Evaluation

The Programming Language

To create my Junior Football System, I used the programming language Visual Basic. I decided that this was the best language to use as it is the language we have been using as a class over the course of sixth form and at GCSE, so I am experienced when it comes to using it. I also strongly believe that it had the capabilities to bring my idea to life and to allow me to make all of the forms that I planned in my design.

The main thing that the language let me do was store the key information needed for Players, their Parents, Coaches, Stock Items, and Users of the system. When this data was entered by a user using the respective form, it was then stored into a .txt text file. This was the best way to store the data, as it was easy to locate, and the data was set out neatly within the file. The next thing that Visual Basic allowed me to do to this data was to edit and delete it. I was able to create forms that would display the information within the text file using textboxes, and I could then alter parts of information and save it with ease. On this edit form, there would be a button labelled 'Delete', which you could press when a certain player, parent etc was displayed to remove that specific record from the file. This was very useful because the club may have to change a piece of information quickly and effectively (e.g. if a mother changes her surname due to marriage, or if a coach moves addresses).

A great feature of the Visual Basic language is it's GUI (Graphical User Interface). Because of this, I could add many objects to my forms such as labels, textboxes, buttons, combo boxes etc. These all allow users to easily operate the system, as I can ensure that the forms are easy to follow by appropriately naming buttons, linking textboxes to labels and giving them appropriate options for fields which used a combo box (e.g. to enter a player's position, give options 'goalkeeper, defender, midfielder, forward').

Another thing which was vital to the functionality of the system which the code allowed me to do was to run validation checks on all the data entered into text files. I used a wide variety of checks, from length checks, presence checks, and format checks to ensure that the data being put into the system was correct and suitable. I also used a double entry check adding a user to the system, as it asks them to enter their password, and then re-enter it to ensure they haven't made any spelling errors. This then checked that both textboxes displayed the same password, and an error message was shown if they did not match.

Finally, I was able to create routines and data structures in the code, which I used to create search forms and sorts within my data. I created routines to Load data from the text files, which I used in my search forms, in which I allowed the user to enter the first name and surname of a player, parent or coach, and if a matching record is found, it then displayed the information saved for that person. The data structures I made were used to declare variables for certain data stored (e.g. for Player's data, I needed to declare variables

FirstName, Surname, Position, Age Group etc). This was very helpful, as it would allow me to save the data entered into textboxes a lot faster.

Comparison to Similar Systems

During my investigation, I conducted research into systems which were available that were similar to mine. From this, I then had a rough idea of what I needed the system to include. I decided that for my Subscription Payments, I would need to store data of:

- Player Name
- Paid/Not Paid?
- Contact Number

When it actually came to my own payments system, I worked out that this would not be enough to successfully track the payments of every player.

I established that I would need to display:

- Player IDs for all Players
- Names of every Player
- The amount each player has paid
- The amount each player has left to pay

And I knew that the 'Make a Payment' function would need to include:

- A dropdown list containing all Age Groups
- A dropdown list containing all PlayerIDs within that Age Group
- A textbox to enter the amount paid
- A button to make payment

From looking at the original plan I had for subs payments compared to what I ended up creating, the method that was implemented was a lot more in depth than the original.

Firstly, I included the Player ID of every player, and their amount paid/left to pay along with their name. The Subscription Payments form in my system contained a rich text box which included the Player ID, Player Name, Amount Paid and Amount Left to Pay for every player within an Age Group, depending on which Age Group the user selected from the dropdown list at the top of the form. That was an important feature to add, as it allowed the user to clearly view the payment information, and it made it easy for them to understand which player had paid what.

The second feature which I added was the 'Make a Payment' function, which was at the bottom of the form underneath the rich text box. Originally, there was not really a way to keep track of payments made, there was just a box next to the player name which was ticked when they had fully paid, and I planned for this system to be similar. Looking back, this was an awful idea at first because it didn't allow for payments to be made in instalments, as there was no way to track how much they had left to pay. To make a

payment, the user has to select an age group, then select the PlayerID of the player within that Age Group that they want to make a payment for (they would know the PlayerID as it would be displayed above in the textbox), and then enter an amount that they are paying for the player. The system would then add amount to the amount they have already paid, and subtract it from the 'amount left to pay figure' when the Make Payment button is pressed. If the user pays less than £1, an error message is displayed, as that is the minimum payment amount, and if the user overpays and enters a payment larger than the amount left to pay, an error message will also be displayed.

Pros and Cons of the System

Looking back at the last few months spent creating my system, I believe that there are many positives to it, but there are also a few drawbacks to it which I have picked up on.

Starting with the pros, I think that the design and layout of my system is of a high quality, and that the colours, fonts and font sizes are all suitable, and don't bring any distraction to the user. I also strongly believe that all of my buttons are labelled appropriately and the forms are all titled accordingly, which leads me to believe that my form is easy for the user to navigate around. Another feature that I like about my system is that it has different levels of access, depending on where the user ranks in the club's hierarchy. If the user is a system administrator, then they have access to all of the system's features, but if they were just a regular staff member, they can only access certain aspects. This helps keep the data secure, as it cannot be edited and deleted by anyone who has access to the system. Finally, another feature that I love from my system is the range of unusual view functions the user has access to. They can view a Player's contact details using the name of the player, they can view the coach of an age group by selecting an age group, and they can also view all players who play in a certain position, and all players within a certain age group. These are helpful features which you wouldn't see in other conventional systems, bringing a uniqueness to my project.

Although I think my system is fantastic and robust. I did run into a few negatives when it came to testing it. Firstly, I noticed that in my form where users are able to search for a specific item of stock using the description. I noticed that this could potentially be an issue, because the user would have to enter the item description exactly as it's listed in the text file. So for example, if the user inputted 'Away Kits Medium' to search for, it would not bring up the information for medium sized away kits, as the description for this item is 'Medium Away Kits'. This means that this search function in particular is not very practical, because if the user doesn't know exactly what the item is named in the file, they cannot search for it, or it will take them ages to guess what it is titled. The second problem I picked up on with my system involves the generation of Usernames. When a new user is created, their username is generated by the system using their name and their date of birth. The system contains a form called 'Edit User', which allows an administrator to change any of the data stored for a user of the system, but when they change the user's first name, surname, or date of birth, it does not change the username accordingly. This isn't too much of a problem

when it comes to the system running, but it is quite irritating. The third and final issue I noticed within the system was related to my 'Edit Stock' form. I accidentally included the 'StockID' at the top of the form, so that the user could edit the StockID for any of the records stored. This could be a problem, because the StockID is automatically generated depending on the number of records already present.

Improvements

If I was to make any improvements to my system, they would be relating to the three negatives listed above.

Firstly, there is the issue of the 'Search Stock' function being hard to use because the description entered has to be exactly the same as it is in the file. To improve this and hopefully overcome this problem, I would change this so the user would search for an item using the StockID instead of the description, although I think this is a more convenient way for the user to search for an item, they would still need to know the ID of the item before searching, meaning that it isn't 100% useful.

Secondly, the next issue was the generated usernames becoming incorrect when details are edited. To get around this issue, I would need to implement code in the Edit User form which generates a new username for the user, using the changes made. For example if the User was called Jeremy David, born 19/11/1993, then his username would be DavidJ19. If the user was to change his first name to Albert, then his username would have to be changed to DavidA19.

Finally, the last issue that I ran into was my edit stock form giving the user the option of editing the stock id. As this isn't wanted, I would simply have to remove the label and textbox from the form. And remove the code where it is declared and linked to the textbox. By making these changes, it will fix some of the problems previously faced with the system, and it would improve it.

As well as those three negatives, there are other aspects of the system which I think could do with improving.

Changes to My Approach in the Future

If I was to redo this project again, I would do a few things differently in my approach, which I think would have been beneficial and would have made my coursework stronger overall. Firstly, I would have kept a log of what features I added/what changes I made when I last sat down and did the work, as I think this would have been helpful to me, because there were times when I sat down to do the project and I didn't know where to start, and there were times where I didn't know what the last thing I did was, which made it hard for me to know what to do/what was unfinished. Similarly, the next thing I would have done differently would have been that I would have worked on my system more often, because I often left

large periods of time between my coursework sessions. Which I believe threw me off track and stopped me from getting in any sort of routine or rhythm. If I set a small period of time every day to work on the coursework, I believe I would have been a lot more productive, and wouldn't have had to cram as much work in towards the end of the time we were allocated.