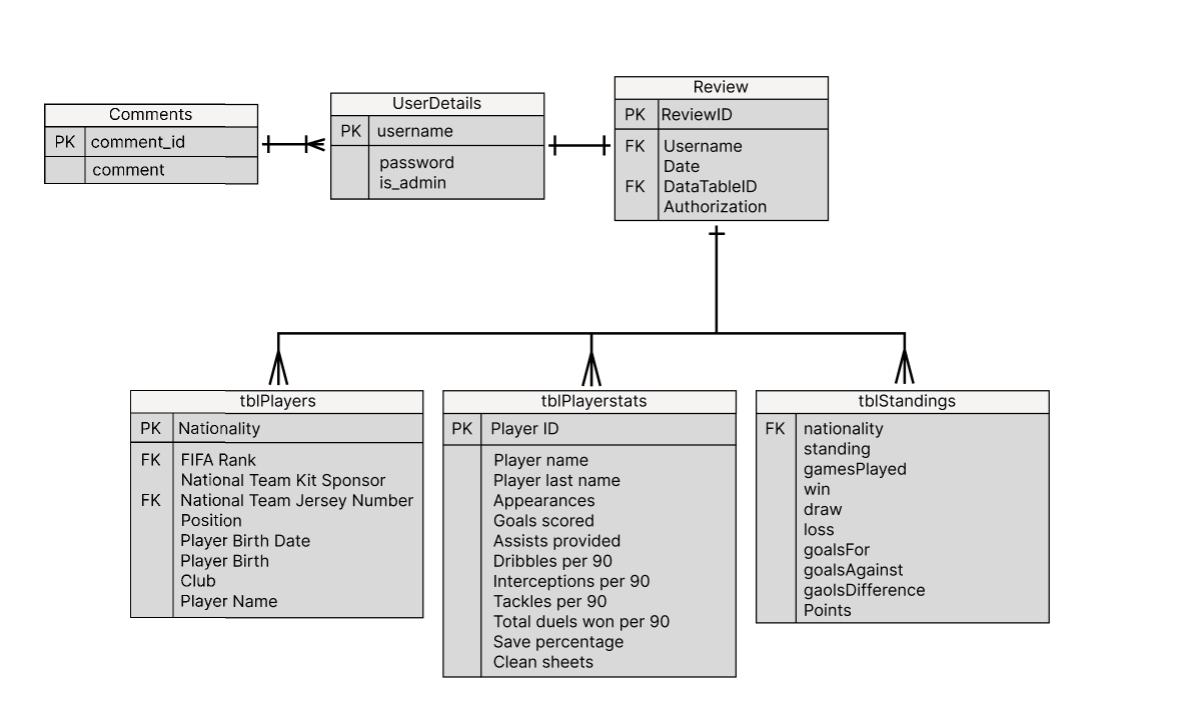
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Context:**  The popular website BigSoccer.com is an online soccer forum where people can browse and search a large variety of FIFA related articles and participate in discussion among the community. The Pimlico State High School librarian want to launch an online World Cup Club to engage more students beyond normal and typical book discussions. To protect student data and security an interactive desktop application within the school network is offered for exchanging FIFA World Cup related Information. The goal is to increase student participation, facilitate continuing debates and arguments, and foster a healthy community of World Cup fans at the PSHS school.  Develop an interactive desktop application according to the requirements outlined in the provided technical proposal. Document the problem-solving process and demonstrate the functionality of the components of the digital solution in a video recording. | **Constraints:**  Constraints involved in this task that could reduce the effectiveness could be that the soccer data may have inconsistencies therefore making it hard to integrate into a SQL data base, The creator of the digital solution could not have the skill to be able to create a solution that integrates a python gui with an SQL database. Another constraint could be that as more and more soccer games are played, and the more players join the data size will get bigger and bigger possible causing issues. Keeping the integrity of the data is another constraint as it would require regular backups, data validation checks and proper error handling mechanisms to prevent corruption. Performance may be a constraint as if the database and solution are not optimized the digital solution might not be able to run on all computers. | **Existing solutions:**  **Solutions Available:**  <https://www.soccer.com>  BigSoccer.com is a leading online soccer forum for global fans, with a massive collection of conversations, articles, and match analyses. Users can register, create profiles, and participate in discussions while creating collections of their Favorite players, teams, and match moments.  **Solutions Problems:**   * The Solution does not meet all the prescribed criteria. * The Librarian will have no access to any of the data. * No Clear treeview of all the data compiled. * No ability to have an admin account | **User Requirements** | | **Essential** | **Desirable** |
| Admin | | * Allow the user to access a report indicating information such as the number of registered users and the number of comments completed. * Be able to delete user accounts and comments. * Upload football team and player details | * Add and delete individual team and player details. * Approve registered users. * View details of all registered student users * Delete registered student users. * Change the password of registered users. * Approve student comments before being publicly available. * View details of all comments completed (based on game or team details). |
| User | | * View football Teams data * View reviews of teams or players (Based on queries) * Be able to register. * Be able to leave comments under each data treeview. | * Football commentary based on game or team. * View just their own commentary. * View and comment on another student’s commentary. |
| |  |  |  | | --- | --- | --- | | **Criteria:** | Prescribed | Self-Determined | | Task | Develop an interactive desktop application according to the requirements outlined in the provided technical proposal. Document the problem-solving process and demonstrate the functionality of the components of the digital solution in a video recording.   * Recognise and describe user interface components, usability principles. * Symbolisation Using mind map and wire frames. * Algorithms developed in pseucode | Develop a digital solution that is efficient and punctual at all of its main processes and functions for a more overall cohesive and better user experience. | | Data | * Upload soccer data to the application database from .csv file. * Display the following information related to football teams:   + Team Details   + Player Details and Statistics   + Match Details and Statistics * Store registration details of student users such as:   + Username   + Password   + First name   + Last name * Store user comments | Be able to be viewed in a tree diagram with the ability to sort and search through the data. | | User Interface | Must incorporate User interface components and usability principles. | Have a cohesive and uniform interface for ease of use. | | Algorithms | * Read and record from a CSV file and store them in a database. * Enable users to register. * Enable user to login. * Enable users to insert a comment on a football or player | Be able to read data from a CSV file and transfer it to a premade SQLite database. |   A diagram of a diagram  Description automatically generated with medium confidence | | | |
|
|
|
|
|

|  |  |
| --- | --- |
| **IA2 Data Story Micah Graffin** | |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Figure 4 Data Sample:** These are the first three entries from the data set retrieved from Kaggle (Kaggle, 2022):   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Position | Team | Games Played | Win | Draw | Loss | Goals For | Goals Against | Goal Difference | Points | | 1 | Argentina | 7 | 6 | 0 | 1 | 15 | 8 | 7 | 18 | | 2 | France | 7 | 5 | 0 | 2 | 16 | 8 | 8 | 15 | | 3 | Croatia | 7 | 4 | 2 | 1 | 8 | 7 | 1 | 14 | | Data Explanation:  The data for this solution has been obtained from Kaggle (Kaggle, 2022). There are two cs files that contain raw data on the topic of Soccer in 2022. The data in the first csv file is cut into Position, Team, Games Played, Win, Draw, Loss, Goals for, Goals Against, Goals difference and points. The second Data Set FIFA Players.csv has Nationality, FIFA Ranking, National Team Kit Sponsor, Position, Player DOB, Club, Player Name, Appearance, Goals Scored, Assists provided, Dribbles per 90, intercepts per 90, Tackles per 90, Total Duels Won per 90, Save Percentage, Clean Sheets and Brand Sponsor/Brand Used. |   **Data set 2:** FIFA Players.csv   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Nationality | FIFA Ranking | National Team Kit Sponsor | Position | National Team Jersey Number | Player DOB | Club | Player Name | Appearances | Goals Scored | Assists Provided | Dribbles per 90 | Interceptions per 90 | Tackles per 90 | Total Duels Won per 90 | Save Percentage | Clean Sheets | Brand Sponsor/Brand Used | | Argentina | 2 | Adidas | GK | 23 | Sep 2, 1992 | Aston Villa | Emiliano Martinez | 7 | 0 | 0 | 0 | 0 | 0 | 0.65 | 46.67% | 43% | Adidas | | Argentina | 2 | Adidas | GK | 1 | Oct 16, 1986 | River | Franco Armani | 0 | - | - | - | - | - | - | - | - | Nike | | Argentina | 2 | Adidas | GK | 12 | May 20, 1992 | Villarreal | Geronimo Rulli | 0 | - | - | - | - | - | - | - | - | Adidas | | |
| **Figure 5 Data Normalization and cleaning:**  Cleaning: Before normalization, the data must be cleaned and pre-processed to eliminate spelling errors, Unicode characters, structural errors, duplicate or irrelevant data and any anomaly that will make it hard to interpret and display the data in a database. For the data that will be used in the solution there where some irrelevant columns deleted, and the format of dates was changed to be more compatible with SQLite.  Normalization: To create an efficient database structure, normalization was used, with the aim of minimizing data redundancy, normalization resulted in several relational tables initially being created. Normalizing reduces redundancy and dependency, typically through a set of rules called normal forms, this is to ensure data integrity and minimize anomalies.  1NF: In this data there was some cleaning to make the data more concise and compatible with the database program.  FIFA 2022 csv did not require cleaning as all the data was valid. Additionally, none of the data can be normalized into sub tables so FIFA.csv will be left as is and migrated to a SQL database as tblstandings.db  FIFA 2022 Players.csv: The data table was exceptionally clean and organized, requiring no cleaning or normalization operations. Each entry strictly adhered to the defined template, removing the need for any preprocessing processes. As a result, this immaculate information allowed for smooth analysis and interpretation, making it easier to discover useful insights. | Tables to be created for the database:  The first SQL Database file called IA2.db will have three main tables created are called tblPlayers, tblStandings and tblPlayerstats. Theses tables will be called on to create treeview diagrams for the data for students to interact with and comment on in the solution. The second Data Base file will be called tblUsers.db and will contain the tables for the comments that the students will create and the user data like username, password and is\_admin. This file will be called on when the user is logging in, when the admin is deleting accounts or comments or when students are making comments. |
| **Entity Relationship:** | |

A white grid with black text

Description automatically generatedA screen shot of a white sheet

Description automatically generated

tblPlayers

tblStandings

A screenshot of a computer

Description automatically generatedA white grid with black text

Description automatically generatedA white grid with black text

Description automatically generated

tblComments

tblUsers

tblPlayerstats

A close-up of a computer screen

Description automatically generated

User Input Function:

This function is designed to take input from the user for their username and password.

For security it hides the characters as ‘\*’.

Algorithms: Login Functions

BEGIN

FUNCTION userInput

    USERINPUT username

    USERINPUT password

    FOR char IN password

        DISPLAY "\*"

    ENDFOR

ENDFUNCTION userInput

FUNCTION login

    USERINPUT loginButton

    WHEN loginButton IS PRESSED

        GET username, password

        IF username OR password IS null

            DISPLAY "Error: Invalid Entry"

        ELSEIF username OR password CONTAINS invalid

            DISPLAY "Error: Invalid Entry"

        ELSE

            hashedPassword = HASH\_SHA1(password)

            CONNECT to Accounts

            SELECT username, hashedPassword FROM tblAccounts

            IF username = username.tblAccounts AND hashedPassword = hashedPassword.tblAccounts

                DECLARE loggedIn BOOL

                GLOBAL loggedIn

                loggedIn = TRUE

            ELSE

                DISPLAY: "Error: Username or Password does not match"

            ENDIF

        ENDIF

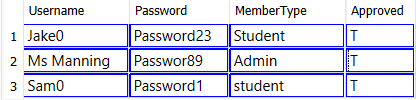
ENDFUNCTION login

userInput

login

END

A diagram of a computer program

Description automatically generated

***Data Flow Diagram (DFD)***

A data flow diagram depicts how information travels through a process or system. It consists of data inputs and outputs, data stores, and the numerous subprocesses that the data passes through. DFDs use standardized symbols and syntax to define multiple entities and their relationships.   
Data flow diagrams illustrate systems and processes that are difficult to define in words. You can use these diagrams to map out an existing system and improve it, or to plan out a new system to be implemented. Visualizing each piece allows you to easily discover inefficiencies and create the best possible system. (Lucidchart, 2023)

Input: Username – Empty

Input Password –Passeord23

Username is Empty

Display Error msg.

Clear Input Box

Input: Username – Jake0

Input Password – Empty

Username is not empty (Continue)

Password Is Empty

Display Error message.

Cleat Input Box

Desk Checks:

Input: Username – Jake0

Input Password – Password23

Username is not empty or a digit (Continue)

Read Data in tblAccounts.

Username, Password found.

Open student window

Login Function:

Takes user input for login button, when button pressed it retrieves their username and password, it checks to see if their username and password is null and if so displays an error.

Checks to see if the username and password has any invalid characters and displays an error if there is.

If the username and password are valid, it hashes the password using SHA1 algorithm.

Then attempts to connect to the account and selects the username and hashed password from tblAccounts.

If entered username and password match the ones in the tbl it sets global value to logedin to “True”.

If the username and password do not match it displays and error saying “Mismatch Username of Password”

Account Details Table Data

Data Components Required:

* Read and record from a CSV file and store them in a database.
* Enable users to register.
* Enable user to login.

Enable users to insert a comment on a football or player.

* Upload soccer data to the application database from .csv file.
* Display the following information related to football teams:
  + Team Details
  + Player Details and Statistics
  + Match Details and Statistics
* Store registration details of student users such as:
  + Username
  + Password
  + First name
  + Last name

Store user comments.(Lever, 2024)

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm: Connect to database and move csv data to sql  CONNECT to database 'IA2.db'  cursor = GET\_CURSOR  EXECUTE SQL query to drop table tblPlayerstats  EXECUTE SQL query to drop table tblPlayers  EXECUTE SQL query to drop table tblStandings  OPEN 'tblStandings.csv' as file  csv\_reader = CREATE\_CSV\_READER for file  headers = GET\_HEADERS from csv\_reader  columns = JOIN headers to form columns string  EXECUTE SQL query to create table tblStandings  (nationality TEXT, standing INTEGER, gamesPlayed INTEGER,  win INTEGER, draw INTEGER, loss INTEGER, goalsFor INTEGER,  goalsAgainst INTEGER, goalsDifference STRING, Points INTEGER)  FOR EACH row IN csv\_reader  EXECUTE SQL query to insert row data into tblStandings table  COMMIT changes to the database | Desk Check:   1. Start the function data\_import 2. Execute SQL query to drop table tblstats 3. Execute SQL query to drop table tblPlayers 4. Execute SQL query to drop table tblStandings 5. Open tblStandings.csv file 6. Create a csv\_reader object for the file 7. Get the headers of the csv file 8. Join the headers to form columns string 9. Create a new table tblStandings with specified columns 10. Iterate over each row in the csv file 11. Execute SQL query to insert row data into tblStandings table 12. Commit the changes to the database. 13. End of function data\_import | Algorithm: allow user to register  REGISTER\_USER  DECLARE username  DECLARE password  OPEN\_CONNECTION to database using self.db\_file  INITIALIZE\_CURSOR    TRY  EXECUTE\_SQL\_QUERY to insert new user into the users table  INSERT INTO users (username, password, is\_admin) VALUES (?, ?, ?)  WITH VALUES (username, password, 0)  COMMIT\_CHANGES to the database  PRINT "User registered successfully"  RETURN True  EXCEPT sqlite3.IntegrityError  PRINT "User already exists."  RETURN False  END REGISTER\_USER | Desk Check:   * Start the function data\_import * Execute SQL query to drop table tblstats * Execute SQL query to drop table tblPlayers * Execute SQL query to drop table tblStandings * Open tblStandings.csv file * Create a csv\_reader object for the file * Get the headers of the csv file * Join the headers to form columns string * Create a new table tblStandings with specified columns * Iterate over each row in the csv file   + For each row:     - Execute SQL query to insert row data into tblStandings table   + Commit the changes to the database. * End of function data\_import |
| Algorithm: Input a comment and saving it to the SQL database  SUBMIT\_COMMENT\_METHOD  comment = GET\_COMMENT from self.comment\_entry  IF comment is not empty  # Save comment to the database  CALL save\_comment\_to\_database(comment)  # Clear the entry field after submission  CALL self.comment\_entry.delete(0, tk.END)  # Optionally, you can display a message to indicate successful submission  SAVE\_COMMENT\_TO\_DATABASE\_METHOD  SET db\_file to "tblUsers.db"  CONNECT to database using db\_file  GET\_CURSOR  TRY  EXECUTE SQL query to create table Comments  (id INTEGER PRIMARY KEY AUTOINCREMENT, comment TEXT)  EXECUTE SQL query to insert comment into Comments table  COMMIT changes to the database  CLOSE connection to the database  PRINT "Comment saved successfully!"  CATCH Exception AS e  PRINT "Error occurred while saving comment:", e | Desk Check:   * Start the SUBMIT\_COMMENT\_METHOD * comment = "This is a test comment" * IF comment is not empty * Start the SAVE\_COMMENT\_TO\_DATABASE\_METHOD * SET db\_file to "tblUsers.db" * CONNECT to database using db\_file * GET\_CURSOR * TRY * EXECUTE SQL query to create table Comments * (id INTEGER PRIMARY KEY AUTOINCREMENT, comment TEXT) * EXECUTE SQL query to insert comment into Comments table * COMMIT changes to the database * CLOSE connection to the database * PRINT "Comment saved successfully!" * End of TRY * CATCH Exception AS e * PRINT "Error occurred while saving comment:", e * End of CATCH * End of SAVE\_COMMENT\_TO\_DATABASE\_METHOD * Clear the entry field after submission * CALL self.comment\_entry.delete(0, tk.END) * Optionally, you can display a message to indicate successful submission * End of IF * End of SUBMIT\_COMMENT\_METHOD | Algorithm:  DELETE\_SELECTED\_USERS\_METHOD  # Get selected items  selection = GET\_SELECTION from self.user\_tree  IF selection is not empty  # Ask for confirmation before deleting  IF MESSAGE\_BOX askyesno("Confirmation", "Are you sure you want to delete the selected user(s)?")  SET db\_file to "tblUsers.db"  CONNECT to database using db\_file  GET\_CURSOR  # Delete selected users from the database  FOR EACH item IN selection  # Assuming "Username" is unique and using it to identify the user  username = GET\_USERNAME from self.user\_tree.item(item, "values")[0]  EXECUTE SQL query to delete user from Users table WHERE username = username  COMMIT changes to the database  CLOSE connection to the database  # Refresh the user treeview  CALL self.user\_tree.delete(\*self.user\_tree.get\_children())  CALL self.load\_users()  MESSAGE\_BOX showinfo("Success", "Selected user(s) deleted successfully.")  ENDIF  ELSE  MESSAGE\_BOX showwarning("Warning", "Please select one or more users to delete.")  ENDIF  END DELETE\_SELECTED\_USERS\_METHOD | Desk Check:   * Start the DELETE\_SELECTED\_USERS\_METHOD * Get selected items * selection = GET\_SELECTION from self.user\_tree * IF selection is not empty * Ask for confirmation before deleting * IF MESSAGE\_BOX askyesno("Confirmation", "Are you sure you want to delete the selected user(s)?") * SET db\_file to "tblUsers.db" * CONNECT to database using db\_file * GET\_CURSOR * FOR EACH item IN selection * username = GET\_USERNAME from self.user\_tree.item(item, "values")[0] * EXECUTE SQL query to delete user from Users table WHERE username = username * COMMIT changes to the database * CLOSE connection to the database * Refresh the user treeview * CALL self.user\_tree.delete(\*self.user\_tree.get\_children()) * CALL self.load\_users() * MESSAGE\_BOX showinfo("Success", "Selected user(s) deleted successfully.") * ENDIF * ELSE * MESSAGE\_BOX showwarning("Warning", "Please select one or more users to delete.") * End of DELETE\_SELECTED\_USERS\_METHOD |

Algorithms – Each DFD Process Gets and algorithm.

A diagram of a software system

Description automatically generated

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Site Map** | | | | **Usability Principles** | | | | |
|  | | | | **Principles** | **How is it addressed** | | | |
| Utility | The state of being useful, [profitable](https://www.google.com/search?client=opera-gx&hs=yST&sca_esv=9922c09ff6e53a84&sxsrf=ACQVn08ivNYmdoZ0KrnqWQ8550vGZXrbhg:1710894041840&q=profitable&si=AKbGX_pt4UlL1m2gNC94R_NJDj6SkP8_L-x-C7qMtu5VjX-uO1akDbVgRdzUzey_rKsG7wjMr7lzfNvi0AfyOGPT_6B1Oz1iINe_GprQP5krSDT3FqUeExA%3D&expnd=1), or beneficial (Oxford Languages, n.d.). How useful is it? Does it provide appropriate functions to allow users to provide input and access output. My solutions provide appropriate functions that allow users and admin to input, read and access data. | | | |
| Portability | The ability to be easily carried or moved (Oxford Languages, n.d.). Is the interface platform-independent? This means the application does not rely on hardware or a particular operating system to run. The solution does require an operating system to run and also requires the latest version of python to be installed, but if all those requirements are met then the program should run smoothly. | | | |
| Responsiveness | The quality of [reacting](https://www.google.com/search?client=opera-gx&hs=1vZ&sca_esv=1a09264f90f8d044&sxsrf=ACQVn0_wkncE2tAp4TKqDKVjxUtfg94iXA:1710918905531&q=reacting&si=AKbGX_q870E3DK3nJ7cu3BOD7pxCEyBcmrM5SWrM1T5VwBHSV3PPOqIMlgETlmFhXpg6la3SCYKgS-lJrviv3rkSEU1RHh0n5bArh0QrhZqZYHZ5lFF_mrQ%3D&expnd=1) quickly and positively (Oxford Languages, n.d.). Will the program adjust to suit the screen size and capabilities of a wide variety of devices.  Does it allow for touch responses? The solution to be created will be able to be interacted with using touch controls but only on a windows system. | | | |
| Learnability / effectiveness | How easy is it for users to accomplish basic tasks the first time they encounter the design? The solution is capable of basic tasks such as searching and sorting | | | |
| Interface Design Factors (Factors Considered) | | Windows Template: | Windows components | | | Reason | Module / Function required | Reason |
| Layout:   * The layout of the interface is neat and easily readable to increase the usability. * The buttons that allow navigation through the pages are in a neat side bar.   Economize:   * spend less; reduce one's expenses.   Communicate:   * share or exchange information, news, or ideas. |  | **Splash window.**  **Search Window** | | | **Identifies program’s title and purpose. Provides access to terms of Use and Acknowledgement forms.**  **Enables user to search for book details and display cover image** | **Menu**  **Enter**  **Terms Of Use**  **Acknowledgements**  **Search**  **(based on Book Title)** | **Provides user with easy access to selected functions.**  **(eg. Search Book Details, Program Settings, Help, About)**  **Opens Book Detail Window**  **Opens window displaying Terms Of Use conditions**  **Opens Window acknowledging source of data and assistance provided.**  **Provides user with ability to display book details based on title of book** |  |
| Wireframes: A wireframe is a preliminary schematic developed early in the digital product design process to assist visualize and express a product's or website's structure.  A wireframe's objective is to design a skeletal layout that is simple to understand while also encouraging iteration and feedback. Reaching an agreement on a decent interface structure is a vital step in the software design process. The term "fidelity" is commonly used to describe wireframes and other digital product designs. In this sense, fidelity refers to the level of realism of a design artifact. (Guilizzoni, n.d.).  Low fidelity designs are often used for the following reasons:   * They are faster to create because they are less detailed. * The product appearance and/or functionality aren’t fully defined, so many of the details are unknown. * To get reviewers or collaborator to focus on the structure, layout, and navigation instead of visual details.   High fidelity may be used in the following scenarios:   * When the placement of controls and flow from one page/screen to another has been decided. * For usability testing with actual or representative users. * As a final checkpoint before coding begins or the project is approved. | | | | | | | | |  |

A computer screen with a white screen

Description automatically generatedA black circle with white lines

Description automatically generatedA black circle with white lines

Description automatically generatedA black circle with white lines

Description automatically generatedA black circle with white lines

Description automatically generated

A screenshot of a computer

Description automatically generated

SQL Used to create Tables:

Table Standings:

cursor.execute("""CREATE TABLE tblStandings

        (nationality TEXT, standing INTEGER, gamesPlayed INTEGER,

        win INTEGER, draw INTEGER, loss INTEGER, goalsFor INTEGER,

        goalsAgainst INTEGER, goalsDifference STRING, Points INTEGER)

        """)

Table PlayerStats:

cursor.execute("""CREATE TABLE tblPlayerstats

            (playerID INTEGER, playername STRING, playerlastname STRING,

            Appearances STRING, goalscored STRING,

            assitsprovided STRING, Dribblesper90 STRING, Interceptionsper90 STRING,

            Tacklesper90 STRING, TotalDuelsWonper90 STRING, Savepercentage STRING, Cleansheets STRING)

            """)

Table Players:

("""CREATE TABLE tblPlayers

        (nationality TEXT, playerPositionID STRING, jerseyNumber INTEGER, playerBirthDay STRING, playerBirthYear STRING, club STRING, playerID STRING,

        playerFName STRING, playerLName STRING)

        """)

Table Comments

cursor.execute("CREATE TABLE IF NOT EXISTS Comments (id INTEGER PRIMARY KEY AUTOINCREMENT, comment TEXT)")

SQL Used in the admin Panel:

SQL used to delete selected comment:

cursor.execute("DELETE FROM Comments WHERE id=?", (comment\_id,))

SQL used to delete selected user from Users. Db

cursor.execute("DELETE FROM Users WHERE username=?"

A screenshot of a computer login screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

See page 2 for screenshots of the SQL tables.

The SQL for pulling the data from the input fields and either adding it to the users database or seeing if it is a login already.

cursor.execute("INSERT INTO users (username, password, is\_admin) VALUES ('ajman0', 'booksbooksbooks', 1)")

cursor.execute("INSERT INTO users (username, password, is\_admin) VALUES ('a', 'a', 1)")

c.execute('''INSERT INTO users (username, password, is\_admin) VALUES (?, ?, ?)''', (username, password, 0))

The SQL for the search function that allows the user to search for specific data in the treeview:

"SELECT \* FROM tblStandings WHERE nationality LIKE ? OR standing LIKE ?", ('%'+str(thing)+'%', '%'+str(thing)+'%'))

The SQL for adding data from a csv file into one of the 3 treeview is:

cursor.execute(f"INSERT INTO tblStandings ({', '.join(headers)}) VALUES ({', '.join(['?' for \_ in headers])})", row)

**Relevant programming elements for the proposed digital solution: The programming language that will be used is python.**

Tkinter Widgets, this solution utilizes various tkinter widgets that make up the main gui and multiple elements inside of the qui like buttons, images, text and Treeview diagrams. Object orientated Programming, the solution follows an object-orientated design by defineing classes for each of the different frames such as (‘main\_window’, ‘homeFrame’, ‘StandingsFrame’, ‘PlayersFrame’, ‘statsFrame’, ‘adminFrame’). Styling with tkinter ttk, the code for the solution uses ‘ttk’ module for styling the tkinter widgets like there colour, size, and position. File handling, the code uses the ‘csv’ module for reading example.csv files and converting and adding them to an SQLite database file using (‘insert\_csv\_data’) Error handling, the coded solution contains error handling through the use of ‘try-except’ code blocks that catch and handle exceptions that may occur during database operations, file handling and file editing.

**Relationship between data, user interface (UI), and program components:**

In the proposed digital solution, the relationship between the data, user interface and programming elements is symbiotic as in they all need each other to function properly. Together all the elements and components form the backbone of the application. The data, stored in SQLite databases, serves as the foundation, holding crucial information such as FIFA standings, player data, and user comments. The user interface elements, comprising buttons, entry widgets, and treeviews, provide the platform for user interaction, guiding navigation and facilitating data input and display. Program components, embodied in Python classes and functions, orchestrate the interactions between the UI and data layers, implementing the application's logic and behavior. Whether retrieving standings data, submitting comments, or updating user information, these components bridge the gap between user input and data manipulation, ensuring a seamless user experience and the smooth operation of the application.

**Relevant interface components for the proposed digital solution:**

The digital solution takes use of a variety of buttons to offer interactive features and elements. Amongst these buttons are navigation buttons for moving between different frames and pages, such as "Admin Page," "Standings," "Players," and "Stats."   
There are "Submit" and "Update" options for adding remarks or changing information.   
"Search" buttons to start the search process.   
Buttons to pick CSV files for data entry by opening a file explorer.

The proposed solution takes use of labels which are blocks of text that can be positions anywhere on a designated page or frame. In the case of this solution that are used for the titles in all the main pages.

Treeview widgets are an important part of the solution. Treeview widgets are used to display data in a tabular format. These treeview widgets include Tables showing standings, player data, and statistics fetched from the database and Tables displaying comments from users.

**Accessibility, Learnability** – The page is of simple design and all the functioning buttons are labelled with what their function is which allows the user to get the hang of the program quicker.

**Safety** – When the password is inputted all character show up as ‘\*’.

Usability Principles:  
Accessibility – Ability of a system to be utilized by a wide demographic.

Effectiveness – ability of a system to allows users to efficiently perform the required work.

Safety – The ability of a system to mitigate the severity of mistakes.

Utility – the ability of a system to provide the required functionality.

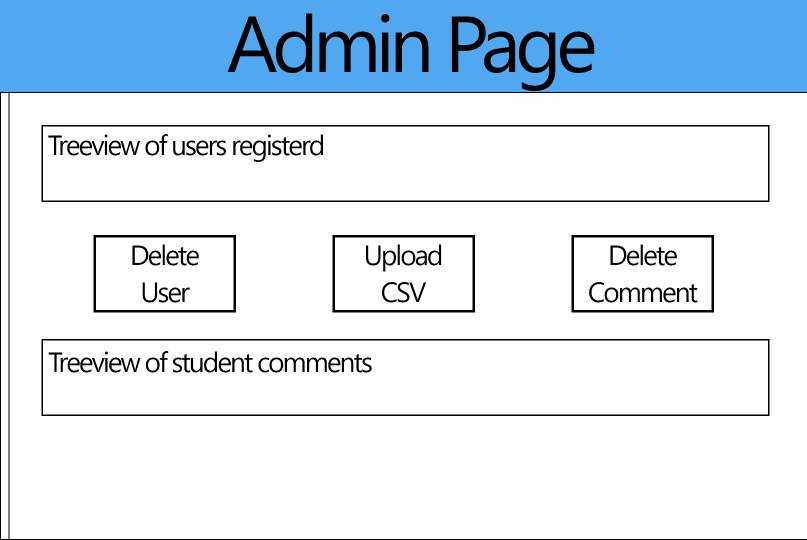
Learnability – a measure of how easy a system it to learn.

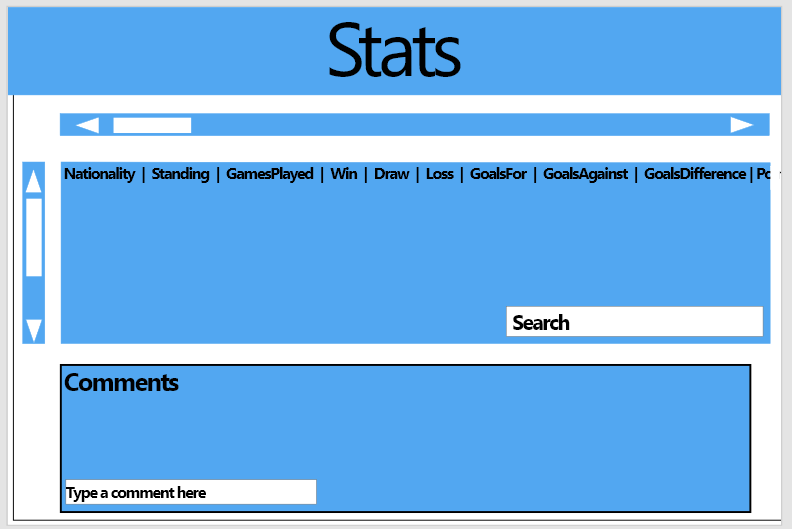
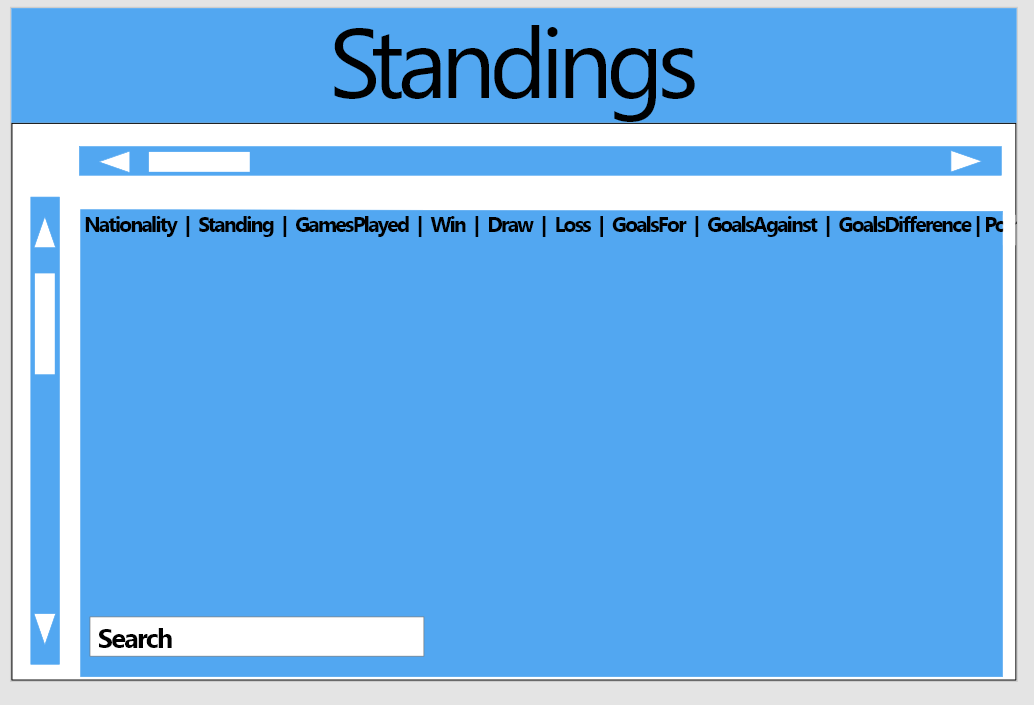
User Interface

CRAP Principles (Williams)

* Proximity - is a way to organize content.
* Alignment - creates the structure and balance of a document.
* Repetition - can be used to create a sense of identity and unity in a document.
* Contrast - contrast is to make certain elements pop and create emphasis.

**Repetition** – The buttons on the side bar repeat a total of 4 times so that all together they can take up the room of the side bar.

A screenshot of a login screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA blue and white rectangular object with black text

Description automatically generated

**Admin Page** users SQLite interfaced code to be able to upload more csv files, delete comments and delete users.

**Contrast** – The background of the page is completely white, and all the important parts are black which creates high contrast for easy use.

Login Window

**Alignment** – All key parts are all aligned to the centre of the page for a more aesthetically pleasing look.

**Data** – This page is the standings page, in this page there is a treeview to see the data that has been pulled from the tblStandings.csv file and transferred into the IA2.db database file. This is the same for the player stats and player’s page.

The search bar will use SQL sceudo code to sort through the data in the treeview.

**Proximity** – for the tree view data display all the titles are spaced apart equally and enough so that it is easy to interpret and read.

**Safety** – There no ability to be able to edit delete or update any data on this page so there is no risk of something being falsely changed.

**Repetition** – There is not much repetition within the page, but this page does repeat a total of 3 times with different types of information in them, this makes it easier for the user to use as once you know how to navigate the first page then they will be familiar with all three pages.

**Interface and coding Components –** The player stats title is a Label an allow the user to understand which page there on, and the tree view is an important part as it is the primary way to interact with the data.

**Learnability / Usability** – The learnability of this page is very high as all the main functions are clearly labelled and all the different methods of input are commonly used between different programs.

**Programing Components** – The title, buttons and index thing text are all tkinter module parts.

**Accessibility** – This page has no apparent accessibility features except for the ability to zoom in so that the text becomes larger and easier to see for the visually impaired.

**Safety** – There is a factor that is concerning that is if you accidentally write a comment and submit it. The page has the ability to delete the comment and has a 10 second comment delay to allow the user to delete the comment before everyone else can see it.

**Learnability / Usability** – The learnability could be better for certain aspects of the page such as to delete a comment the user has to left click the comment for the option to delete which is not directly apparent to the users, another things is that as the scrolls bars are quite small they are very sensitive and may give the user a hard time trying to navigate the page.

**Repetition** – There is not much repetition within the page, but this page does repeat a total of 3 times with different types of information in them, this makes it easier for the user to use as once you know how to navigate the first page then they will be familiar with all three pages.

**Contrast** – The Contrast on this page could be better as the black and blue are not the best contrasting colours, but all components are easily seen still.

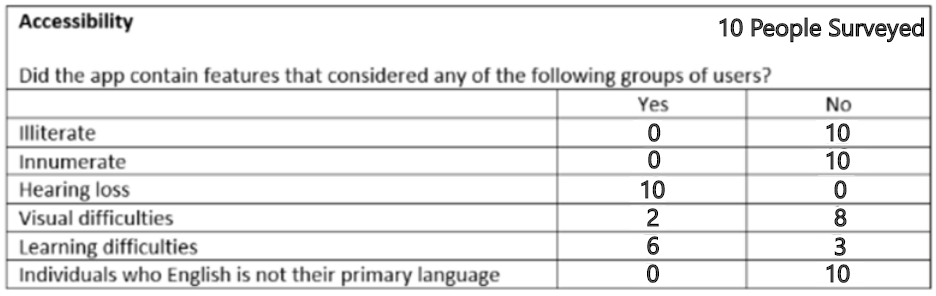
**Proximity** – for the tree view data display all the titles are spaced apart equally and enough so that it is easy to interpret and read.

Main Window Without Index

**Alignment** – The Banner and side buttons are aligned to the top and left of the screen to allow room for the main index panel that will be filled with various windows.

**Contrast** – The white background contrast well with the black text and the blue banner making everything easier to see.

|  |  |  |  |
| --- | --- | --- | --- |
| Element | Prescribed criteria | Judgment | Evaluation |
| Prototype | Task Requirements  Develop an interactive desktop application according to the requirements outlined in the provided technical proposal. Document the problem-solving process and demonstrate the functionality of the components of the digital solution in a video recording.   * Recognise and describe user interface components, usability principles. * Symbolisation Using mind map and wire frames. | All Task requirements  Have been met to a  Satisfactory level. | The proposed solution and program I have created is an interactive application that follows the requirements outlines in the  technical proposal. The Technical proposal states that the solution must have the ability for students and teachers/admin to  interact with FIFA data in ways that will be mentioned in the interface components Evaluation. In this task there is a detailed mind map that was created with *Mind Map Maker* that outlines the data that will be used, the pages in the solution that will be  created, the users that will be interacting with the solution and outlines the basics of user interface components. There is also  recognition of user interface components and usability principles like the CARP principles on page 7. |
|  | Functions Required   * Upload data. * Display data. * Enter data. * Save data. * Store data. | All the functions for  the solution that are  required have been  met to a satisfactory standard. | The coded solution that was created in the programming language python integrates the ability to upload data in the admin panel by uploading a csv file into the IA2.db file, the ability to display data in the standings, players, and stats page through the use  treeview that displays the data in rows separated by label columns, the ability to enter data through the use of the comments  where the user is able to make comments on the data, save data; all of the data that is inputted through either registering, making a comment of uploading as csv file is saved either is IA2.db or tblUsers.db. |
|  | Interface components  Teacher/admin   * Allow the user to access a report indicating information such as the number of registered users and the number of comments completed. * Be able to delete user accounts and comments. * Upload football team and player details.   Students   * View football Teams data * View reviews of teams or players (Based on queries) * Be able to register. * Be able to leave comments under each data treeview |  | In the admin page of my solution allows the admin to delete a specific selected user or comment. There is two treeview that show all the users that have accounts and whether  they are an admin or not, and the admin has the ability to upload a csv file through the  click of the open csv button to the IA2.db database.  All the functions that are required to be in the solution for the teacher/admin and the students have been included and completed.  In the standings, stats and players page the user has the ability to view FIFA data in  a treeview and are able to leave comments but the comments carry over to all of the  three pages which is something that could be changed. Users also have the ability to  register a account that has limited functionality. |
| User  Experience | * Meets Task Requirements (utility) * Complete Required Actions {Effectiveness) * Easy to use. * Responsive * Reliable (no errors) |  | The task that has been created meets all task requirements for utility stated above. It is easy to use as all the functions and buttons are label for a easy to learn user experience and the program is reliable as no errors occur and If the user inputs something wrong there is error handling to stop the application from crashing like in the user\_interface.py file on line 388-390 where there is  error handling for if the admin tries to delete a user but they haven’t selected an account:  The solution meets most task requirements and is easy to use, responsive and reliable.  tk.messagebox.showwarning("Warning", "Please select one or more users to delete.") |
| Coded  components | * Accuracy * Efficiency | The solution is mostly accurate and is efficient with all its functions and coded components. | The coded components in the solution provide an accurate and efficient way to view FIFA related data on Players, Standings and Stats. It is efficient as the code is split into 5 different python files that all have their own functions that all are called on in the User\_interface.py file. “userUtilities.py” has the functions for logging in and signing up. “main\_window” is the main page that you run to start the program that has the base Gui frame functions in it. “data.py” pulls that data from the csv files and imports them  into the IA2.db file and creates the tables if they don’t exist. “data\_display” has the functionality of the treeviews that are  displayed in the standigs, stats, players and admin page, it also holds the search bar functionality. By having the code split up into the different files the coding process becomes more efficient and uniform. |



Personal, Economic, and social impacts:

Personal impacts, the effect of an individual's actions on other people (or other people on them), in an organisation, or in society (Bloch, 2012). Economic impacts, seek to establish the net change in a host economy (INTERNATIONAL ASSOCIATION OF EVENT HOSTS, 2020). Social Impacts, the effect on people and communities that happens because of an action or inaction, an activity, project, programme, or policy (Parrett, 2018).

The solution can have lots of different types of personal, social, and economic impacts. Personally, it gives players a way to interact with content about FIFA, which improves their appreciation and knowledge of the game. It gives users access to statistics, player details, and standings, giving them important facts and insights. Through social media elements like comments, where users may exchange ideas and talk about FIFA-related subjects, the program encourages community connection among fans, encouraging participation and teamwork. Depending on its monetization plan, the program may reasonably contribute to revenue generating through channels like sponsorships, ads, or premium features. Additionally, by boosting interest and involvement in the sport, it might indirectly benefit FIFA-related businesses, such ticketing portals or item sales.

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria \ Questions**  **10 People Tested** | **Response** | | |
| Yes | No | Comments |
| **Reliability**   * **The program opens when executed.** * **All windows open when selected.** * **All windows close when expected.** | 8 | 2 | The program is responsive when opening and executing functions. The program hangs for a second before loading instantly but that is an issue with VScode |
| **Layout / Presentation**   * **Suitable font sizes used?** * **Suitable font type used?** * **Colours used were appropriate.** * **Objects (eg. titles, instructions, questions, input boxes, buttons), were positioned in expected location.** | 6 | 4 | The layout is presentable, and all the font is big enough to read. The main pages all had the default grey background, as well as all the buttons and labels. |
| **Accuracy**   * **Was the data accurate to the data that was stored and inputted.** | 9 | 1 | All the data is taken straight from an official FIFA database. One user had a problem with the import csv example not being proper and real data. |
| **Input Permitted**   * **Did the program allow for flexible data entry (eg. accepted lowercase, upper case and title case)?** | 10 | 0 | The program excepts uppercase and lowercase characters along with integers. |
| **Ease of use**   * **Was the program easy to use?** * **Was it easy to navigate the solution?** * **Was the data clear and usable?** * **Was the layout easy to understand?**   **Third Party Testings**  **Was the data relevant to what needs to be shown?** | 7 | 3 | The program was said to be easy to use as all functions where clearly labelled and where simple to use.  The data in the treeview was said to not be the easiest to read as all the data is just displayed in long, small font lines.  The data was relevant to what needed to be shown as the solution is for FIFA world cup data and that is what was displayed. |

Improvements to be made for next time:

* Higher security, have the function to hash all passwords and block the register function for only the admin to use so that not anyone can just create an account.
* Improve User Experience, this could include redesigning the layout for better usability, adding intuitive navigation features, and providing feedback to users.
* Improve accessibility, Ensure the application is accessible to users with disabilities by adhering to accessibility standards (e.g., WCAG). This involves implementing features such as keyboard navigation, screen reader support, and semantic HTML markup.
* I could make the wireframes more like the final code. The main functionality is the same, but the layout is different.

# Bibliography

Guilizzoni, P. (n.d.). *What are wireframes and why are they used?* Retrieved from balsamiq: https://balsamiq.com/learn/articles/what-are-wireframes/

Lever, T. (2024). *DIS12A Digital Solutions.* Retrieved from Qlearn Cources: https://qlearn.eq.edu.au

Lucidchart. (2023). *Data flow diagram examples, symbols, types, and tips.* Retrieved from Lucidchart: https://www.lucidchart.com/blog/data-flow-diagram-tutorial#:~:text=A%20data%20flow%20diagram%20shows,various%20entities%20and%20their%20relationships.

Oxford Languages. (n.d.). *Dictionary*. Retrieved from Oxford Languages: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiRwPL8yIGFAxUe1TgGHSGMDzMQvecEegQIMBAI&url=https%3A%2F%2Flanguages.oup.com%2Fgoogle-dictionary-en&usg=AOvVaw3kNBXVjbAIAFeyFQVCJmJF&opi=89978449

Williams, R. (n.d.). *The Non-Designers Design Book .* Writing Centre @ lewlsu.

# Bibliography

Bloch, B. (2012). *Differences between personal, social and institutional impact*. Retrieved from Future Learn: https://www.futurelearn.com/info/courses/measuring-entrepreneurial-impact/0/steps/55997#:~:text=Personal%20impact%20is%20about%20the,or%20negative%20impact%20on%20others.

Guilizzoni, P. (n.d.). *What are wireframes and why are they used?* Retrieved from balsamiq: https://balsamiq.com/learn/articles/what-are-wireframes/

INTERNATIONAL ASSOCIATION OF EVENT HOSTS. (2020). *Economic Impacts*. Retrieved from INTERNATIONAL ASSOCIATION OF EVENT HOSTS: https://www.eventhosts.org/resources/event-impact-standards/economic-impacts-2/#:~:text=Economic%20Impact%20studies%20seek%20to,establish%20the%20resulting%20net%20outcome.

Lever, T. (2024). *DIS12A Digital Solutions.* Retrieved from Qlearn Cources: https://qlearn.eq.edu.au

Lucidchart. (2023). *Data flow diagram examples, symbols, types, and tips.* Retrieved from Lucidchart: https://www.lucidchart.com/blog/data-flow-diagram-tutorial#:~:text=A%20data%20flow%20diagram%20shows,various%20entities%20and%20their%20relationships.

Oxford Languages. (n.d.). *Dictionary*. Retrieved from Oxford Languages: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiRwPL8yIGFAxUe1TgGHSGMDzMQvecEegQIMBAI&url=https%3A%2F%2Flanguages.oup.com%2Fgoogle-dictionary-en&usg=AOvVaw3kNBXVjbAIAFeyFQVCJmJF&opi=89978449

Parrett, E. (2018). *What is Social Impact and How Do I Measure It?* Retrieved from Good Finance: https://www.goodfinance.org.uk/latest/post/what-social-impact-and-how-do-i-measure-it#:~:text=Social%20impact%20can%20be%20defined,work%20with%20and%20buy%20from.

Williams, R. (n.d.). *The Non-Designers Design Book .* Writing Centre @ lewlsu.