0

Player	Position	Rating	Games Played Ratio
Player One	Batter	65.41691	0.5
Player Two	Batter	58.63836	0.8
Player Three	Batter	56.39495	0.889
Player Four	Batter/Bow...	69.40204	0.667
Player Five	Bowler	69.43949	0.417
Player Six	Batter/Bow...	62.28592	0.667
Player Seven	Batter/Bow...	64.89811	0.933
Player Eight	Batter/Bow...	75.96548	0.533

16.2



Search Fixtures: Date

Enter fixture number: ^&*@E6$

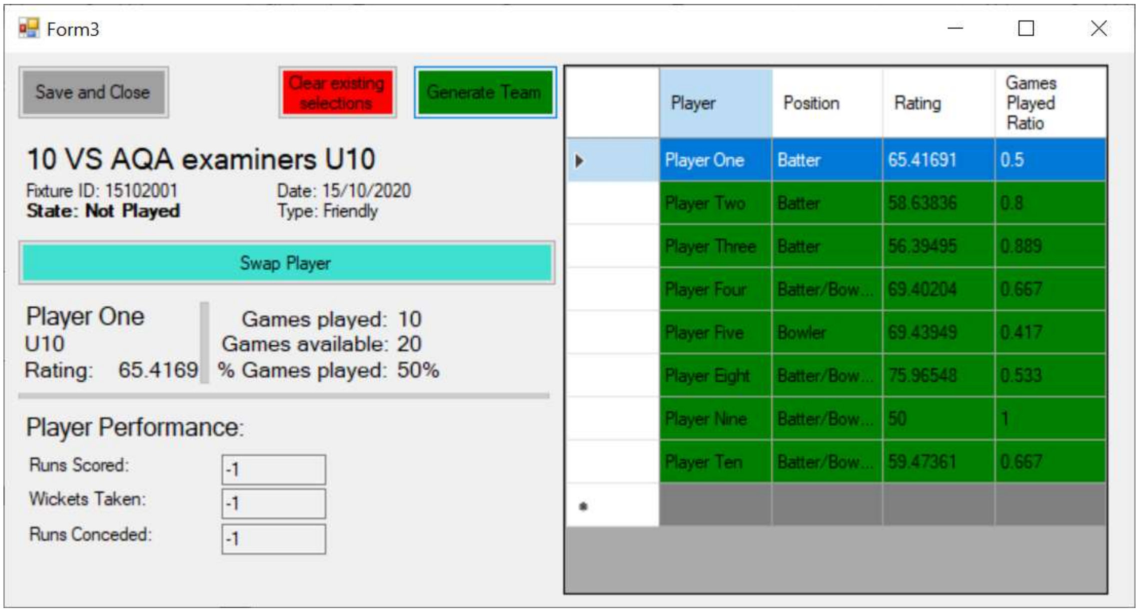
Purple text = Incomplete

Number	ID		ate
0	0201210		1/2021
1	0301210		1/2021
2	0309200		9/2020
3	0703210		3/2020
4	11032101	Test6	11/03/2021
5	15102001	AQA examiners U10	15/10/2020

NEAProgram

Please enter a valid number

16.3



10 VS AQA examiners U10

Fixture ID: 15102001
State: Not Played
Date: 15/10/2020
Type: Friendly

Player One
U10
Rating: 65.4169

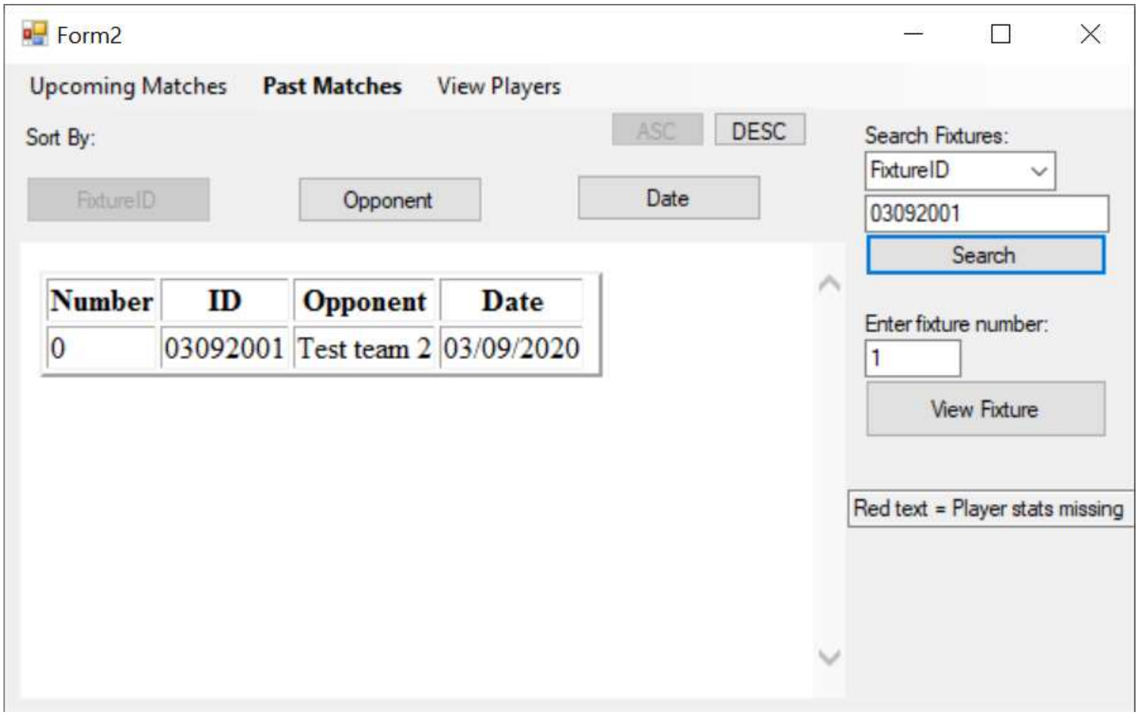
Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

Runs Scored: -1
Wickets Taken: -1
Runs Conceded: -1

Player	Position	Rating	Games Played Ratio
Player One	Batter	65.41691	0.5
Player Two	Batter	58.63836	0.8
Player Three	Batter	56.39495	0.889
Player Four	Batter/Bow...	69.40204	0.667
Player Five	Bowler	69.43949	0.417
Player Eight	Batter/Bow...	75.96548	0.533
Player Nine	Batter/Bow...	50	1
Player Ten	Batter/Bow...	59.47361	0.667

17.1



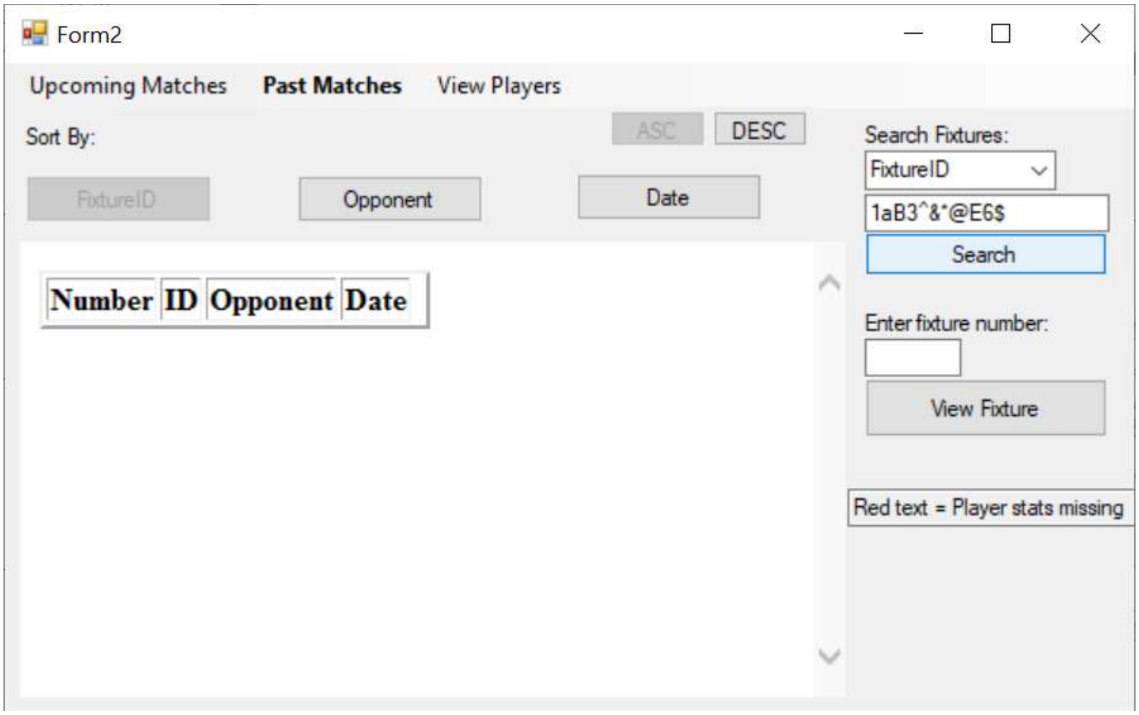
Search Fixtures: FixtureID

03092001

Enter fixture number: 1

Number	ID	Opponent	Date
0	03092001	Test team 2	03/09/2020

17.2



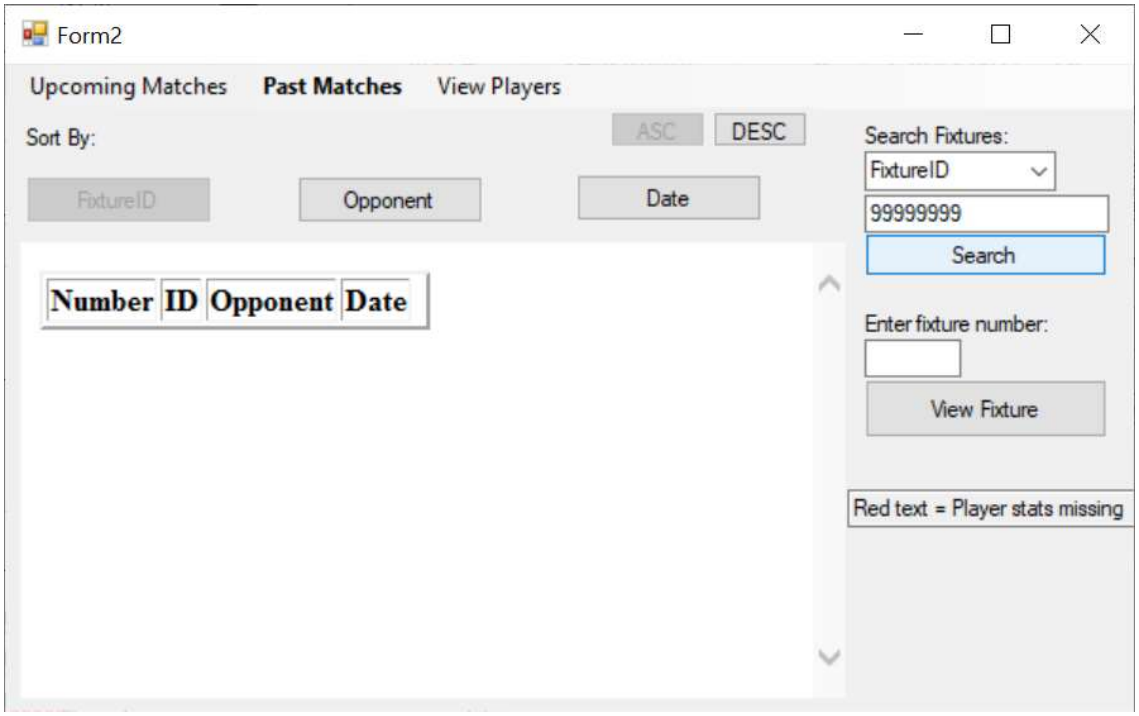
Search Fixtures: FixtureID

1aB3^&*@E6$

Enter fixture number:

Number	ID	Opponent	Date

17.3

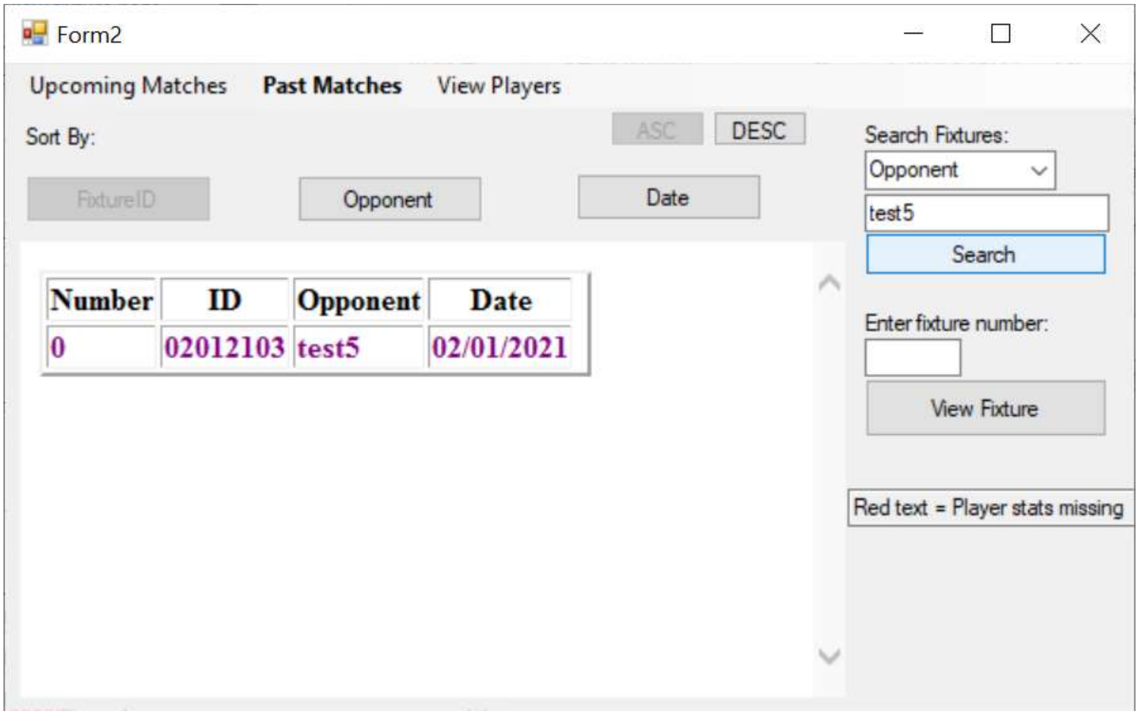


Search Fixtures: FixtureID

99999999

Number	ID	Opponent	Date

18.1

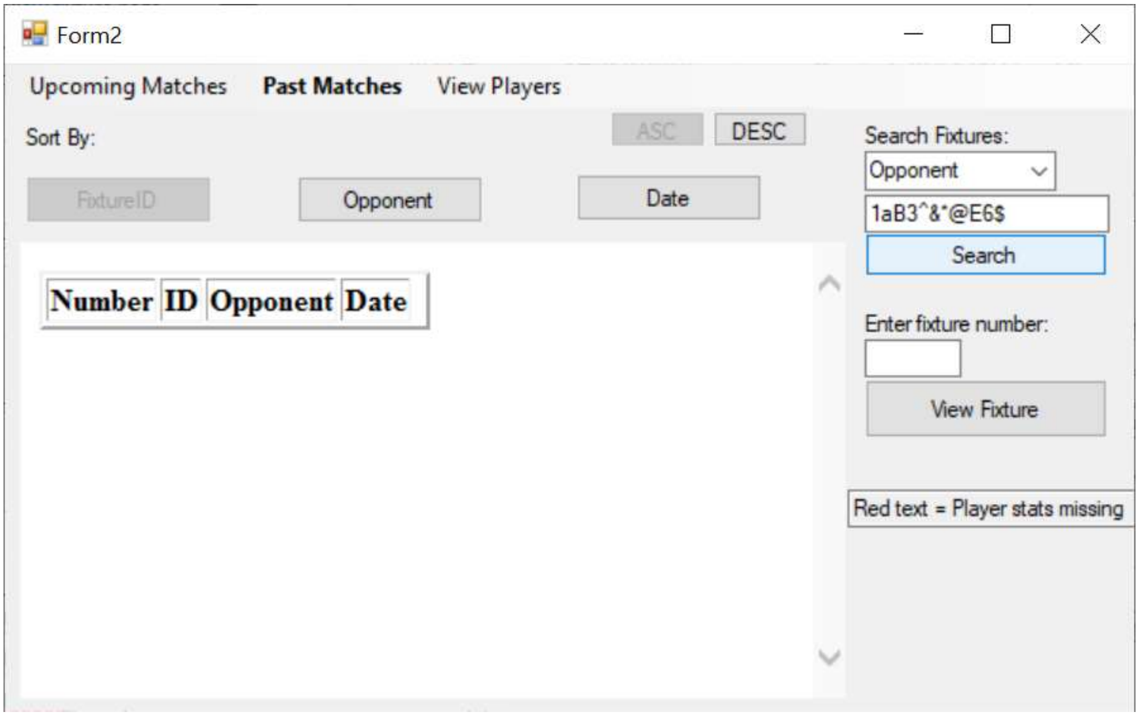


Search Fixtures: Opponent

test5

Number	ID	Opponent	Date
0	02012103	test5	02/01/2021

18.2

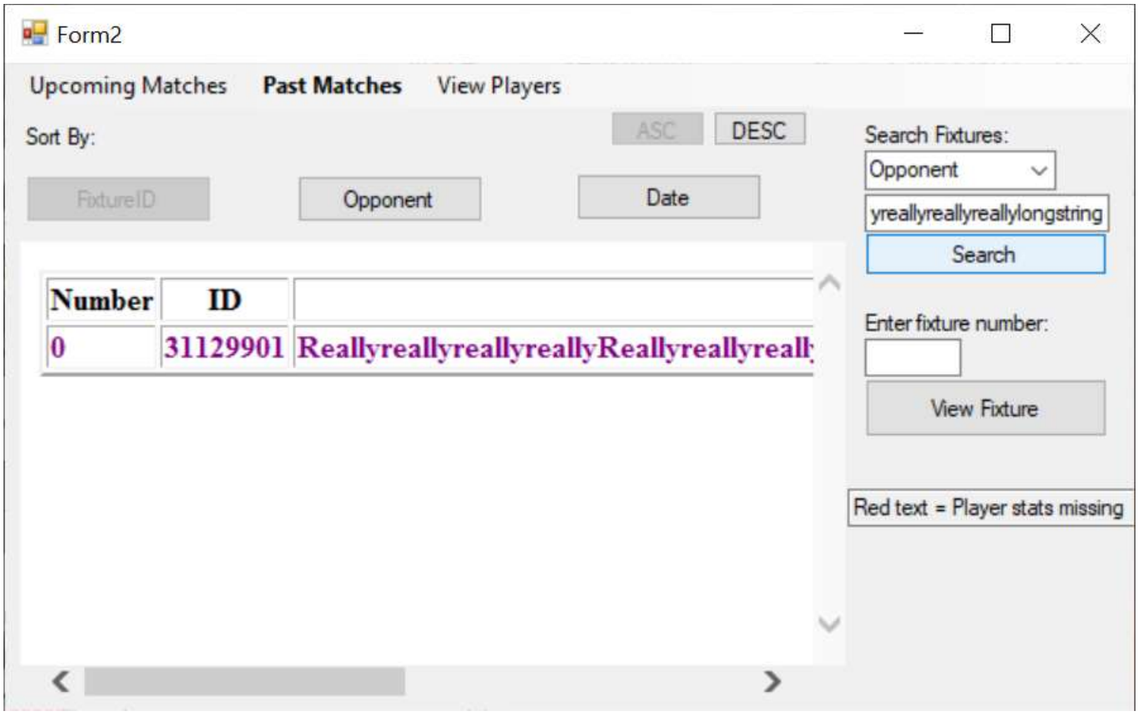


Search Fixtures: Opponent

1aB3^&*@E6$

Number	ID	Opponent	Date

18.3

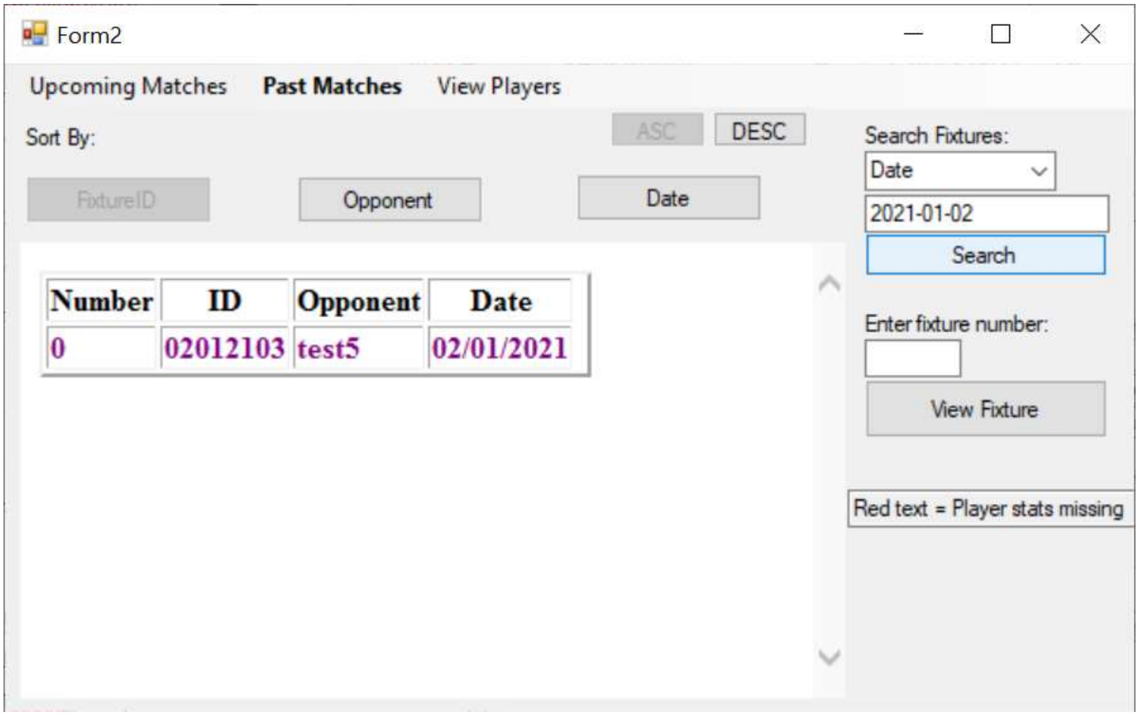


Search Fixtures: Opponent

yreallyreallyreallylongstring

Number	ID	
0	31129901	ReallyreallyreallyreallyReallyreallyreall

19.1

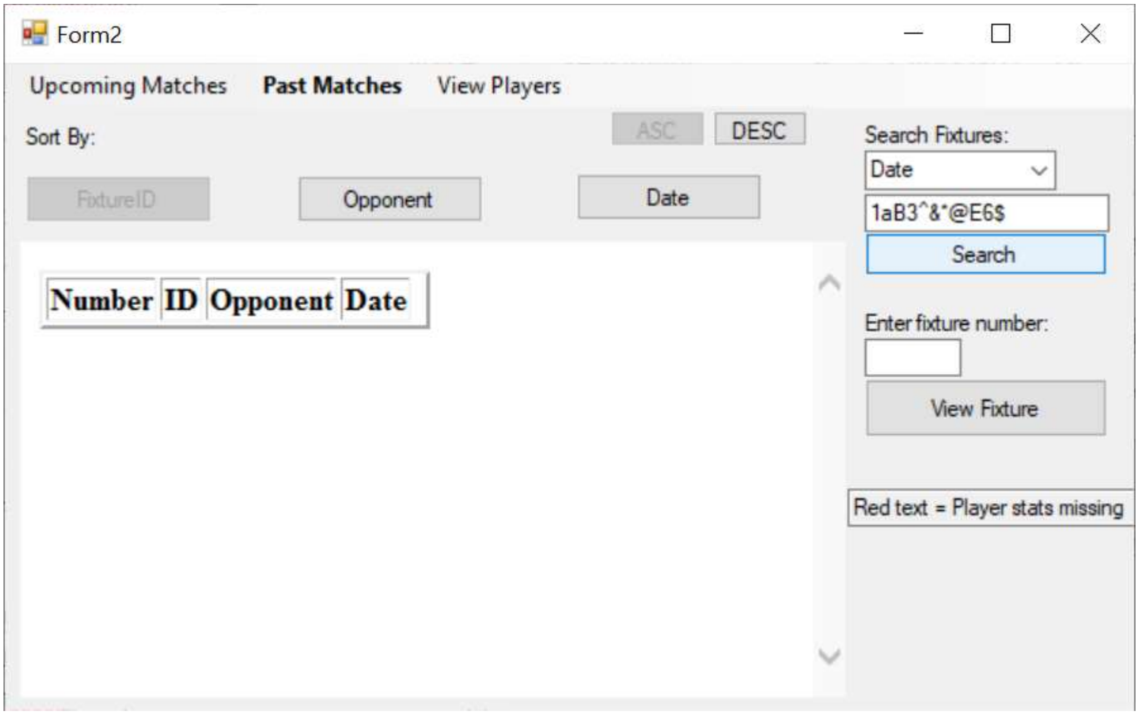


Search Fixtures: Date

2021-01-02

Number	ID	Opponent	Date
0	02012103	test5	02/01/2021

19.2

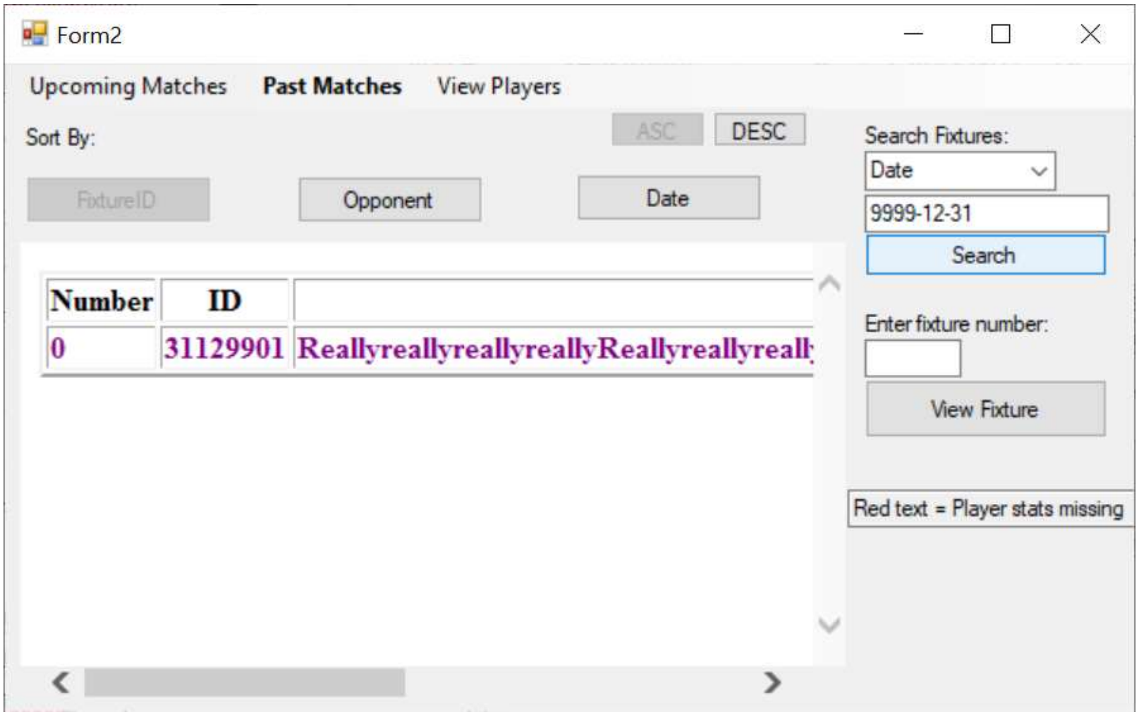


Search Fixtures: Date

1aB3^&*@E6$

Number	ID	Opponent	Date

19.3



Sort By:

Search Fixtures: Date

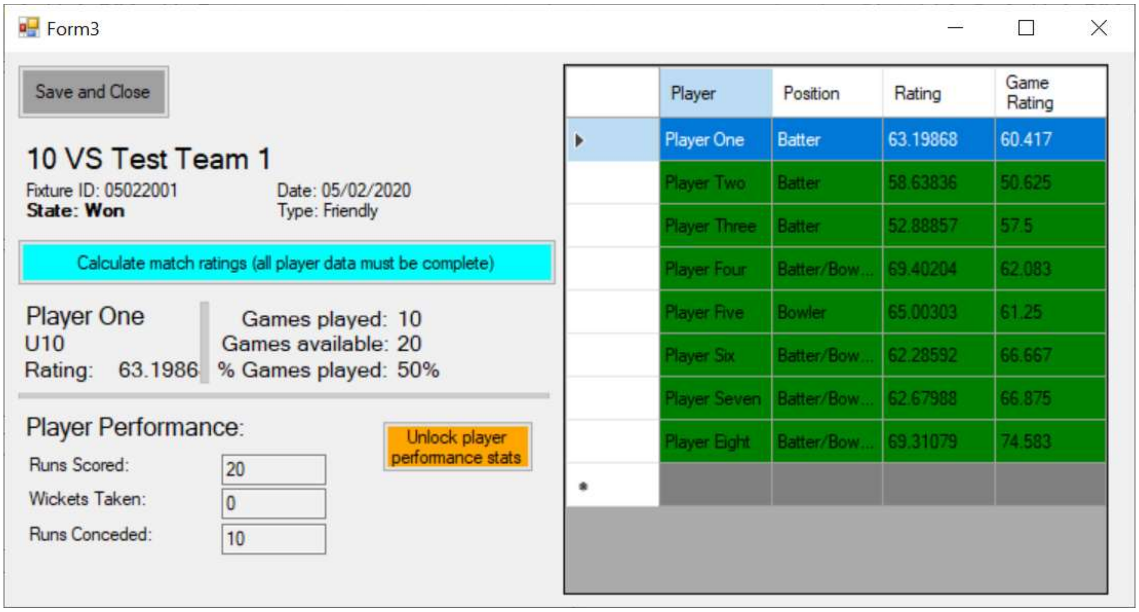
9999-12-31

Enter fixture number:

Number	ID	
0	31129901	ReallyreallyreallyreallyReallyreallyreall

Red text = Player stats missing

20.1


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 1

Fixture ID: 05022001 Date: 05/02/2020

State: Won Type: Friendly

Player One

U10

Rating: 63.1986

Games played: 10

Games available: 20

% Games played: 50%

Player Performance:

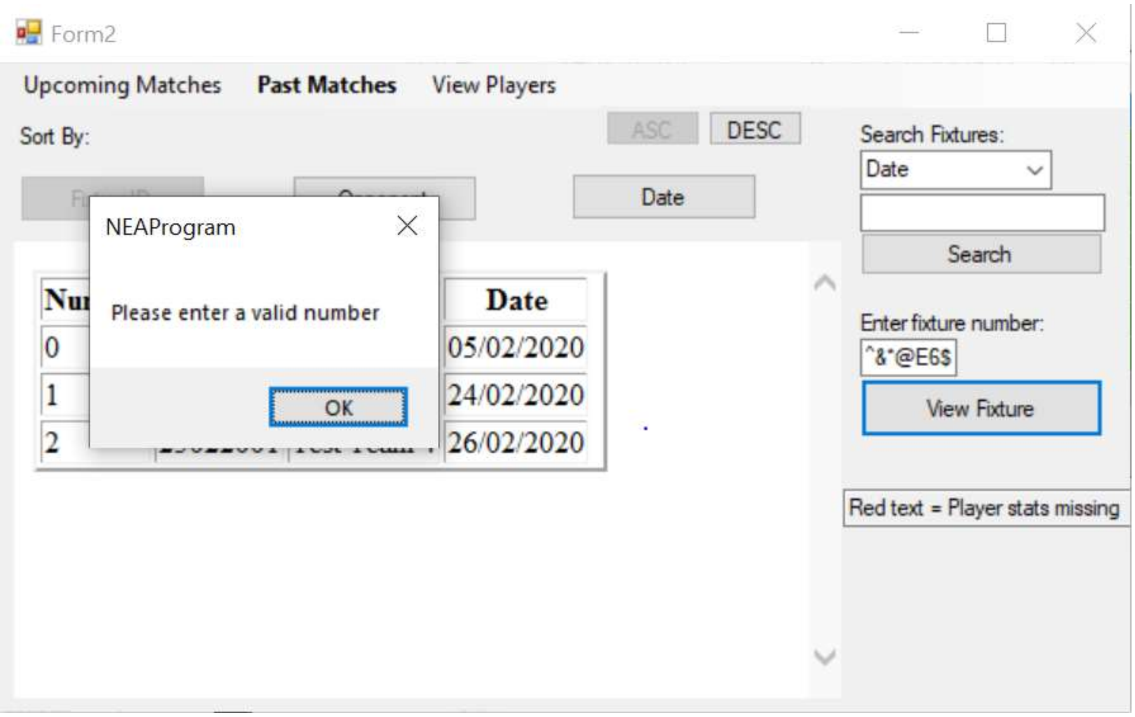
Runs Scored: 20

Wickets Taken: 0

Runs Conceded: 10

	Player	Position	Rating	Game Rating
▶	Player One	Batter	63.19868	60.417
	Player Two	Batter	58.63836	50.625
	Player Three	Batter	52.88857	57.5
	Player Four	Batter/Bow...	69.40204	62.083
	Player Five	Bowler	65.00303	61.25
	Player Six	Batter/Bow...	62.28592	66.667
	Player Seven	Batter/Bow...	62.67988	66.875
	Player Eight	Batter/Bow...	69.31079	74.583
*				

20.2



NEAProgram

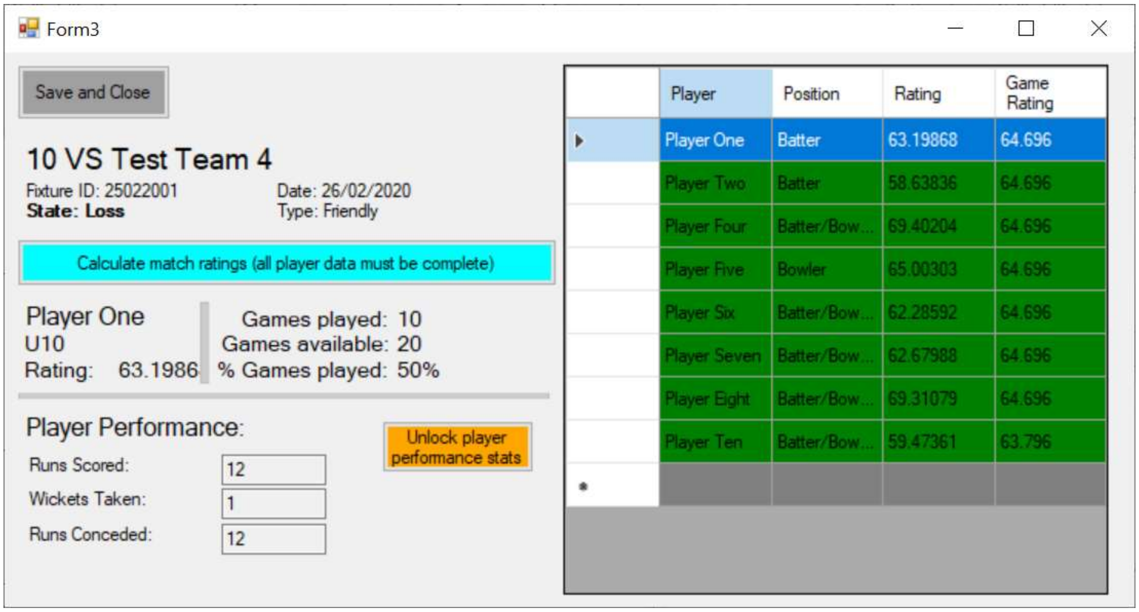
Please enter a valid number

Nu		Date
0		05/02/2020
1		24/02/2020
2		26/02/2020

Enter fixture number: ^&*@E6$

Red text = Player stats missing

20.3


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

Player One
U10
Rating: 63.1986

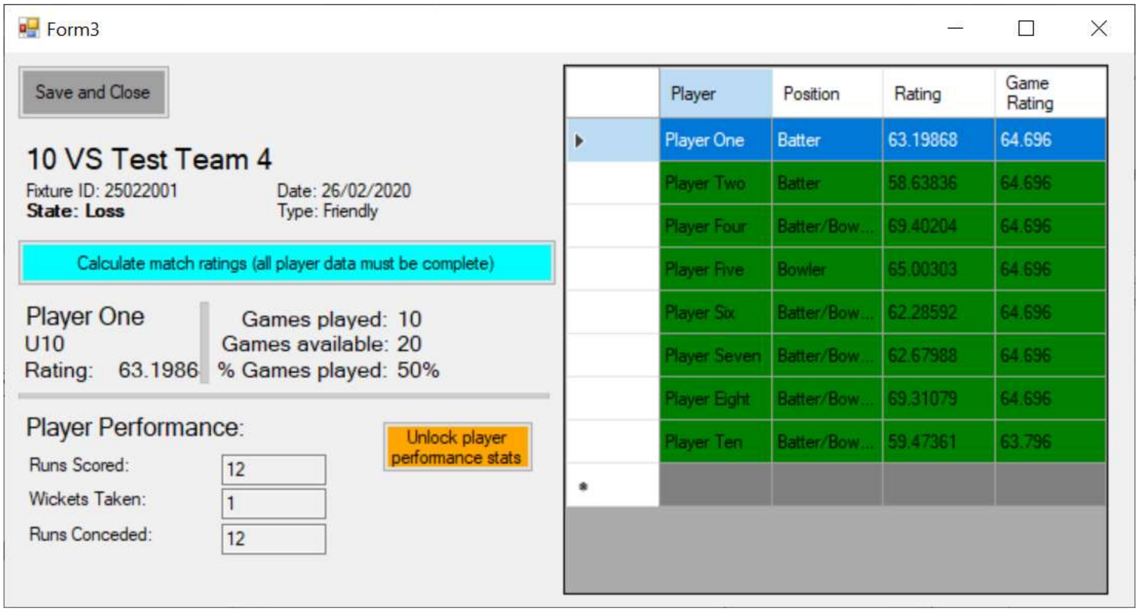
Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

Runs Scored: 12
Wickets Taken: 1
Runs Conceded: 12

	Player	Position	Rating	Game Rating
▶	Player One	Batter	63.19868	64.696
	Player Two	Batter	58.63836	64.696
	Player Four	Batter/Bow...	69.40204	64.696
	Player Five	Bowler	65.00303	64.696
	Player Six	Batter/Bow...	62.28592	64.696
	Player Seven	Batter/Bow...	62.67988	64.696
	Player Eight	Batter/Bow...	69.31079	64.696
	Player Ten	Batter/Bow...	59.47361	63.796
*				

21.1


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

Player One
U10
Rating: 63.1986

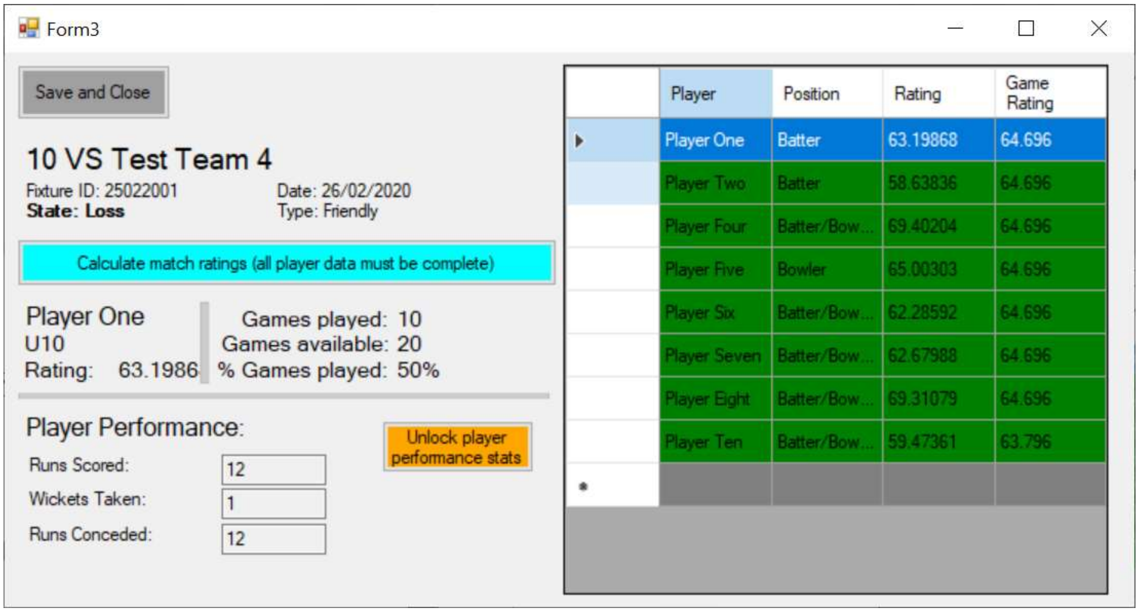
Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

Runs Scored: 12
Wickets Taken: 1
Runs Conceded: 12

	Player	Position	Rating	Game Rating
▸	Player One	Batter	63.19868	64.696
	Player Two	Batter	58.63836	64.696
	Player Four	Batter/Bow...	69.40204	64.696
	Player Five	Bowler	65.00303	64.696
	Player Six	Batter/Bow...	62.28592	64.696
	Player Seven	Batter/Bow...	62.67988	64.696
	Player Eight	Batter/Bow...	69.31079	64.696
	Player Ten	Batter/Bow...	59.47361	63.796
*				

21.2


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

Player One
U10
Rating: 63.1986

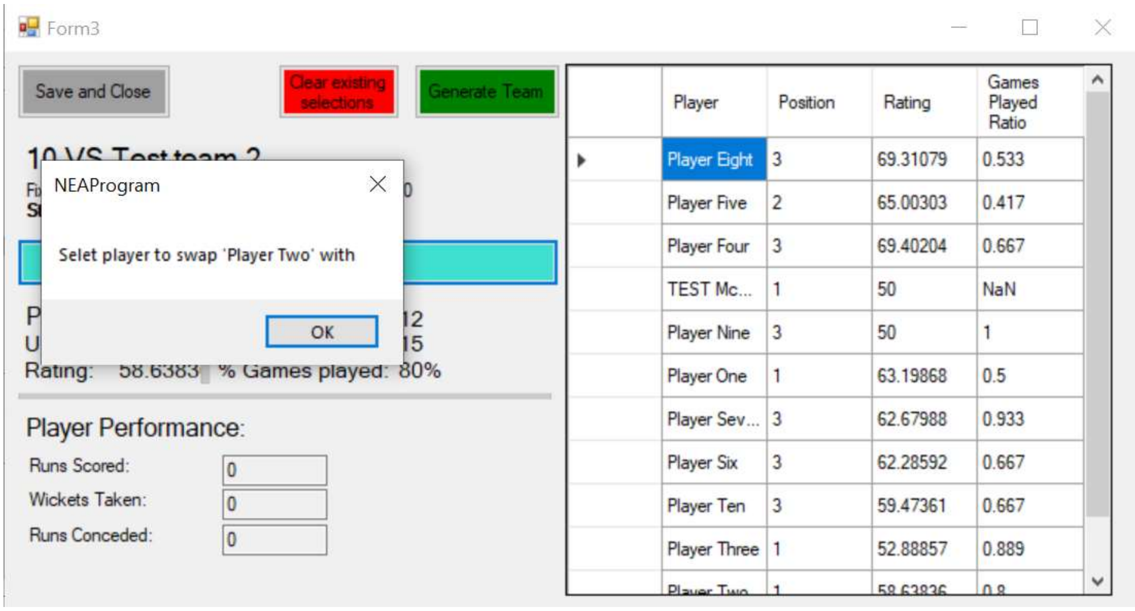
Games played: 10
Games available: 20
% Games played: 50%

Player Performance:

Runs Scored: 12
Wickets Taken: 1
Runs Conceded: 12

	Player	Position	Rating	Game Rating
▸	Player One	Batter	63.19868	64.696
	Player Two	Batter	58.63836	64.696
	Player Four	Batter/Bow...	69.40204	64.696
	Player Five	Bowler	65.00303	64.696
	Player Six	Batter/Bow...	62.28592	64.696
	Player Seven	Batter/Bow...	62.67988	64.696
	Player Eight	Batter/Bow...	69.31079	64.696
	Player Ten	Batter/Bow...	59.47361	63.796
*				

22.1



10 VS Test team 2

NEAProgram

Selet player to swap 'Player Two' with

OK

Rating: 58.6383 % Games played: 80%

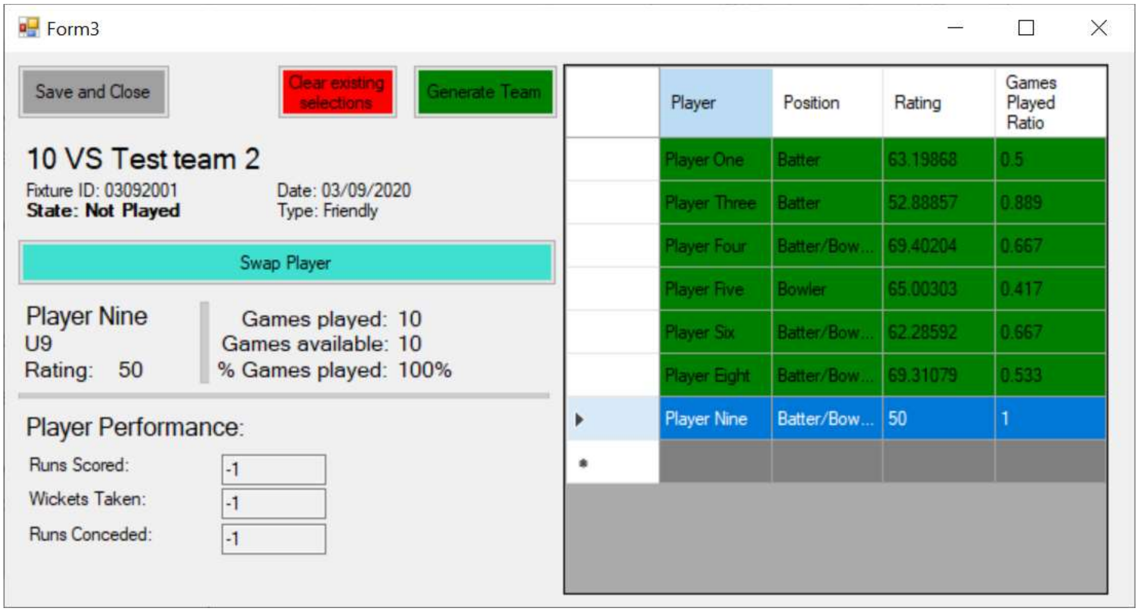
Player Performance:

Runs Scored: 0

Wickets Taken: 0

Runs Conceded: 0

Player	Position	Rating	Games Played Ratio
Player Eight	3	69.31079	0.533
Player Five	2	65.00303	0.417
Player Four	3	69.40204	0.667
TEST Mc...	1	50	NaN
Player Nine	3	50	1
Player One	1	63.19868	0.5
Player Sev...	3	62.67988	0.933
Player Six	3	62.28592	0.667
Player Ten	3	59.47361	0.667
Player Three	1	52.88857	0.889
Player Two	1	58.63836	0.8



10 VS Test team 2

Fixture ID: 03092001 Date: 03/09/2020

State: Not Played Type: Friendly

Player Nine Games played: 10

U9 Games available: 10

Rating: 50 % Games played: 100%

Player Performance:

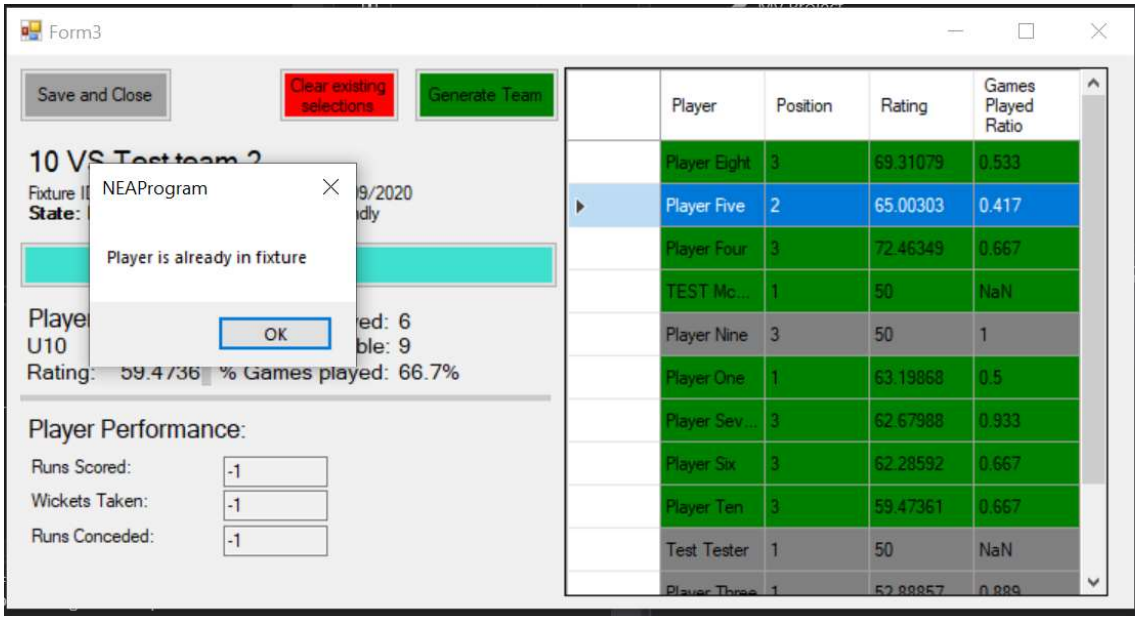
Runs Scored: -1

Wickets Taken: -1

Runs Conceded: -1

Player	Position	Rating	Games Played Ratio
Player One	Batter	63.19868	0.5
Player Three	Batter	52.88857	0.889
Player Four	Batter/Bow...	69.40204	0.667
Player Five	Bowler	65.00303	0.417
Player Six	Batter/Bow...	62.28592	0.667
Player Eight	Batter/Bow...	69.31079	0.533
Player Nine	Batter/Bow...	50	1

22.2



10 VS Test team 2

Fixture I… …9/2020

State: … …dly

NEAProgram

Player is already in fixture

OK

Playe… …ed: 6

U10 …ble: 9

Rating: 59.4736 % Games played: 66.7%

Player Performance:

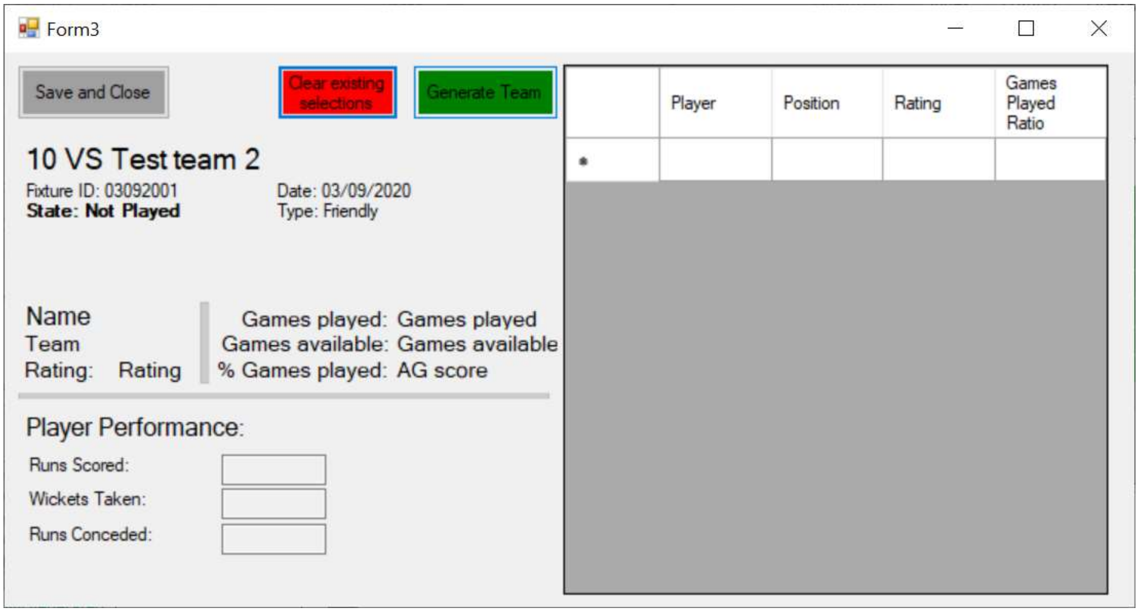
Runs Scored: -1

Wickets Taken: -1

Runs Conceded: -1

Player	Position	Rating	Games Played Ratio
Player Eight	3	69.31079	0.533
Player Five	2	65.00303	0.417
Player Four	3	72.46349	0.667
TEST Mc...	1	50	NaN
Player Nine	3	50	1
Player One	1	63.19868	0.5
Player Sev...	3	62.67988	0.933
Player Six	3	62.28592	0.667
Player Ten	3	59.47361	0.667
Test Tester	1	50	NaN
Player Three	1	52.88857	0.889

23.1



10 VS Test team 2

Fixture ID: 03092001 Date: 03/09/2020

State: Not Played Type: Friendly

Name Games played: Games played

Team Games available: Games available

Rating: Rating % Games played: AG score

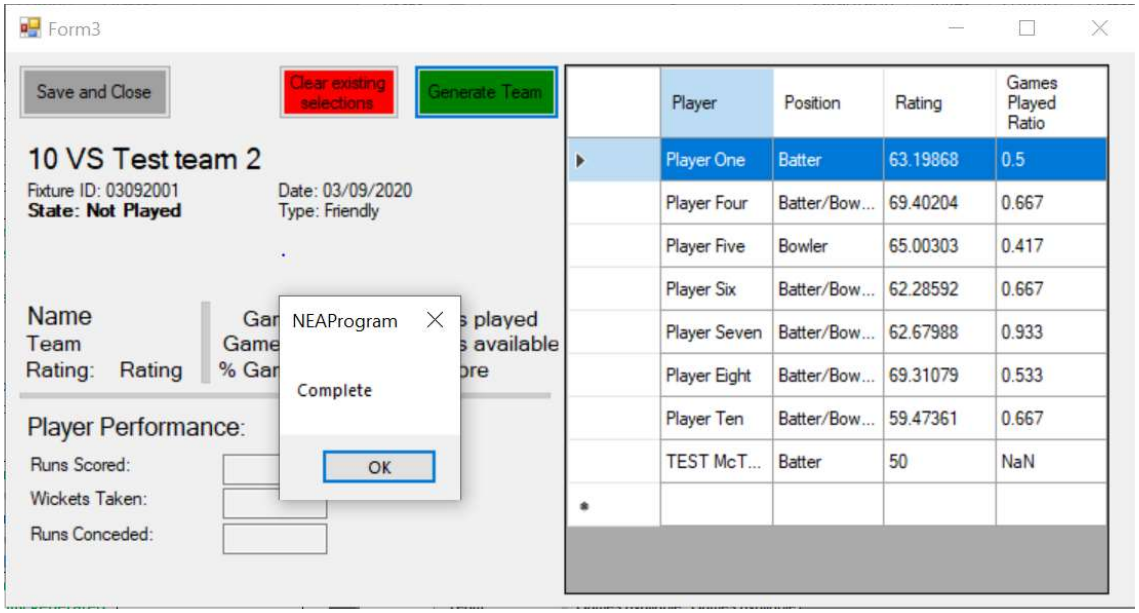
Player Performance:

Runs Scored:

Wickets Taken:

Runs Conceded:

Player	Position	Rating	Games Played Ratio



10 VS Test team 2

Fixture ID: 03092001
State: Not Played

Date: 03/09/2020
Type: Friendly

Name
Team
Rating: Rating

Gar
Game
% Gar

s played
s available
ore

Player Performance:

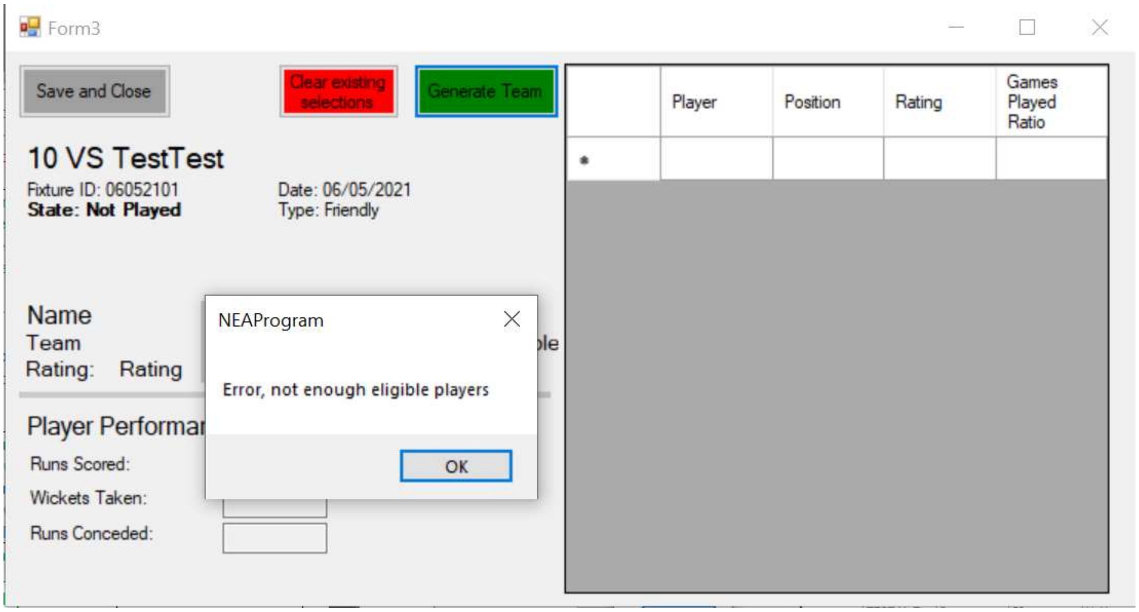
Runs Scored:
Wickets Taken:
Runs Conceded:

NEAProgram

Complete

Player	Position	Rating	Games Played Ratio
Player One	Batter	63.19868	0.5
Player Four	Batter/Bow...	69.40204	0.667
Player Five	Bowler	65.00303	0.417
Player Six	Batter/Bow...	62.28592	0.667
Player Seven	Batter/Bow...	62.67988	0.933
Player Eight	Batter/Bow...	69.31079	0.533
Player Ten	Batter/Bow...	59.47361	0.667
TEST McT...	Batter	50	NaN

23.2



10 VS TestTest

Fixture ID: 06052101
State: Not Played

Date: 06/05/2021
Type: Friendly

Name
Team
Rating: Rating

ple

Player Performar

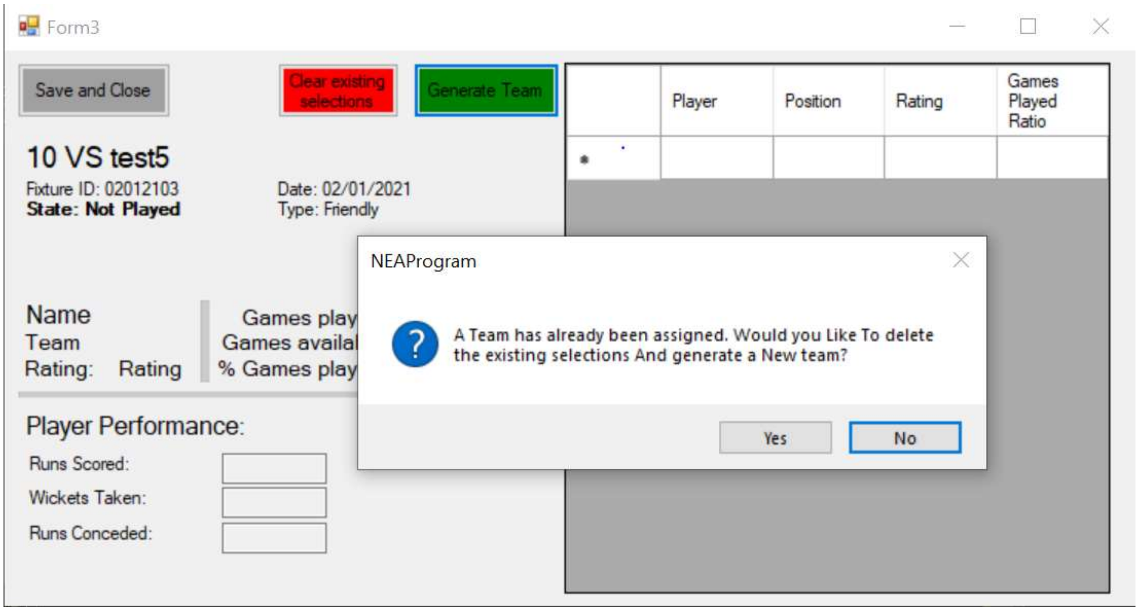
Runs Scored:
Wickets Taken:
Runs Conceded:

NEAProgram

Error, not enough eligible players

Player	Position	Rating	Games Played Ratio

23.3



10 VS test5

Fixture ID: 02012103

State: Not Played

Date: 02/01/2021

Type: Friendly

Name

Team

Rating: Rating

Games play

Games availa

% Games play

Player Performance:

Runs Scored:

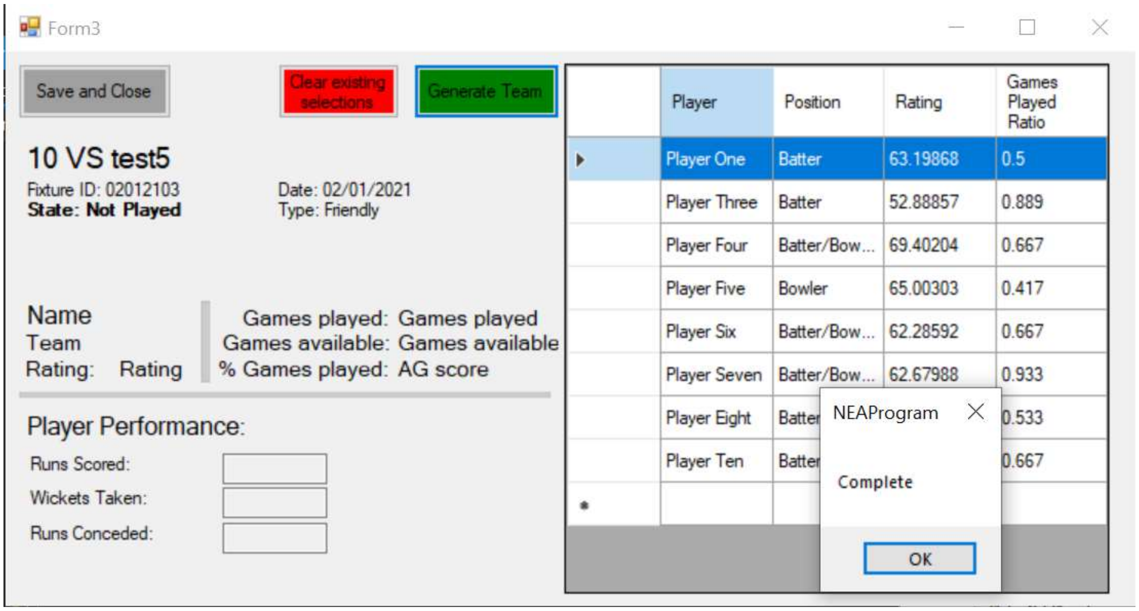
Wickets Taken:

Runs Conceded:

	Player	Position	Rating	Games Played Ratio
*				

NEAProgram

A Team has already been assigned. Would you Like To delete the existing selections And generate a New team?



10 VS test5

Fixture ID: 02012103

State: Not Played

Date: 02/01/2021

Type: Friendly

Name

Team

Rating: Rating

Games played: Games played

Games available: Games available

% Games played: AG score

Player Performance:

Runs Scored:

Wickets Taken:

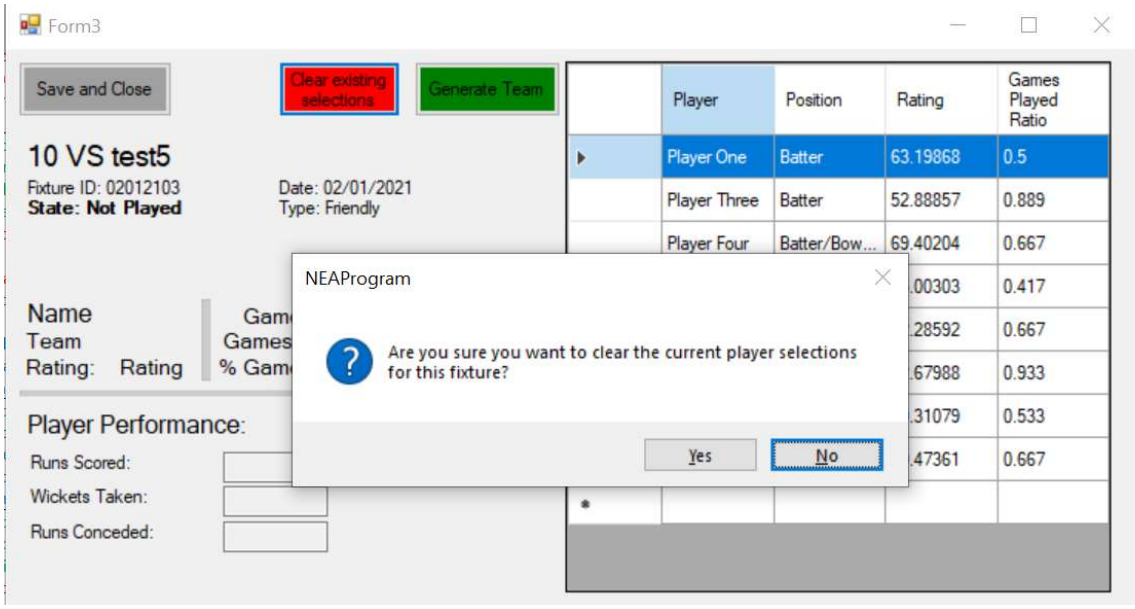
Runs Conceded:

	Player	Position	Rating	Games Played Ratio
▶	Player One	Batter	63.19868	0.5
	Player Three	Batter	52.88857	0.889
	Player Four	Batter/Bow...	69.40204	0.667
	Player Five	Bowler	65.00303	0.417
	Player Six	Batter/Bow...	62.28592	0.667
	Player Seven	Batter/Bow...	62.67988	0.933
	Player Eight	Batter		0.533
	Player Ten	Batter		0.667
*				

NEAProgram

Complete

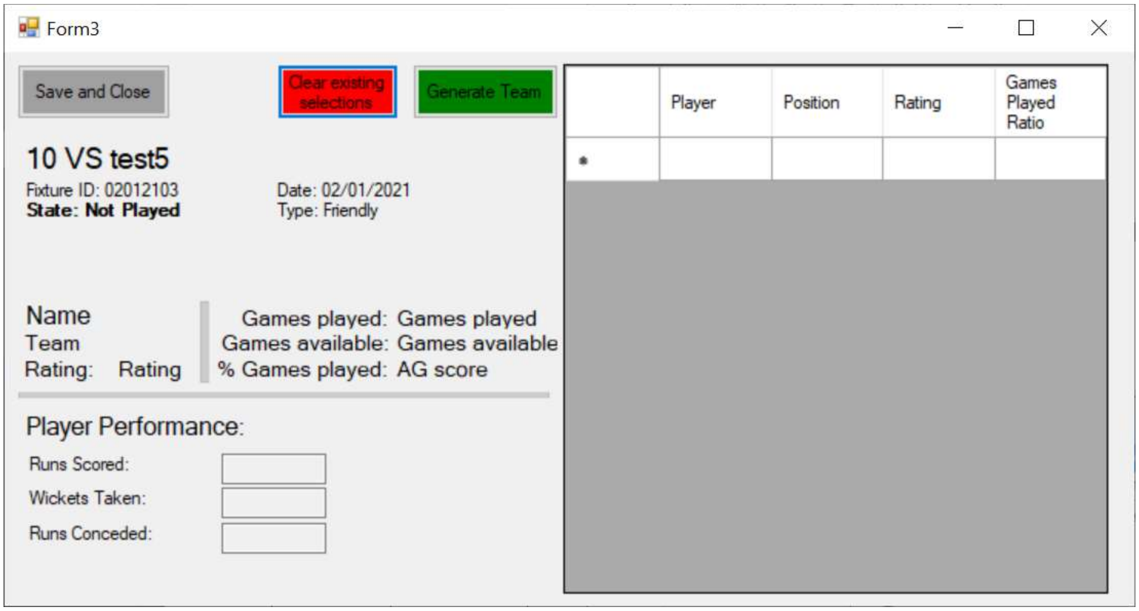
24.1



10 VS test5
Fixture ID: 02012103
State: Not Played
Date: 02/01/2021
Type: Friendly

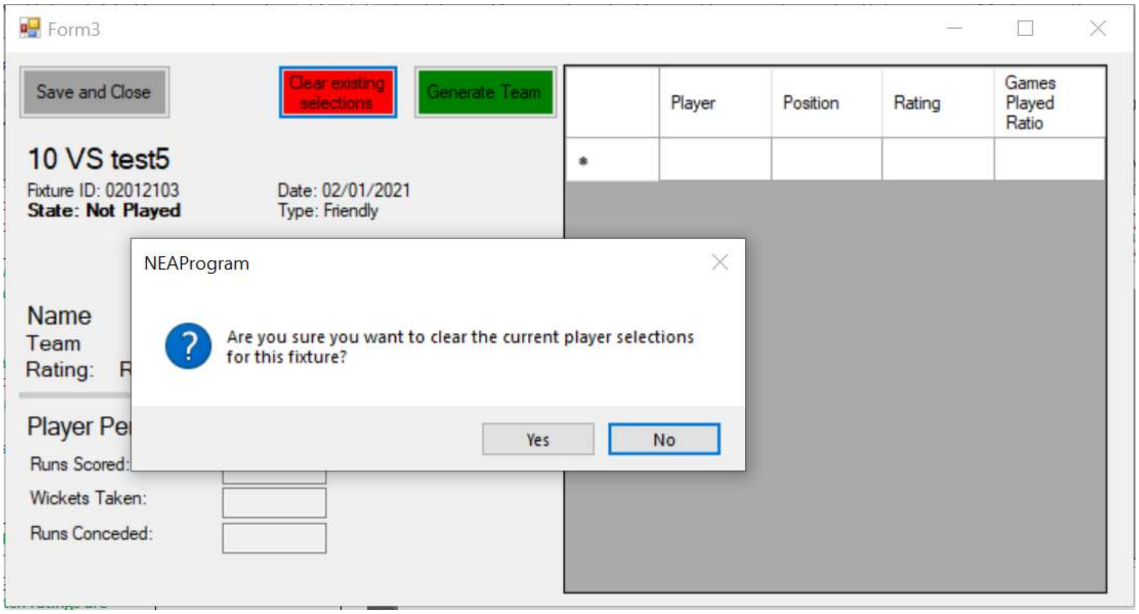
Player	Position	Rating	Games Played Ratio
Player One	Batter	63.19868	0.5
Player Three	Batter	52.88857	0.889
Player Four	Batter/Bow...	69.40204	0.667
		.00303	0.417
		.28592	0.667
		.67988	0.933
		.31079	0.533
		.47361	0.667

Are you sure you want to clear the current player selections for this fixture?



10 VS test5
Fixture ID: 02012103
State: Not Played
Date: 02/01/2021
Type: Friendly

24.2



10 VS test5

Fixture ID: 02012103

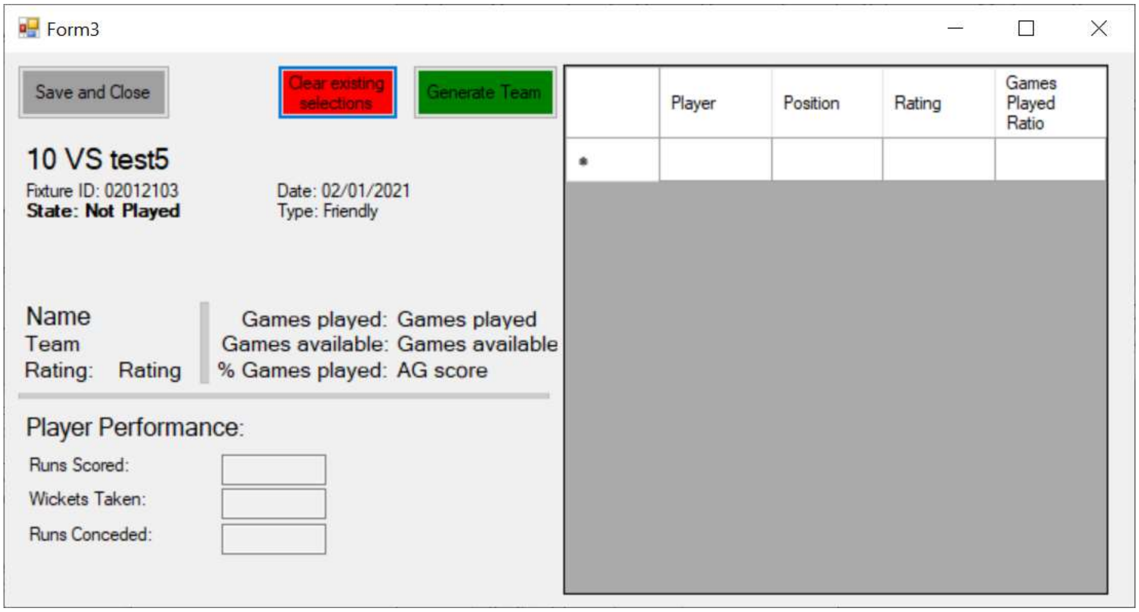
State: Not Played

Date: 02/01/2021

Type: Friendly

NEAProgram

Are you sure you want to clear the current player selections for this fixture?



10 VS test5

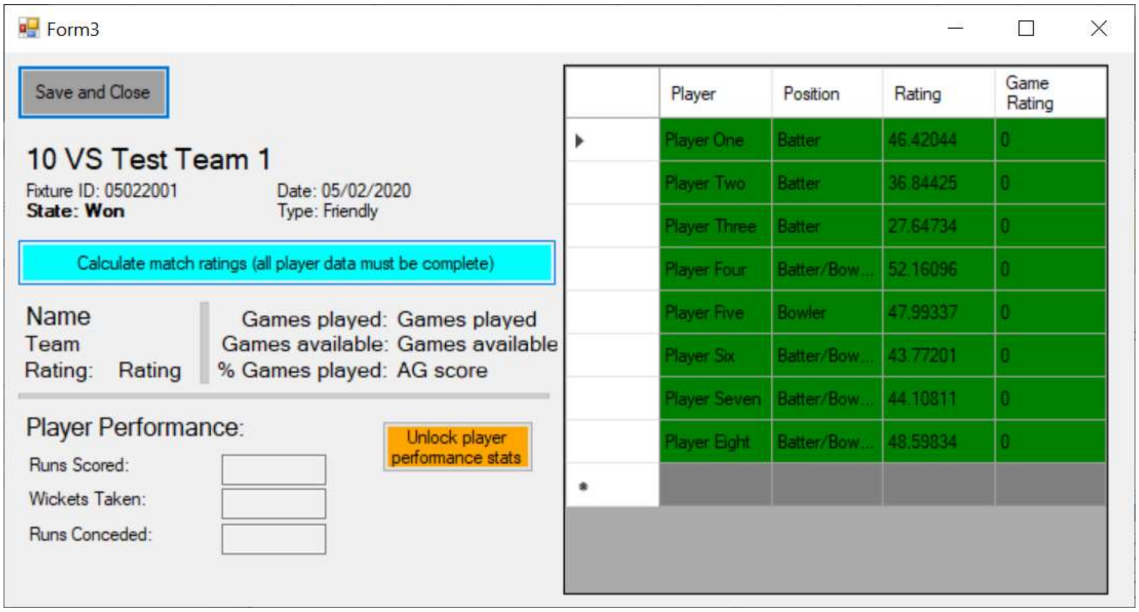
Fixture ID: 02012103

State: Not Played

Date: 02/01/2021

Type: Friendly

25.1


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 1

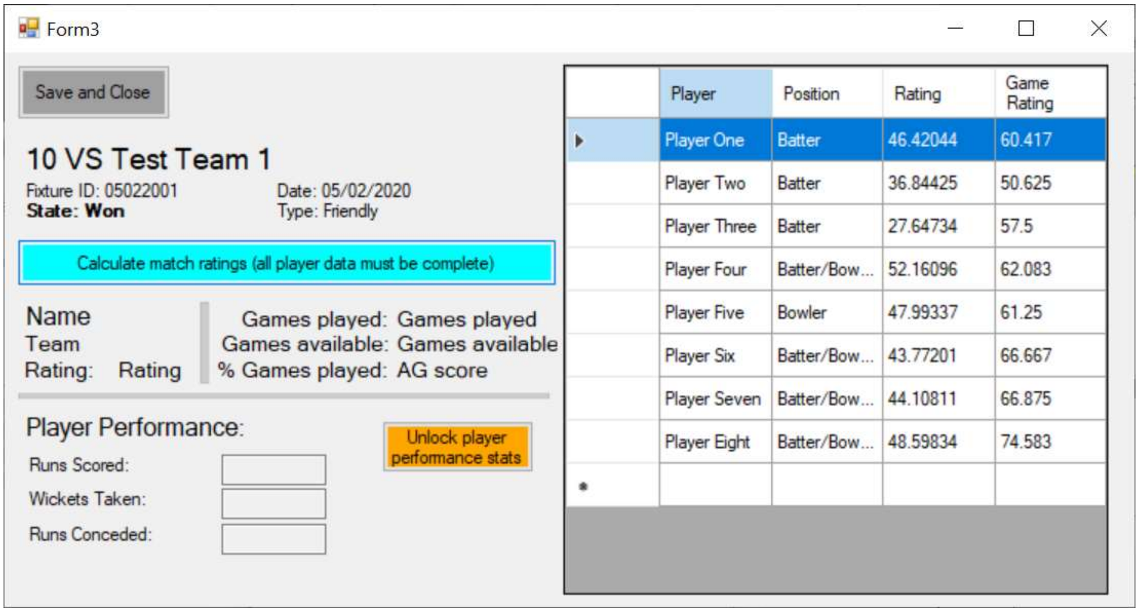
Fixture ID: 05022001 Date: 05/02/2020
State: Won Type: Friendly

Name Games played: Games played
Team Games available: Games available
Rating: Rating % Games played: AG score

Player Performance:

Runs Scored:
Wickets Taken:
Runs Conceded:

	Player	Position	Rating	Game Rating
▶	Player One	Batter	46.42044	0
	Player Two	Batter	36.84425	0
	Player Three	Batter	27.64734	0
	Player Four	Batter/Bow...	52.16096	0
	Player Five	Bowler	47.99337	0
	Player Six	Batter/Bow...	43.77201	0
	Player Seven	Batter/Bow...	44.10811	0
	Player Eight	Batter/Bow...	48.59834	0
*				


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 1

Fixture ID: 05022001 Date: 05/02/2020
State: Won Type: Friendly

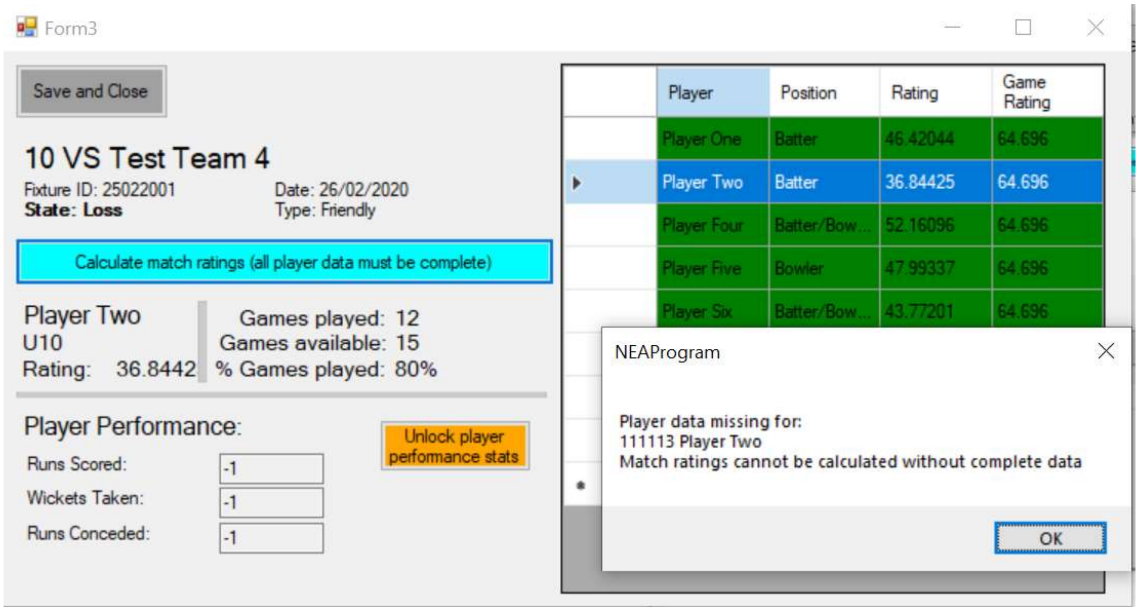
Name Games played: Games played
Team Games available: Games available
Rating: Rating % Games played: AG score

Player Performance:

Runs Scored:
Wickets Taken:
Runs Conceded:

	Player	Position	Rating	Game Rating
▶	Player One	Batter	46.42044	60.417
	Player Two	Batter	36.84425	50.625
	Player Three	Batter	27.64734	57.5
	Player Four	Batter/Bow...	52.16096	62.083
	Player Five	Bowler	47.99337	61.25
	Player Six	Batter/Bow...	43.77201	66.667
	Player Seven	Batter/Bow...	44.10811	66.875
	Player Eight	Batter/Bow...	48.59834	74.583
*				

25.2


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats
OK

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

Player Two
U10
Rating: 36.8442

Games played: 12
Games available: 15
% Games played: 80%

Player Performance:

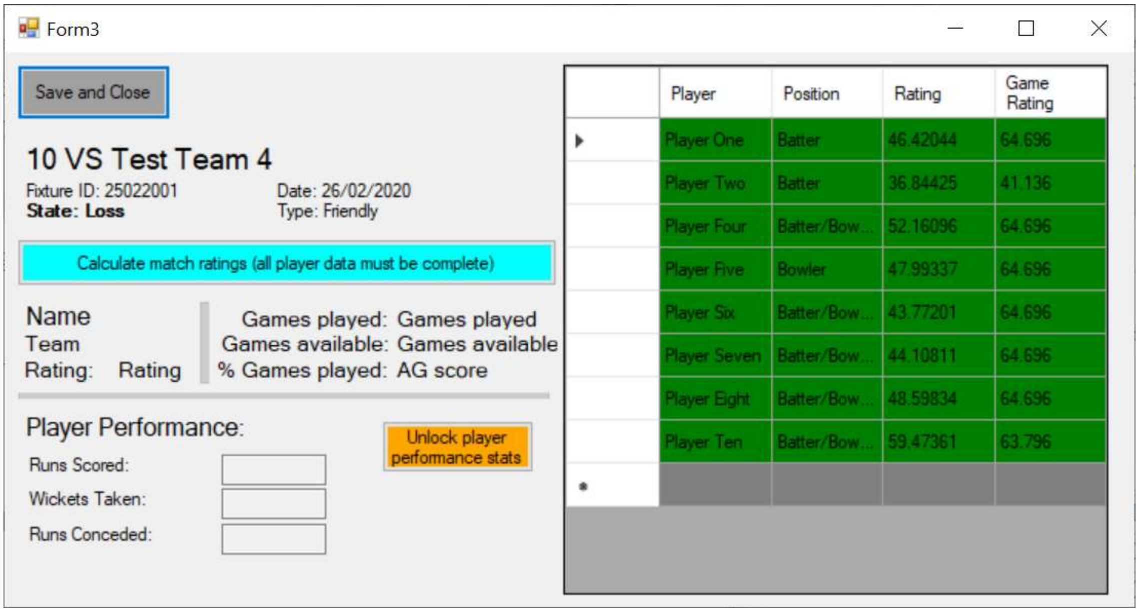
Runs Scored: -1
Wickets Taken: -1
Runs Conceded: -1

Player	Position	Rating	Game Rating
Player One	Batter	46.42044	64.696
Player Two	Batter	36.84425	64.696
Player Four	Batter/Bow...	52.16096	64.696
Player Five	Bowler	47.99337	64.696
Player Six	Batter/Bow...	43.77201	64.696

NEAProgram

Player data missing for:
111113 Player Two
Match ratings cannot be calculated without complete data

25.3


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

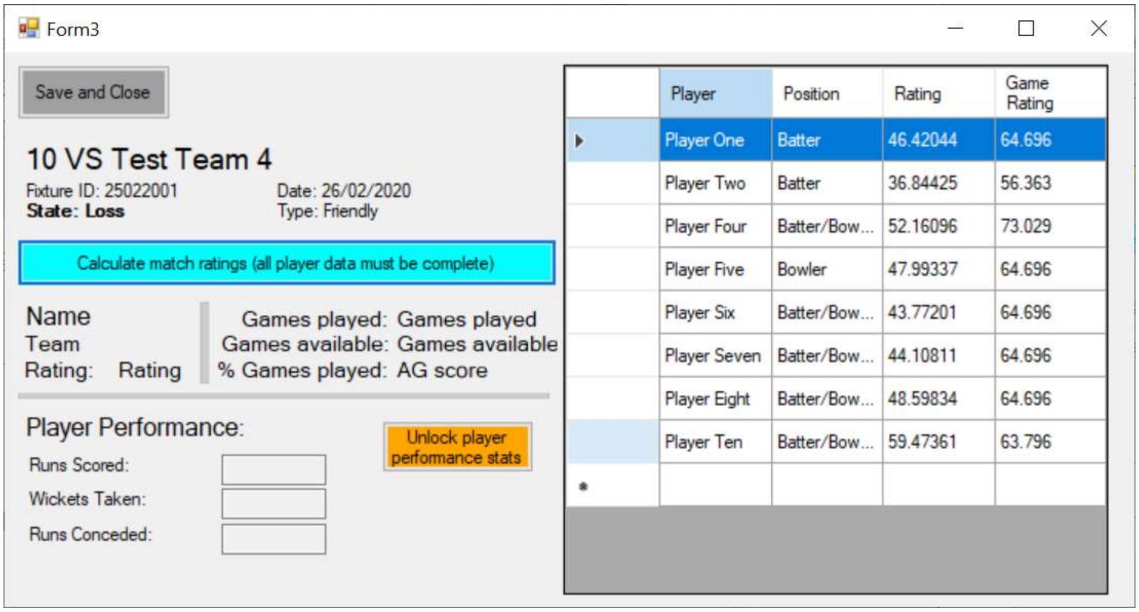
Name
Team
Rating: Rating

Games played: Games played
Games available: Games available
% Games played: AG score

Player Performance:

Runs Scored:
Wickets Taken:
Runs Conceded:

Player	Position	Rating	Game Rating
Player One	Batter	46.42044	64.696
Player Two	Batter	36.84425	41.136
Player Four	Batter/Bow...	52.16096	64.696
Player Five	Bowler	47.99337	64.696
Player Six	Batter/Bow...	43.77201	64.696
Player Seven	Batter/Bow...	44.10811	64.696
Player Eight	Batter/Bow...	48.59834	64.696
Player Ten	Batter/Bow...	59.47361	63.796


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

Name
Team
Rating: Rating

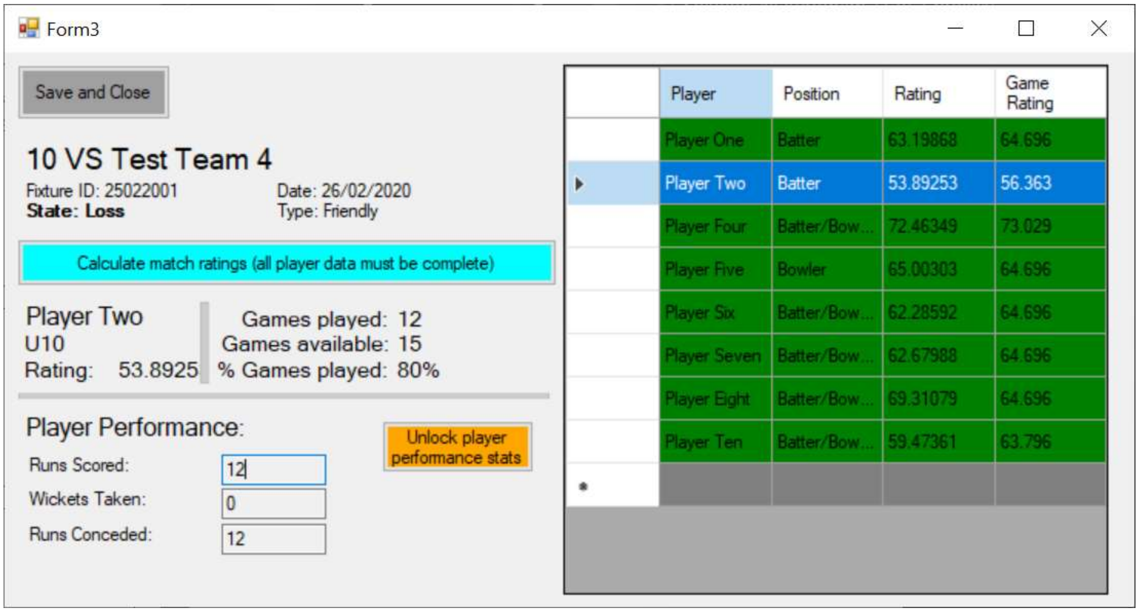
Games played: Games played
Games available: Games available
% Games played: AG score

Player Performance:

Runs Scored:
Wickets Taken:
Runs Conceded:

Player	Position	Rating	Game Rating
Player One	Batter	46.42044	64.696
Player Two	Batter	36.84425	56.363
Player Four	Batter/Bow...	52.16096	73.029
Player Five	Bowler	47.99337	64.696
Player Six	Batter/Bow...	43.77201	64.696
Player Seven	Batter/Bow...	44.10811	64.696
Player Eight	Batter/Bow...	48.59834	64.696
Player Ten	Batter/Bow...	59.47361	63.796

26.1


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

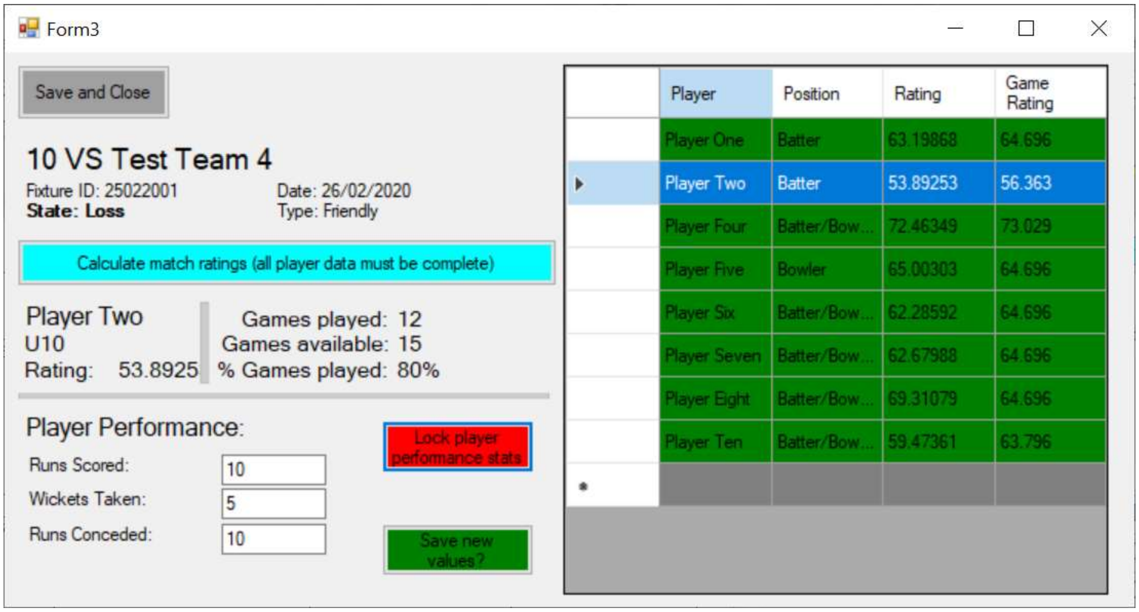
Player Two
U10
Rating: 53.8925

Games played: 12
Games available: 15
% Games played: 80%

Player Performance:

Runs Scored: 12
Wickets Taken: 0
Runs Conceded: 12

Player	Position	Rating	Game Rating
Player One	Batter	63.19868	64.696
Player Two	Batter	53.89253	56.363
Player Four	Batter/Bow...	72.46349	73.029
Player Five	Bowler	65.00303	64.696
Player Six	Batter/Bow...	62.28592	64.696
Player Seven	Batter/Bow...	62.67988	64.696
Player Eight	Batter/Bow...	69.31079	64.696
Player Ten	Batter/Bow...	59.47361	63.796


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Lock player performance stats
Save new values?

10 VS Test Team 4

Fixture ID: 25022001
State: Loss
Date: 26/02/2020
Type: Friendly

Player Two
U10
Rating: 53.8925

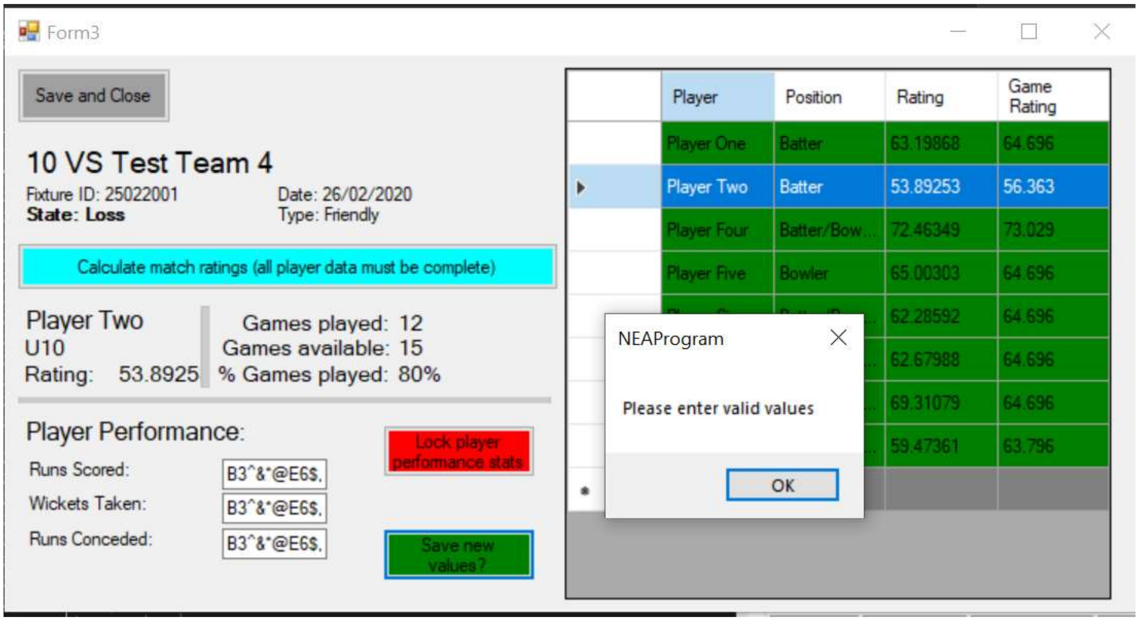
Games played: 12
Games available: 15
% Games played: 80%

Player Performance:

Runs Scored: 10
Wickets Taken: 5
Runs Conceded: 10

Player	Position	Rating	Game Rating
Player One	Batter	63.19868	64.696
Player Two	Batter	53.89253	56.363
Player Four	Batter/Bow...	72.46349	73.029
Player Five	Bowler	65.00303	64.696
Player Six	Batter/Bow...	62.28592	64.696
Player Seven	Batter/Bow...	62.67988	64.696
Player Eight	Batter/Bow...	69.31079	64.696
Player Ten	Batter/Bow...	59.47361	63.796

26.2


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Lock player performance stats
Save new values?
NEAProgram
OK

10 VS Test Team 4

Fixture ID: 25022001
State: Loss
Date: 26/02/2020
Type: Friendly

Player Two
U10
Rating: 53.8925

Games played: 12
Games available: 15
% Games played: 80%

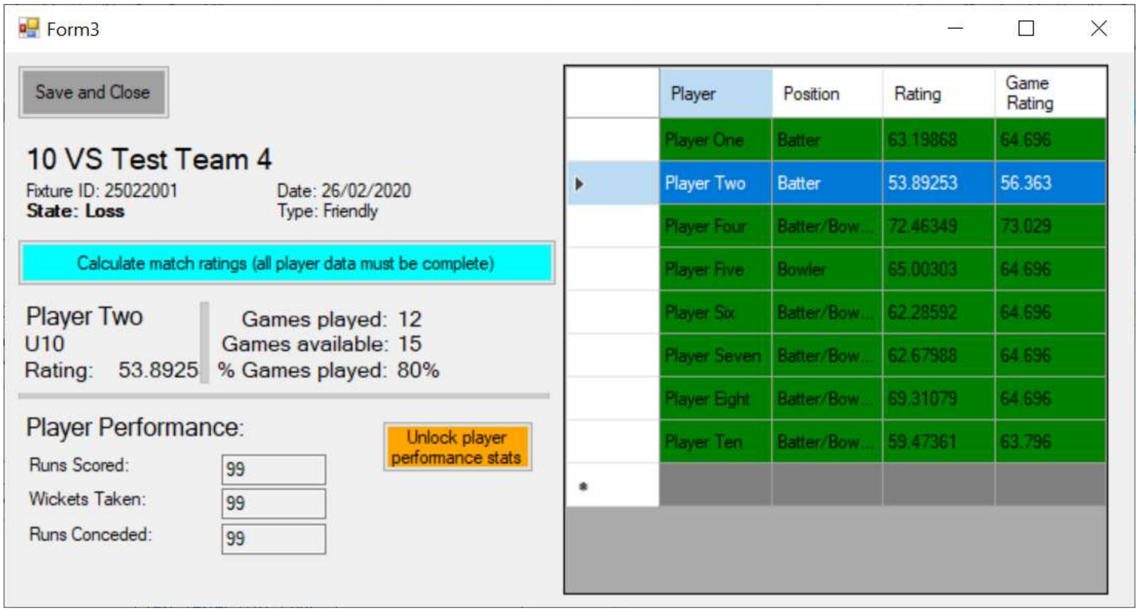
Player Performance:

Runs Scored: B3^&*@E6$,
Wickets Taken: B3^&*@E6$,
Runs Conceded: B3^&*@E6$,

Please enter valid values

Player	Position	Rating	Game Rating
Player One	Batter	63.19868	64.696
Player Two	Batter	53.89253	56.363
Player Four	Batter/Bow...	72.46349	73.029
Player Five	Bowler	65.00303	64.696
[illegible]	[illegible]	62.28592	64.696
[illegible]	[illegible]	62.67988	64.696
[illegible]	[illegible]	69.31079	64.696
[illegible]	[illegible]	59.47361	63.796

26.3


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 4

Fixture ID: 25022001 Date: 26/02/2020
State: Loss Type: Friendly

Player Two
U10
Rating: 53.8925

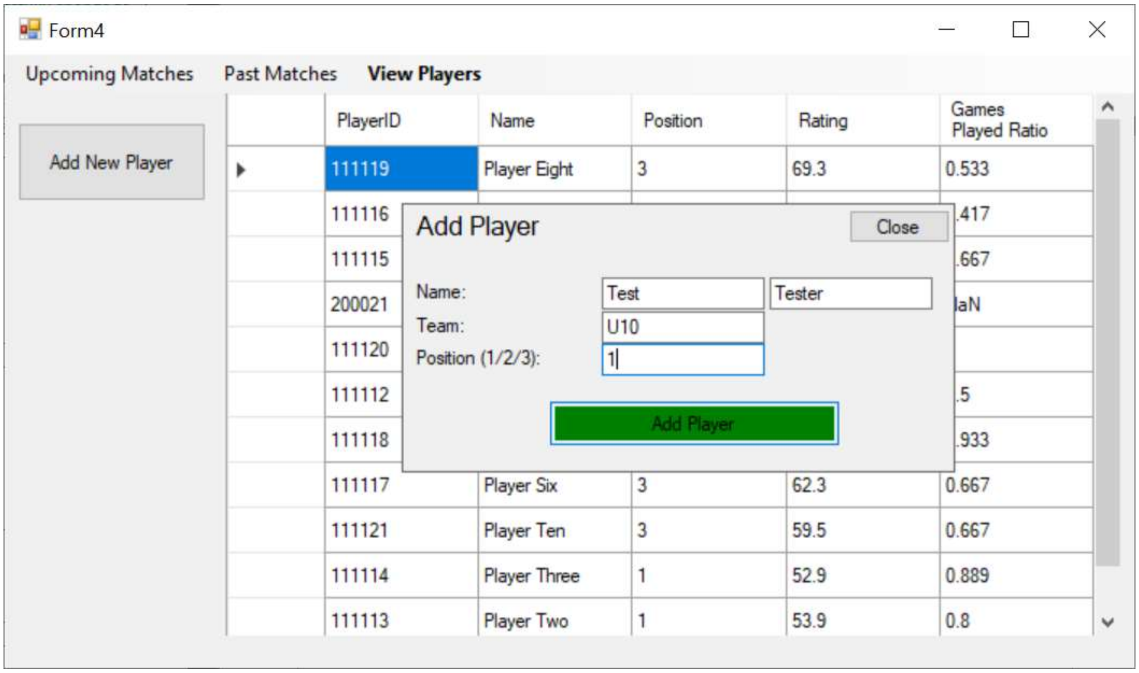
Games played: 12
Games available: 15
% Games played: 80%

Player Performance:

Runs Scored: 99
Wickets Taken: 99
Runs Conceded: 99

Player	Position	Rating	Game Rating
Player One	Batter	63.19868	64.696
Player Two	Batter	53.89253	56.363
Player Four	Batter/Bow...	72.46349	73.029
Player Five	Bowler	65.00303	64.696
Player Six	Batter/Bow...	62.28592	64.696
Player Seven	Batter/Bow...	62.67988	64.696
Player Eight	Batter/Bow...	69.31079	64.696
Player Ten	Batter/Bow...	59.47361	63.796

27.1

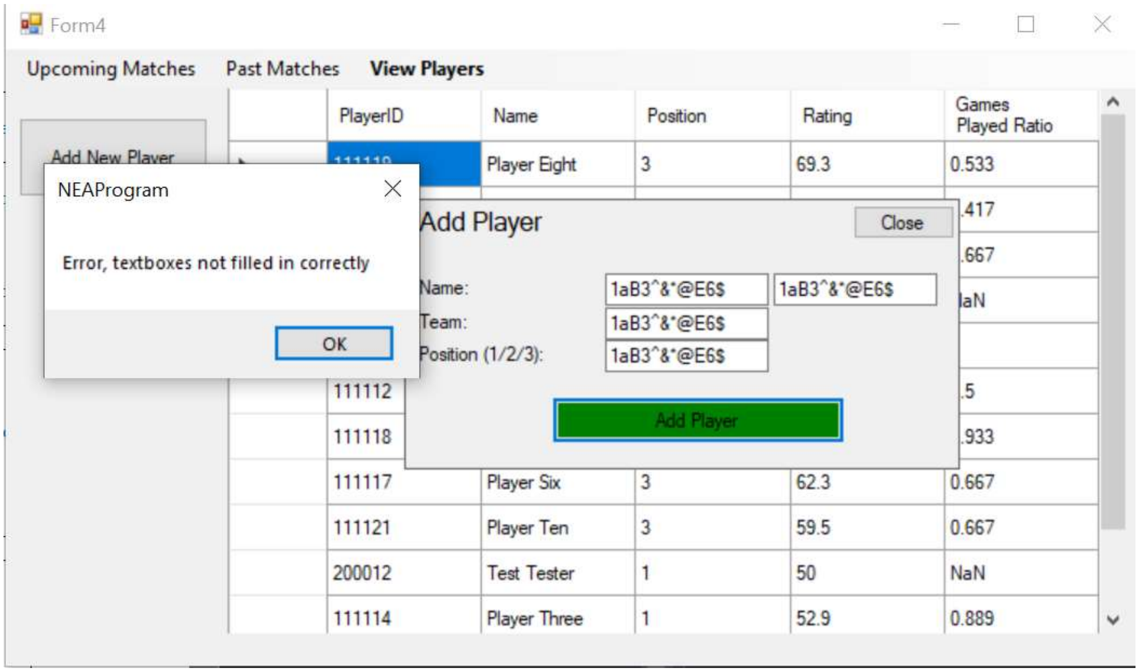


PlayerID	Name	Position	Rating	Games Played Ratio
111119	Player Eight	3	69.3	0.533
111116				.417
111115				.667
200021				laN
111120				
111112				.5
111118				.933
111117	Player Six	3	62.3	0.667
111121	Player Ten	3	59.5	0.667
111114	Player Three	1	52.9	0.889
111113	Player Two	1	53.9	0.8

Add Player

Name: Test Tester
Team: U10
Position (1/2/3): 1

27.2



	PlayerID	Name	Position	Rating	Games Played Ratio
▸	111119	Player Eight	3	69.3	0.533
					.417
					.667
					laN
	111112				.5
	111118				.933
	111117	Player Six	3	62.3	0.667
	111121	Player Ten	3	59.5	0.667
	200012	Test Tester	1	50	NaN
	111114	Player Three	1	52.9	0.889

NEAProgram

Error, textboxes not filled in correctly

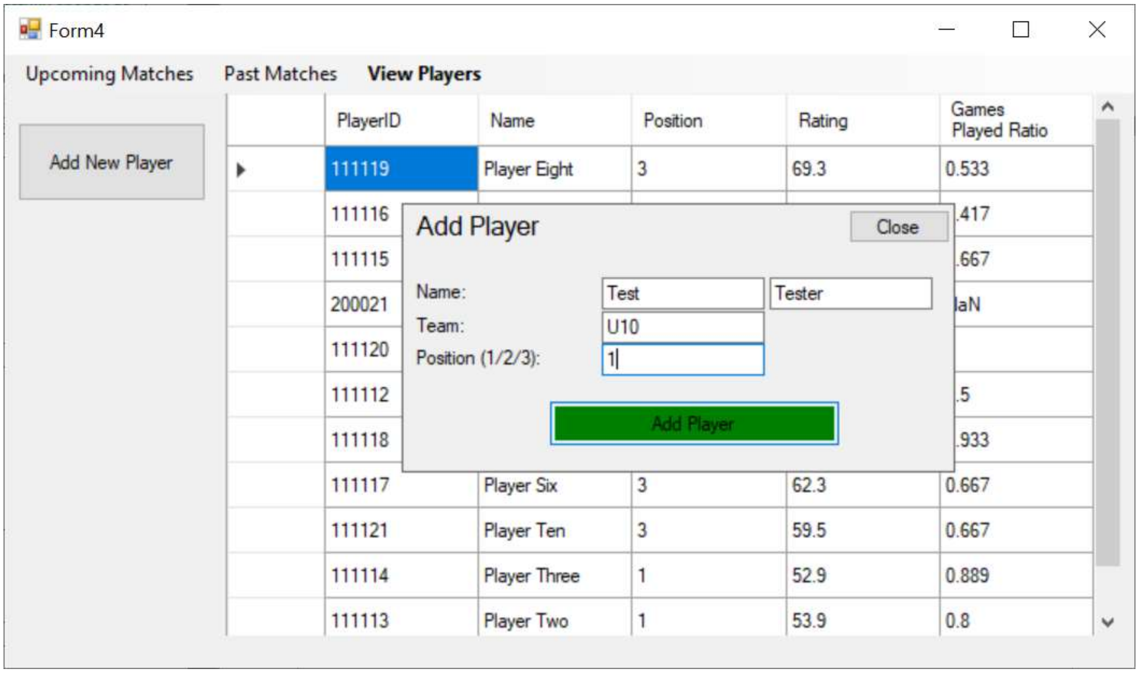
Add Player

Name: 1aB3^&*@E6$ 1aB3^&*@E6$

Team: 1aB3^&*@E6$

Position (1/2/3): 1aB3^&*@E6$

27.3



	PlayerID	Name	Position	Rating	Games Played Ratio
▸	111119	Player Eight	3	69.3	0.533
	111116				.417
	111115				.667
	200021				laN
	111120				
	111112				.5
	111118				.933
	111117	Player Six	3	62.3	0.667
	111121	Player Ten	3	59.5	0.667
	111114	Player Three	1	52.9	0.889
	111113	Player Two	1	53.9	0.8

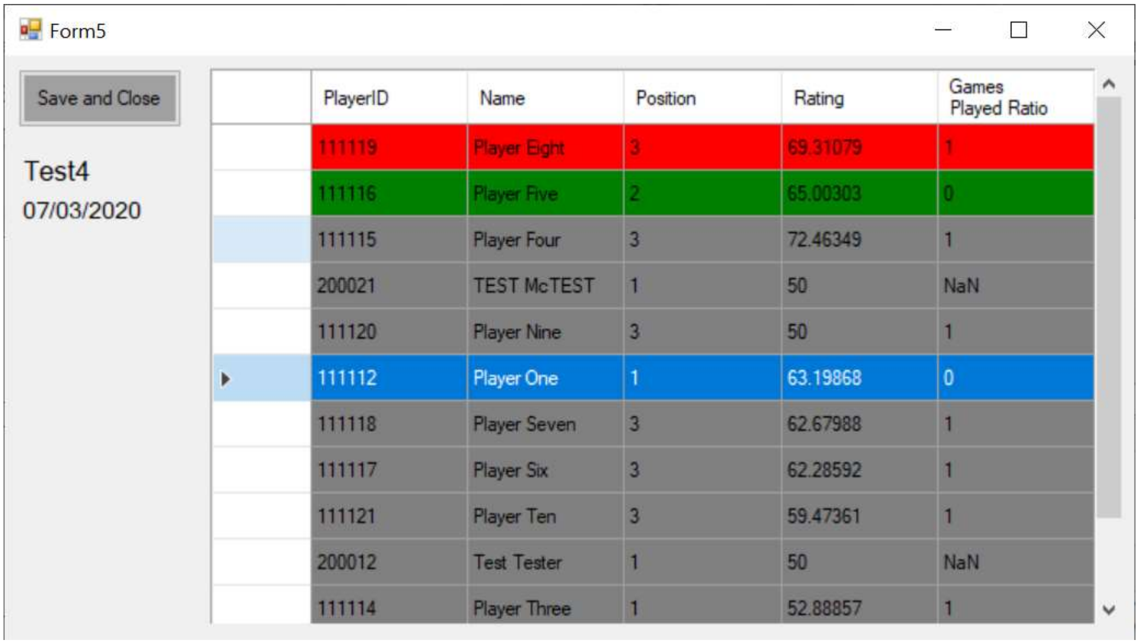
Add Player

Name: Test Tester

Team: U10

Position (1/2/3): 1

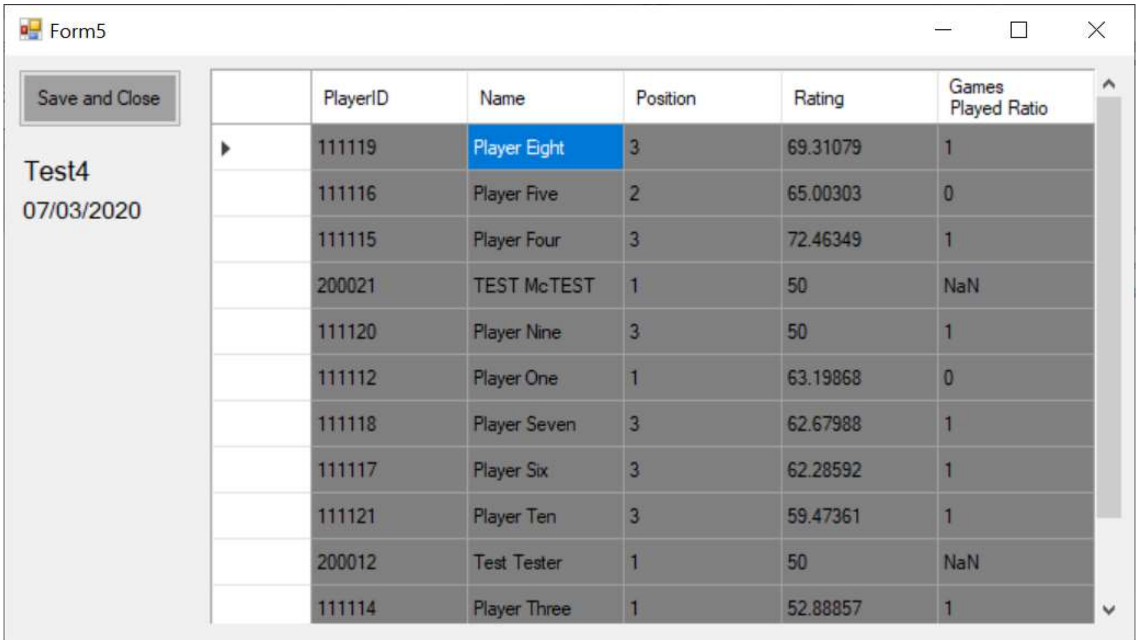
28.1



Test4
07/03/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
	111119	Player Eight	3	69.31079	1
	111116	Player Five	2	65.00303	0
	111115	Player Four	3	72.46349	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
▸	111112	Player One	1	63.19868	0
	111118	Player Seven	3	62.67988	1
	111117	Player Six	3	62.28592	1
	111121	Player Ten	3	59.47361	1
	200012	Test Tester	1	50	NaN
	111114	Player Three	1	52.88857	1

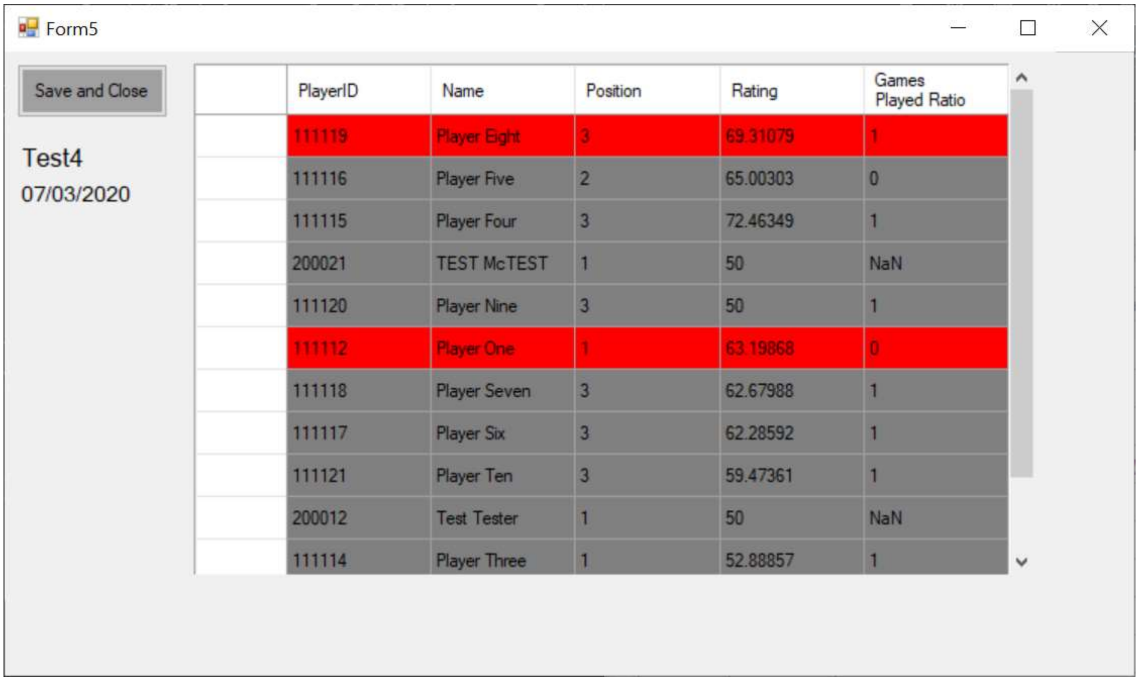
28.2



Test4
07/03/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
▸	111119	Player Eight	3	69.31079	1
	111116	Player Five	2	65.00303	0
	111115	Player Four	3	72.46349	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	63.19868	0
	111118	Player Seven	3	62.67988	1
	111117	Player Six	3	62.28592	1
	111121	Player Ten	3	59.47361	1
	200012	Test Tester	1	50	NaN
	111114	Player Three	1	52.88857	1

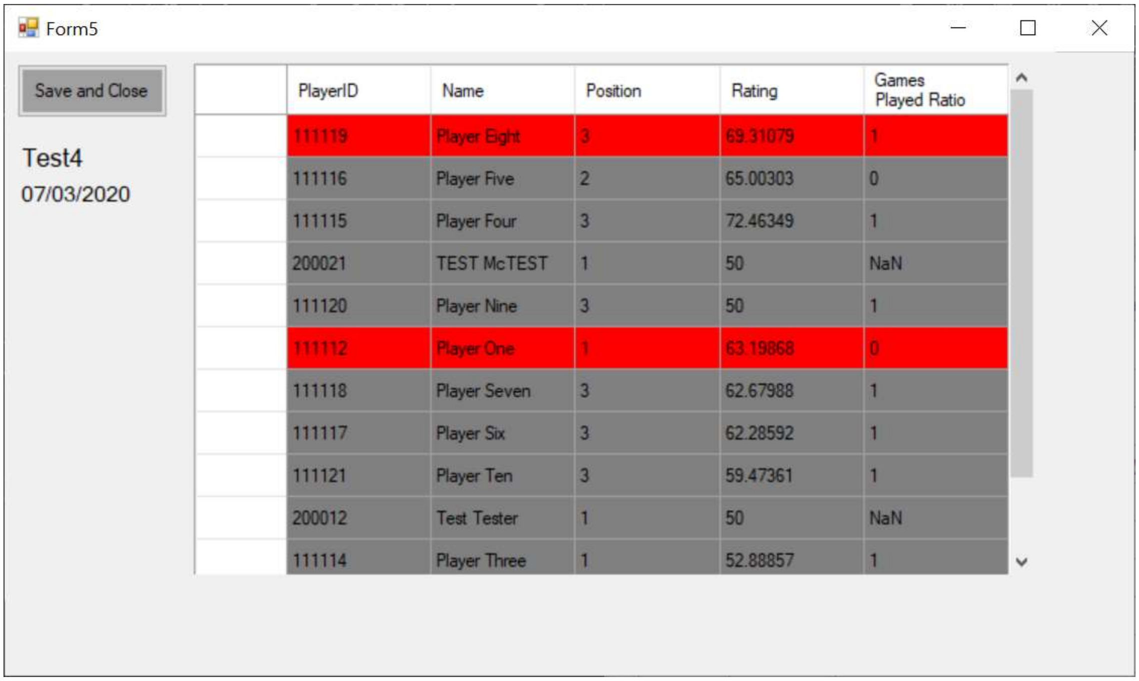
29.1



Test4
07/03/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
	111119	Player Eight	3	69.31079	1
	111116	Player Five	2	65.00303	0
	111115	Player Four	3	72.46349	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	63.19868	0
	111118	Player Seven	3	62.67988	1
	111117	Player Six	3	62.28592	1
	111121	Player Ten	3	59.47361	1
	200012	Test Tester	1	50	NaN
	111114	Player Three	1	52.88857	1

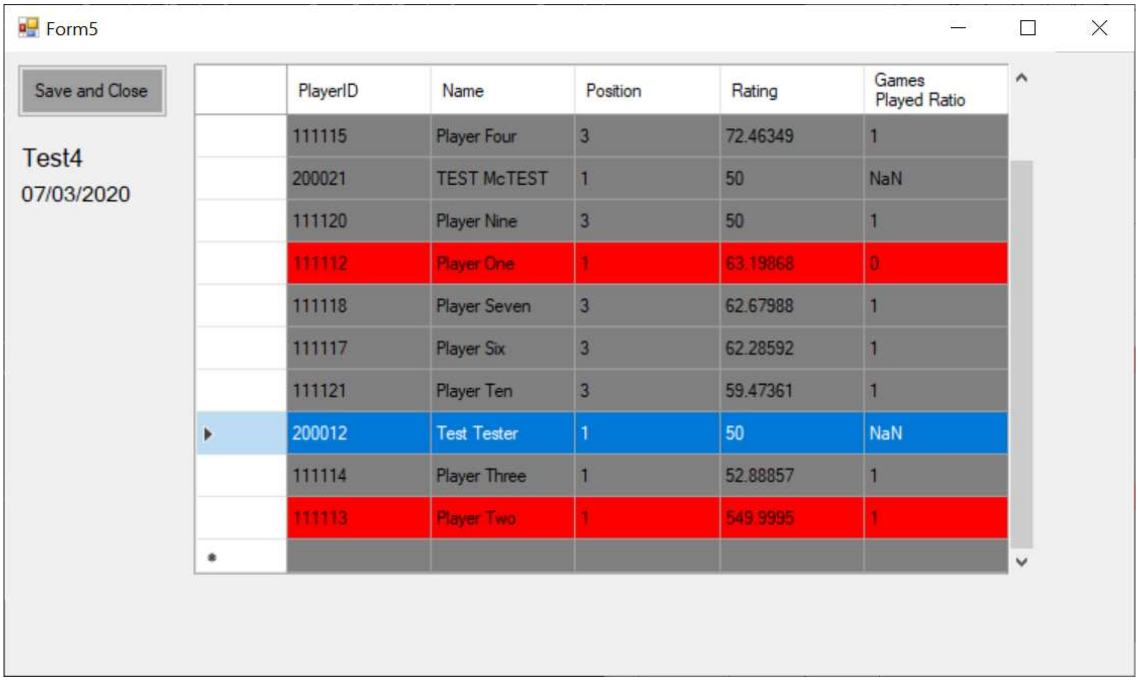
29.2



Test4
07/03/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
	111119	Player Eight	3	69.31079	1
	111116	Player Five	2	65.00303	0
	111115	Player Four	3	72.46349	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	63.19868	0
	111118	Player Seven	3	62.67988	1
	111117	Player Six	3	62.28592	1
	111121	Player Ten	3	59.47361	1
	200012	Test Tester	1	50	NaN
	111114	Player Three	1	52.88857	1

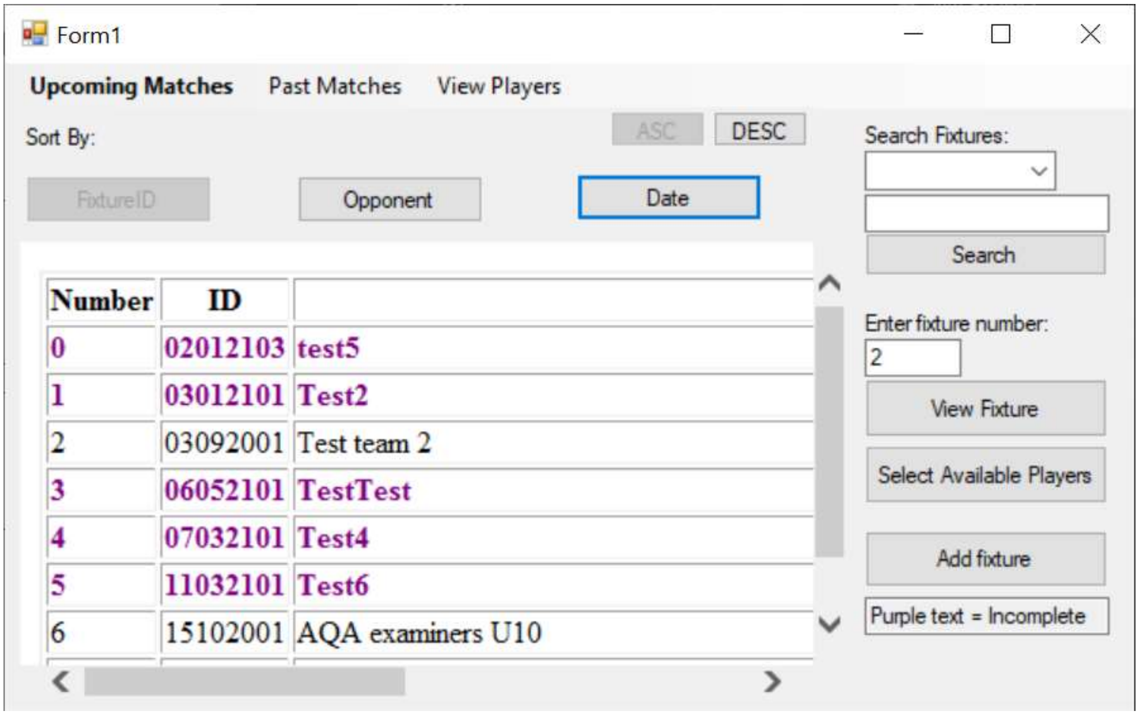
29.3



Test4
07/03/2020

	PlayerID	Name	Position	Rating	Games Played Ratio
	111115	Player Four	3	72.46349	1
	200021	TEST McTEST	1	50	NaN
	111120	Player Nine	3	50	1
	111112	Player One	1	63.19868	0
	111118	Player Seven	3	62.67988	1
	111117	Player Six	3	62.28592	1
	111121	Player Ten	3	59.47361	1
▸	200012	Test Tester	1	50	NaN
	111114	Player Three	1	52.88857	1
	111113	Player Two	1	549.9995	1
*					

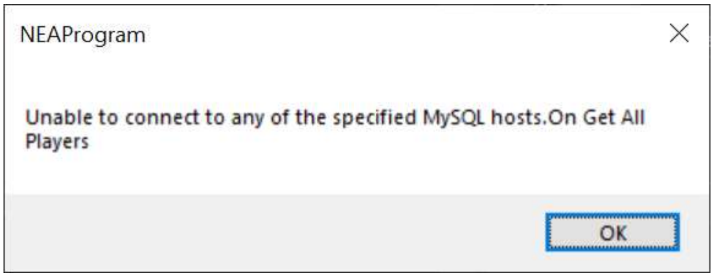
30.1



Enter fixture number: 2

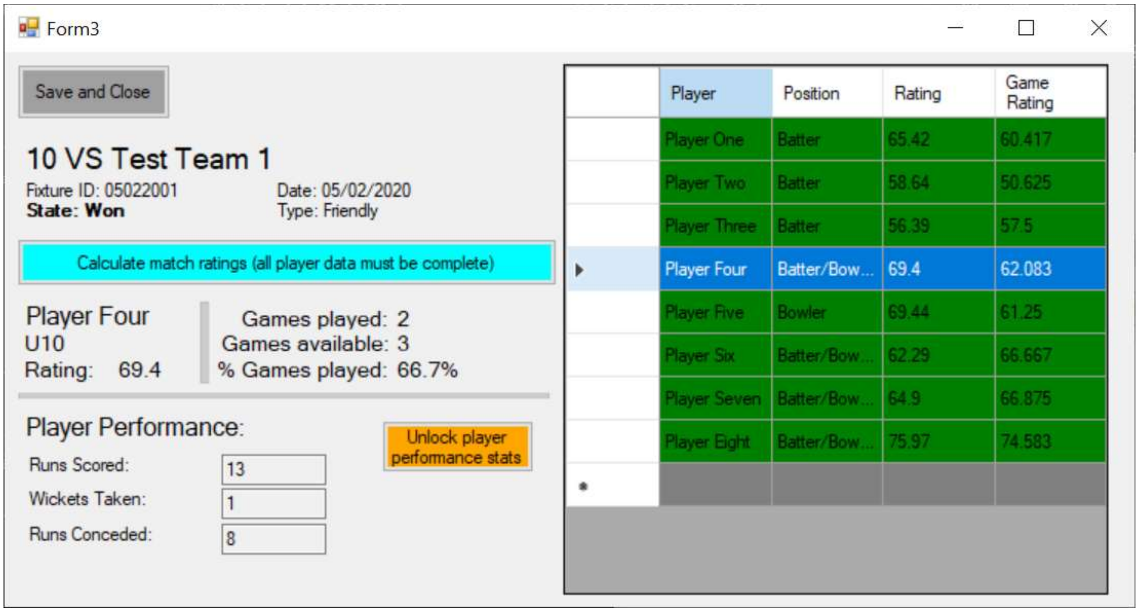
Number	ID	
0	02012103	test5
1	03012101	Test2
2	03092001	Test team 2
3	06052101	TestTest
4	07032101	Test4
5	11032101	Test6
6	15102001	AQA examiners U10

30.2



Unable to connect to any of the specified MySQL hosts.On Get All Players

31


Form3
Save and Close
Calculate match ratings (all player data must be complete)
Unlock player performance stats

10 VS Test Team 1

Fixture ID: 05022001
State: Won

Date: 05/02/2020
Type: Friendly

Player Four
U10
Rating: 69.4

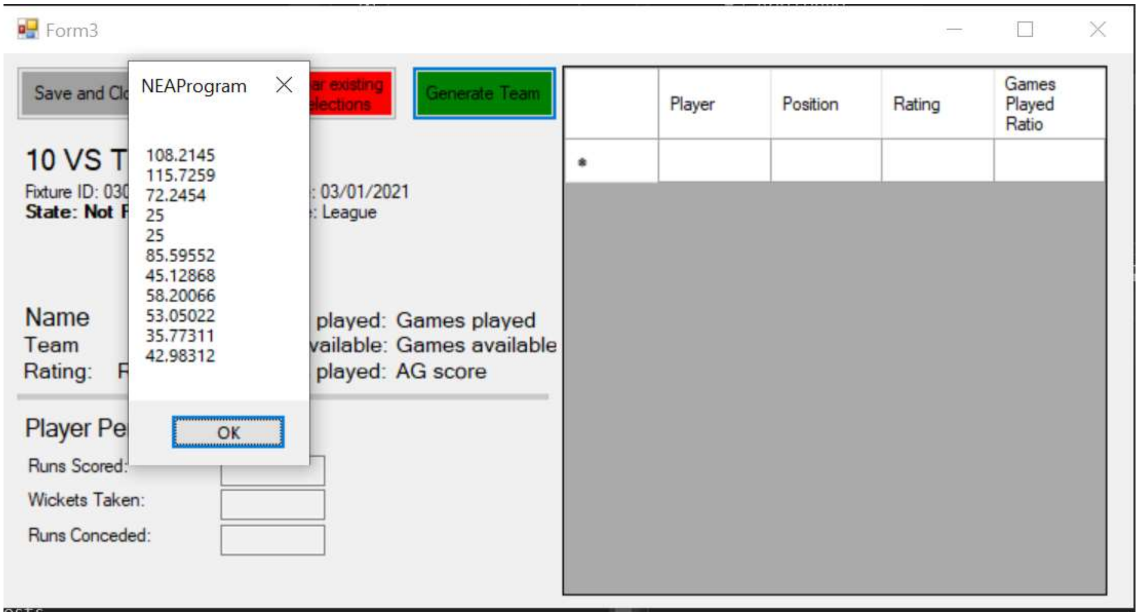
Games played: 2
Games available: 3
% Games played: 66.7%

Player Performance:

Runs Scored: 13
Wickets Taken: 1
Runs Conceded: 8

	Player	Position	Rating	Game Rating
	Player One	Batter	65.42	60.417
	Player Two	Batter	58.64	50.625
	Player Three	Batter	56.39	57.5
▸	Player Four	Batter/Bow...	69.4	62.083
	Player Five	Bowler	69.44	61.25
	Player Six	Batter/Bow...	62.29	66.667
	Player Seven	Batter/Bow...	64.9	66.875
	Player Eight	Batter/Bow...	75.97	74.583
*				

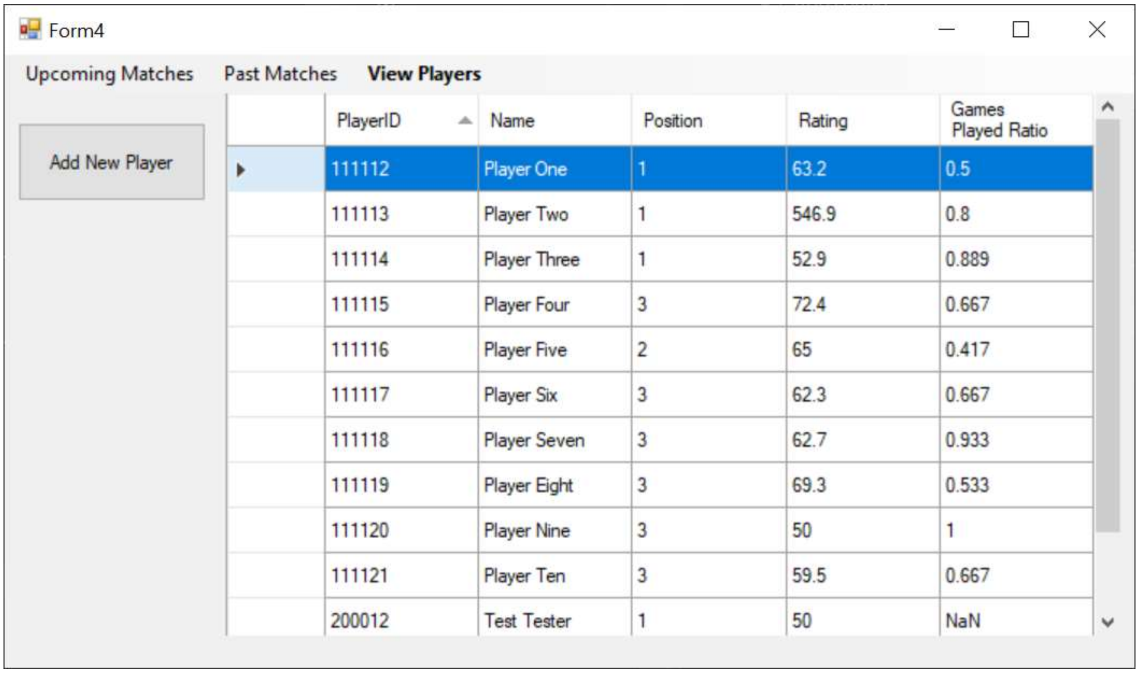
32



NEAProgram

```
108.2145
115.7259
72.2454
25
25
85.59552
45.12868
58.20066
53.05022
35.77311
42.98312
```

33



PlayerID	Name	Position	Rating	Games Played Ratio
111112	Player One	1	63.2	0.5
111113	Player Two	1	546.9	0.8
111114	Player Three	1	52.9	0.889
111115	Player Four	3	72.4	0.667
111116	Player Five	2	65	0.417
111117	Player Six	3	62.3	0.667
111118	Player Seven	3	62.7	0.933
111119	Player Eight	3	69.3	0.533
111120	Player Nine	3	50	1
111121	Player Ten	3	59.5	0.667
200012	Test Tester	1	50	NaN

Sources

https://pulse-static-files.s3.amazonaws.com/ecb/document/2018/05/08/21b95b93-1fa6-4a45-8d2d-d9d8a4658152/Junior-Cricket-Formats-short-version.pdf

http://cricketcoachingblog.co.uk/2016/01/18/pairs-cricket/

https://www.chiddcc.uk/

https://play-cricket.ecb.co.uk/hc/en-us/sections/360000018049-Play-Cricket-API-documentation

https://westsycl.play-cricket.com/

https://teamer.net/

https://www.youtube.com/watch?v=qnPo0DY_QF4&list=PLKXxOWAJgVPNJhBdZ35JDLAPGJ-a7ml44

https://online.godalming.ac.uk/mod/assign/view.php?id=32898

https://en.wikipedia.org/wiki/Cricket#Fielding

https://en.wikipedia.org/wiki/Fielding_(cricket)