

**Enjabulweni Independent School  
Grade 12  
Banzile Z. Nhlebela  
IT PAT  
Project Design Document**

Contents

[User Interface Design 4](#_Toc83863992)

[**Login Screen** 4](#_Toc83863993)

[**Description** 4](#_Toc83863994)

[**Data** 4](#_Toc83863995)

[**Actions** 4](#_Toc83863996)

[**Create New Account Screen** 6](#_Toc83863997)

[**Description** 6](#_Toc83863998)

[**Data** 7](#_Toc83863999)

[**Actions** 7](#_Toc83864000)

[**Forgot Password Screen** 7](#_Toc83864001)

[**Description** 7](#_Toc83864002)

[**Data** 8](#_Toc83864003)

[**Actions** 8](#_Toc83864004)

[**Help Screen** 9](#_Toc83864005)

[**Description** 9](#_Toc83864006)

[**Data** 9](#_Toc83864007)

[**Actions** 9](#_Toc83864008)

[**Main Menu Screen** 11](#_Toc83864009)

[**Description** 11](#_Toc83864010)

[**Data** 11](#_Toc83864011)

[**Actions** 11](#_Toc83864012)

[**Student Log Screen** 13](#_Toc83864013)

[**Description** 13](#_Toc83864014)

[**Data** 13](#_Toc83864015)

[**Actions** 13](#_Toc83864016)

[**Student Offence Register Screen** 15](#_Toc83864017)

[**Description** 15](#_Toc83864018)

[**Data** 15](#_Toc83864019)

[**Actions** 15](#_Toc83864020)

[**Offence Log Screen** 17](#_Toc83864021)

[**Description** 17](#_Toc83864022)

[**Data** 17](#_Toc83864023)

[**Actions** 17](#_Toc83864024)

[**Parent Log** 19](#_Toc83864025)

[**Description** 19](#_Toc83864026)

[**Data** 19](#_Toc83864027)

[**Actions** 19](#_Toc83864028)

[**Edit Profile Screen** 21](#_Toc83864029)

[**Description** 21](#_Toc83864030)

[**Data** 21](#_Toc83864031)

[**Actions** 21](#_Toc83864032)

[Sequencing and Program Flow 23](#_Toc83864033)

[Class Diagrams Design 42](#_Toc83864034)

[Secondary Storage Design 43](#_Toc83864035)

[**User Data** 43](#_Toc83864036)

[**Help Data** 43](#_Toc83864037)

[**Security Data** 43](#_Toc83864038)

[**Student Data** 44](#_Toc83864039)

[**Student Offence Data** 44](#_Toc83864040)

[**Parent Data** 45](#_Toc83864041)

[**Offence Data** 45](#_Toc83864042)

[**Temporary User Data** 46](#_Toc83864043)

[Explanation of Secondary Storage Design 46](#_Toc83864044)

[Table Relationships 47](#_Toc83864045)

[Explanation/Description of Primary Data Structure 47](#_Toc83864046)

[**Student Class** 47](#_Toc83864047)

[**Cons Class** 47](#_Toc83864048)

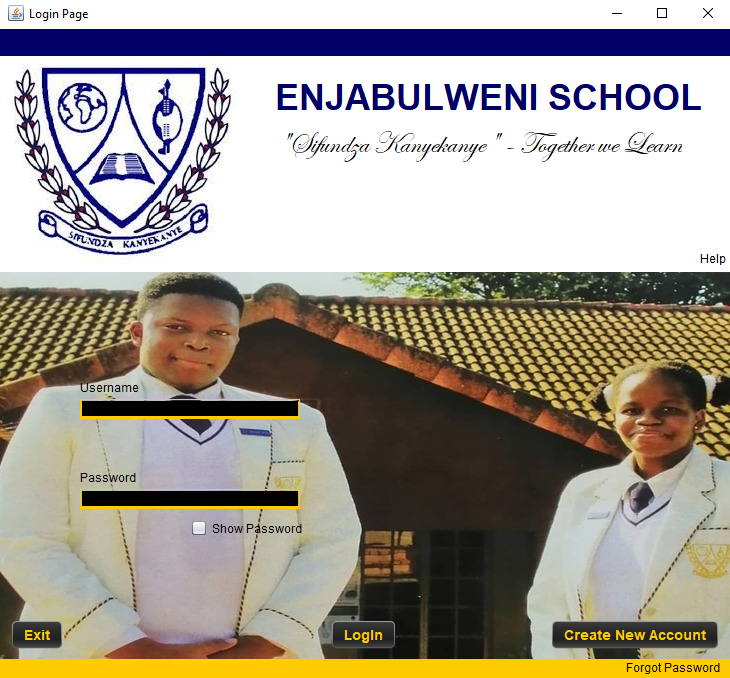
[**Login Class** 48](#_Toc83864049)

[**Parents Class** 48](#_Toc83864050)

[**Offence Class** 48](#_Toc83864051)

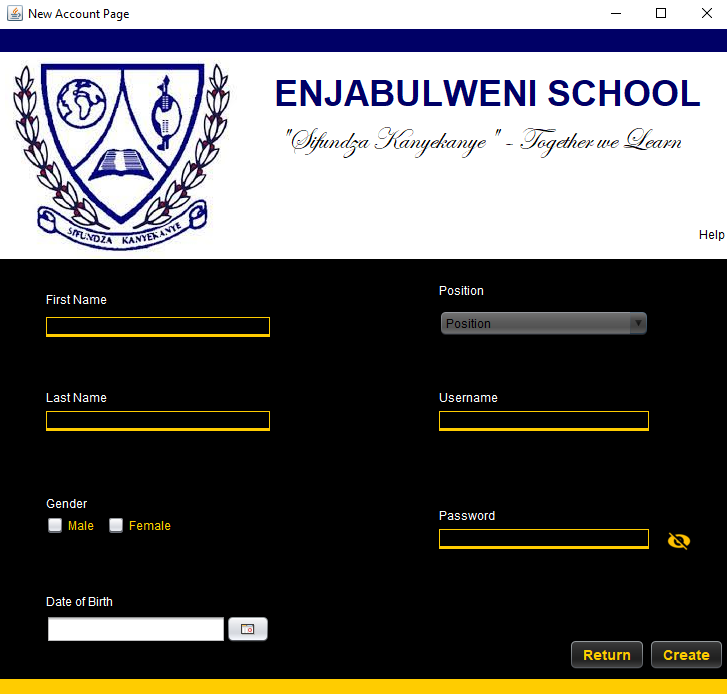
# **User Interface Design**

## **Login Screen**



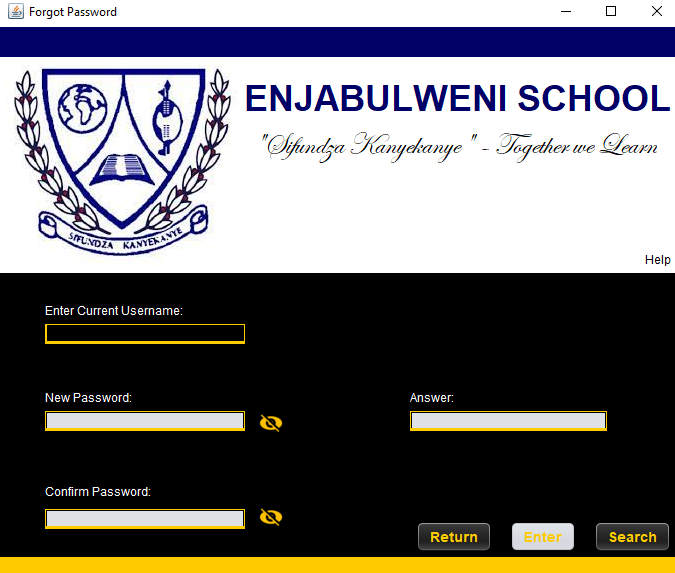
|  |  |
| --- | --- |
| **Description:** | This is the first screen that pops up at the beginning of the program; It is the first screen that the user can interact with. It allows the user to login using the “Login” button and it also allows the user to create a new account using the “create new account” button. The “Help” screen is accessed using the “Help” label and if the user already has an account and they have forgotten their password, they can access their password using the “Forgot Password” label |
| **Data:** | The user’s login details i.e., Username and Password |
| **Actions:** | **Login**   * The program searches the database checking whether the users input in the text fields correspond with the data in the database. If the input matches and is correct, the login screen will close and the user will proceed to the main menu screen. If the input does not match, red stars will appear under the both text fields and a suitable message will pop up. Appropriate messages will be displayed if the input is empty.   **Create New Account**   * The program will close the login screen and display a new screen that will allow the user to enter the details of the account that they wish to create. *More on this screen later in the document.*   **Help**   * The Help screen will pop up and the login screen will not close. *More on this screen further on in the document.*   **Forgot Password**   * The program will close the login page and open a new screen that will allow the user to access their password if they have forgotten it. *More on this screen further in the document.*   **Exit**   * The program will close upon the interaction with the “Exit” button.   **Show Password**   * This checkbox reveals the user’s password when it is checked and conceals it when it is unchecked |

## **Create New Account Screen**



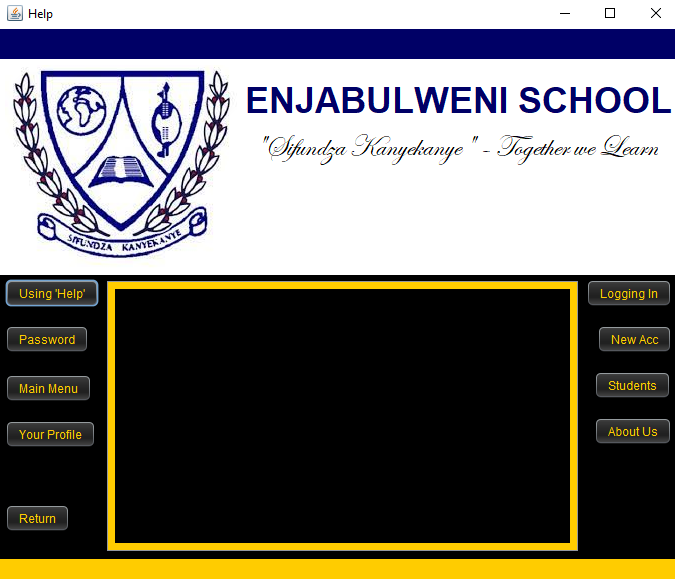
|  |  |
| --- | --- |
| **Description:** | This is the “New Account” page that is used for the purposes of creating a new account. There are text fields, check boxes, a date picker and a combo box that will be used to capture the user’s account details. When the “Create” button is pressed, the program will search the database and check if the username that has been entered already exists and if that is the case, an appropriate message will pop up informing the user. If that is not the case then the program will validate all the fields that have been entered. If there is an error with any of the fields, a message in red will appear under the structure used to capture the data and if there are no errors, green stars will appear under the fields and an appropriate message will pop up, informing the user that the account has been created. The page will then be closed by the program and the login page will be reopened. |
| **Data:** | The user’s personal details as well as the username and password that will be used for the account. |
| **Actions:** | **Create**   * The account is created granted that all the fields have been validated and approved by the program. The user will know all fields are acceptable when green stars appear under all the fields.   **Return**   * The program will close the current screen and return to the login page. |

## **Forgot Password Screen**

****

|  |  |
| --- | --- |
| **Description:** | This is the screen that will appear when the “Forgot Password” label is clicked on the login page. There are four text fields and three buttons present in this form. The user will enter their current username and click ‘search’. The program will then search the database for their username and if it is found, appropriate messages will be displayed and the user will be asked to answer their security question. Once the user provides and answer, the program will compare it to the answer present and if it is incorrect, the question is displayed again and appropriate messages are displayed informing the user. Once a correct answer is entered, the user will then be allowed to enter their new password and change their old one. After the use of the search button, username and answer field, said items are disabled. The enter button, the password fields and the answer field are disabled upon the appearance of the screen |
| **Data:** | This page accepts the current username of the user, the answer to their security question and their new password |
| **Actions:** | **Search**   * The program searches the database for the username entered and its corresponding security question and answer. Once found, the program will ask the user for the answer to their security question that will be displayed   **Enter**   * Once the answer to the security question is entered, the user clicks this button and the program then checks the answer. If the answer is correct, the user can then enter their new password and click the button again to enter their new password into the database   **Return**   * The screen will be closed and the login page will be reopened. |

## **Help Screen**

****

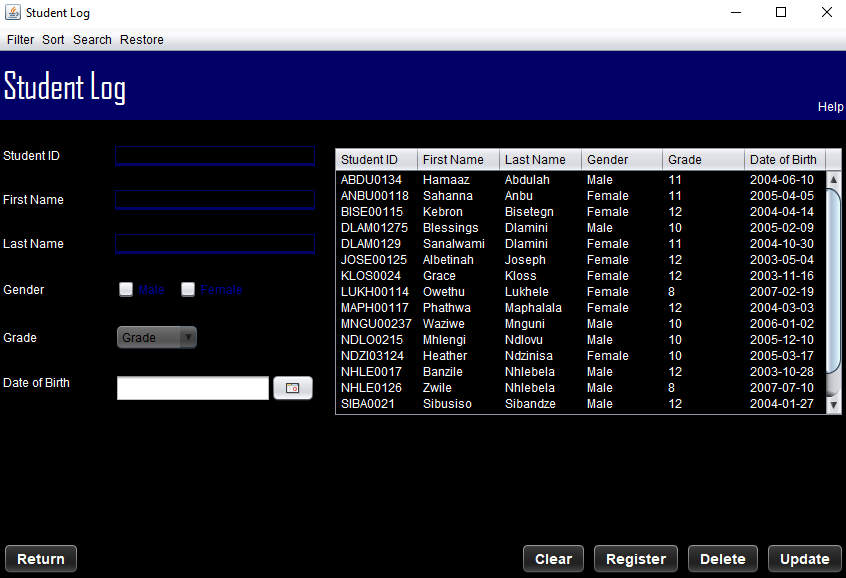
|  |  |
| --- | --- |
| **Description:** | This is the screen that will appear once the ‘Help’ label or button is clicked. Said labels are found on all screens except the ‘Main Menu’ screen as there is a ‘Help’ button present. This screen has nine buttons in total. Eight of these buttons will be used to access the different topics they represent. Once one of the buttons is clicked, text will appear in the text area surrounded by a yellow border. |
| **Data:** | This page does not accept any data |
| **Actions:** | **Using Help**   * Once clicked, this button will prompt the display of text explaining how to use the help screen and how to access it, in the text area.   **Password**   * Once clicked, this button will prompt the display of text explaining how to obtain a password when forgotten and how to access the screen that will allow the user to perform this action, in the text area.   **Main Menu**   * Once clicked, this button will prompt the display of text explaining the functions of the Main Menu Screen, in the text area.   **Your Profile**   * Once clicked, this button will prompt the display of text explaining how to use the help screen and how to access it, in the text area.   **Students**   * Once clicked, this button will prompt the display of text explaining how to use the Student Log, Parent Log, Offence Log and Student Offence Register screens, their functions and how to access them, in the text area.   **New Acc**   * Once clicked, this button will prompt the display of text explaining how to create a new account and how to access the screen that will allow the user to perform this action and its functions, in the text area.   **Logging In**   * Once clicked, this button will prompt the display of text explaining the login page’s functions, in the text area.   **About Us**   * Once clicked, this button will prompt the display of text that will give a detailed explanation of the school, in the text area.   **Return**   * Once clicked, this button will close the help screen. |

## **Main Menu Screen**

****

|  |  |
| --- | --- |
| **Description:** | This is the page that will appear after the user has successfully logged in. This page has a total of seven buttons. Six of said buttons lead to different screens. Once clicked, the current screen will close and the corresponding screen will appear. |
| **Data:** | This page does not accept any data. |
| **Actions:** | **Student Log**   * Once clicked, the current screen will close and the Student Log Screen will open. *More on this screen further in the document.*   **Offence Reg**   * Once clicked, the current screen will close and the Student Offence Register Screen will open. *More on this screen further in the document.*   **Edit Profile**   * Once clicked, the current screen will close and the Edit Profile Screen will open. *More on this screen further in the document.*   **Help**   * Once clicked, the current screen will close and the Help Screen will open. *More on this screen further in the document.*   **Parent Log**   * Once clicked, the current screen will close and the parent log screen will open. *More on this screen further in the document.*   **Offence**   * Once clicked, the current screen will close and the offence log screen will open. *More on this screen further in the document.*   **Log Out**   * Once clicked, the user will be asked if they want to log out. If the user indeed wishes to log out, the current screen will close and the Login Screen will open. |

## **Student Log Screen**

****

|  |  |
| --- | --- |
| **Description:** | This is the screen that pops up when the ‘Student Log’ button is clicked in the Main Menu screen. It accepts all student data via text fields, check boxes, a combo box and a date picker. Once the register button is clicked and all the data is acceptable, the input data will then be displayed on the table. The user can click on a row and edit the data as well as delete the data from the database. Certain features will be disabled according to the position of the user |
| **Data:** | This screen accepts all student data. |
| **Actions:** | **Register**   * Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the data will be inserted into the database and will be displayed in the table on the screen. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Update**   * A record in the table must be selected. Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the update will be executed and will be reflected on the table. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Delete**   * A record in the table must be selected. Upon clicking this button, the program will delete the selected record from the database and this action will be reflected on the table as well.   **Clear**   * Upon clicking this button, the program will reset all the fields on the screen.   **Filter**   * The user will be given different options that can be used to display a specific portion of the data. For example, the user could want to display grade 8s only or display girls only   **Search**   * The user will be able to search for a student(s) using either their first or last name. The user will be asked to enter the desired value and the program will display the results   **Sort**   * The user will be able to sort the data either in ascending or descending order either by the Student ID or the Grade   **Restore**   * Displays all data from the database without any tampering and also any updates that may have been executed by the user.   **Return**   * The program will close this screen and reopen the Main Menu screen. |

## **Student Offence Register Screen**

****

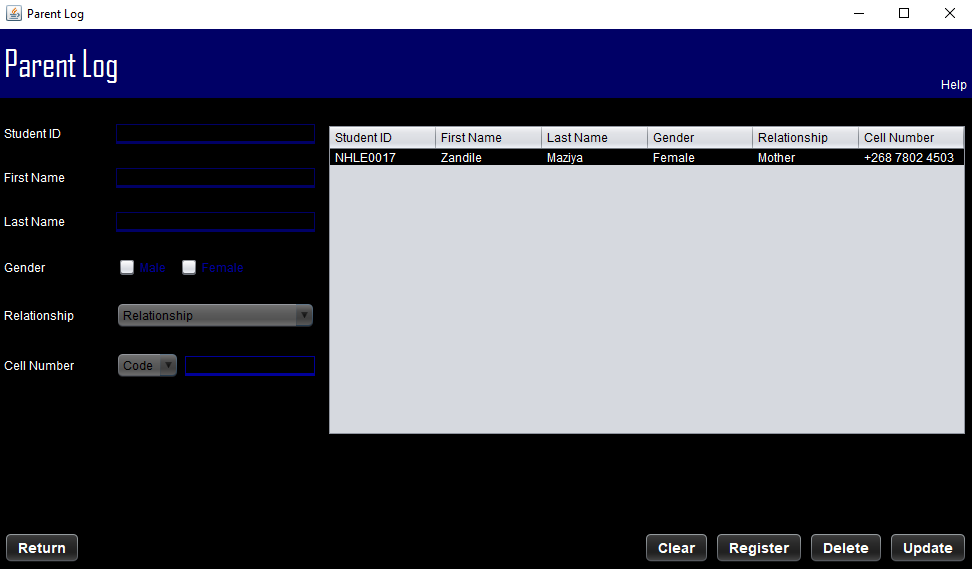
|  |  |
| --- | --- |
| **Description:** | This is the screen that pops up when the ‘Offence Reg’ button is clicked in the Main Menu screen. It accepts all student offence data via text fields, check boxes, a combo box and a date picker. Once the register button is clicked and all the data is acceptable, the input data will then be displayed on the table. The user can click on a row and edit the data as well as delete the data from the database. Certain features will be disabled or enabled according to the position of the user |
| **Data:** | This screen accepts all student offence data |
| **Actions:** | **Register**   * Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the data will be inserted into the database and will be displayed in the table on the screen. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Update**   * A record in the table must be selected. Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the update will be executed and will be reflected on the table. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Delete**   * A record in the table must be selected. Upon clicking this button, the program will delete the selected record from the database and this action will be reflected on the table as well.   **Clear**   * Upon clicking this button, the program will reset all the fields on the screen   **Filter**   * The user will be given different options that can be used to display a specific portion of the data. For example, the user could want to display those who have committed level 1 offences.   **Sort**   * The user will be able to sort the data either in ascending or descending order by the sanction, subject, student ID etc.   **Restore**   * Displays all data from the database without any tampering   **Return**   * The program will close this screen and reopen the Main Menu screen. |

## **Offence Log Screen**



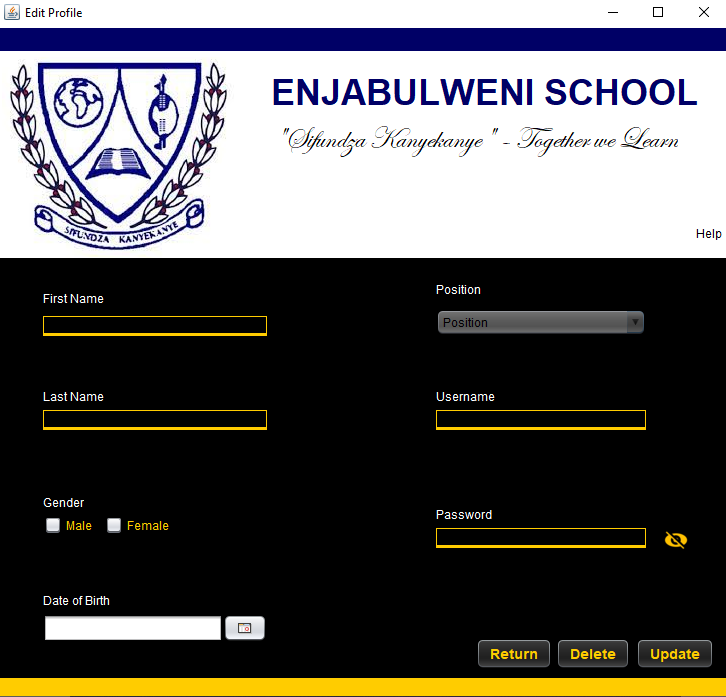
|  |  |
| --- | --- |
| **Description:** | This is the screen that will pop up when the ‘Offences’ button is clicked. All offence details will be displayed when the screen pops up. The user can then edit these details and they will be changed in the database as well. The user can also delete or add an offence all together. |
| **Data:** | This screen accepts the Offence ID, Description and Level of offence. |
| **Actions:** | **Register**   * Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the data will be inserted into the database and will be displayed in the table on the screen. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Update**   * A record in the table must be selected. Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the update will be executed and will be reflected on the table. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Delete**   * A record in the table must be selected. Upon clicking this button, the program will delete the selected record from the database and this action will be reflected on the table as well.   **Clear**   * Upon clicking this button, the program will reset all the fields on the screen.   **Filter**   * The user will be given different options that can be used to display a specific portion of the data. For example, the user could want to display level 1 offences only.   **Sort**   * The user will be able to sort the data either in ascending or descending order by the level of offence and offence ID   **Restore**   * Displays all data from the database without any tampering |

## **Parent Log**



|  |  |
| --- | --- |
| **Description:** | This is the screen that pops up when the ‘Parent Log’ button is clicked in the Main Menu screen. It accepts all student parent data via text fields, check boxes and combo boxes. Once the register button is clicked and all the data is acceptable, the input data will then be displayed on the table. The user can click on a row and edit the data as well as delete the data from the database. Certain features will be disabled or enabled according to the position of the user |
| **Data:** | This screen accepts all student parent data |
| **Actions:** | **Register**   * Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the data will be inserted into the database and will be displayed in the table on the screen. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Update**   * A record in the table must be selected. Upon clicking this button, the program will validate all the data in the fields. If all the data is acceptable, green stars will appear under all fields and the update will be executed and will be reflected on the table. If not all the data is acceptable, an appropriate message will appear under the respective field in red.   **Delete**   * A record in the table must be selected. Upon clicking this button, the program will delete the selected record from the database and this action will be reflected on the table as well.   **Clear**   * Upon clicking this button, the program will reset all the fields on the screen   **Return**   * The program will close this screen and reopen the Main Menu screen. |

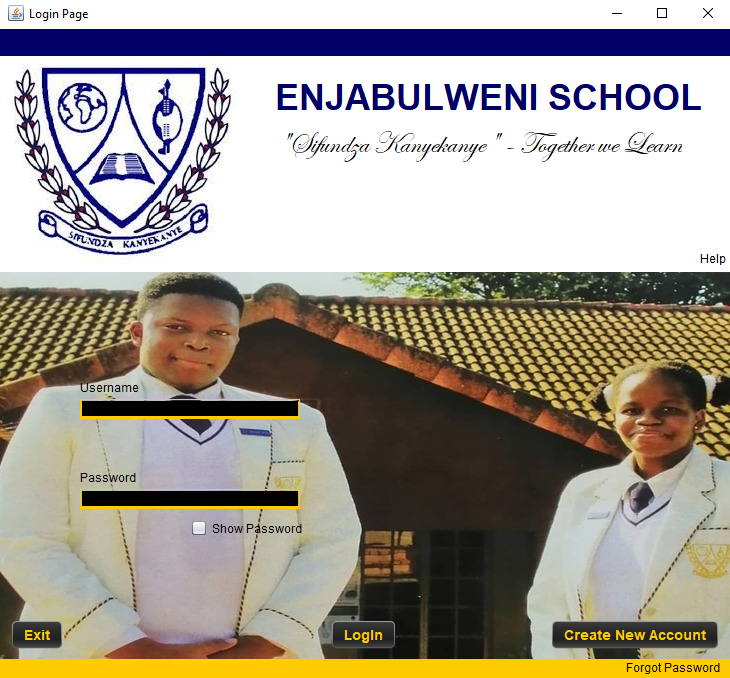
## **Edit Profile Screen**

****

|  |  |
| --- | --- |
| **Description:** | This is the screen that will pop up when the ‘Edit Profile’ button is clicked. All the user’s details will be displayed when the screen pops up. The user can then edit their details and they will be changed in the database as well. The user can also delete their profile all together. |
| **Data:** | This screen accepts the username of the user and allows the user to edit current data belonging to them. |
| **Actions:** | **Update**   * This will update the user information once edited by the user. If there is any data that is not valid/acceptable, appropriate messages will be displayed informing the user of the issue.   **Delete**   * This will delete the user profile as a whole. This action will also be reflected in the database and the program will then close the current screen and open the login page.   **Return**   * This will close the current screen and reopen the Main Menu screen. |

# **Sequencing and Program Flow**

* The Login Screen is displayed



* The user types in both their Username and Password:
  + The user clicks on the ‘Login’ button:
    - The program searches the ‘accountdetails’ table:

Un 🡨 input

Pass 🡨 input  
Begin if

If un is empty

Display “enter username”

Label1 🡨 enter username

Else

Display “\*\*\*\*”

End if

Begin if

If pass is empty

Display “enter password”

Else

Display “\*\*\*\*”

End if

Begin if

If data is valid

Get connection to database

Search for un and pass in table

Begin if

If un and pass are found

Display welcome message

Close screen

Open main menu screen

Else

Display “Password or username incorrect”

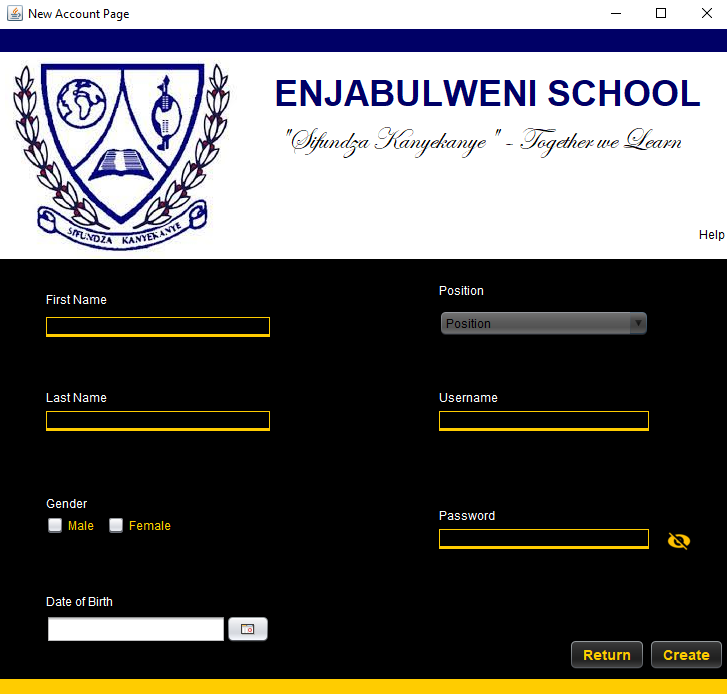
End if

Close connection to database

End if

* Or the user wishes to Create a New Account:
  + The user clicks the “Create New Account” button:
    - Close current screen

Open New Account screen



* + The user inputs their details and clicks “Create”:
    - Validate all the fields

Display appropriate messages under each field

Get connection to database

Search table for username

Begin if

If username does not exist

Display “Account created successfully”

Close current screen

Reopen Login screen

Else

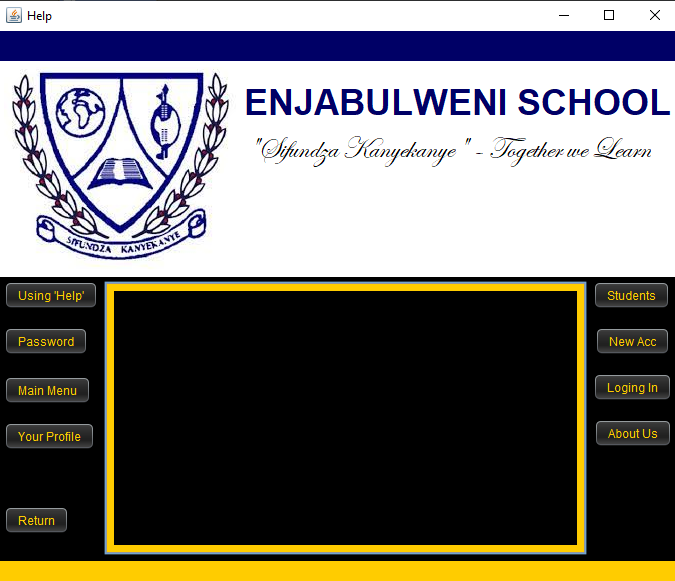
Display “Username already exists”

Clear username field

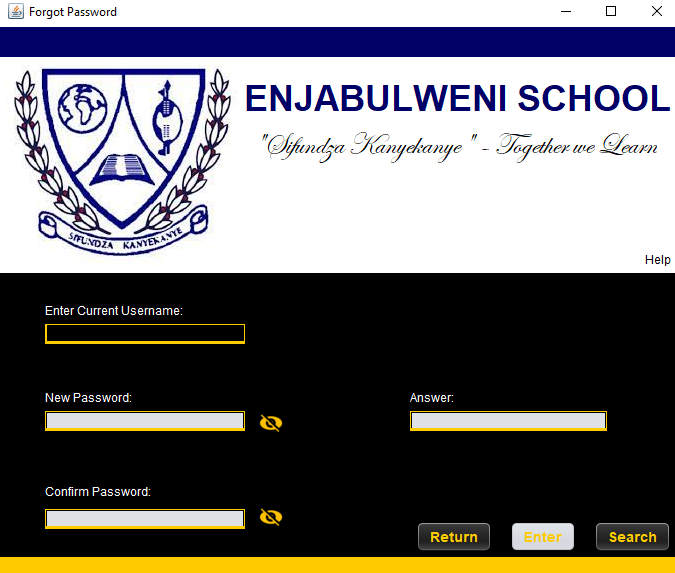
End if

Close connection

* Or the user wants to access the ‘Help’ page
  + The user clicks the “Help” label:
    - Display Help page



* + The user clicks one of the 8 available buttons:
    - Display corresponding text in text area
  + The user clicks the return button:
    - Close the current page
* Or the user has forgotten their account
  + The user clicks the “Forgot Password” label:
    - The program displays the ‘Forgot password’ screen:



The user enters their username and clicks ‘search’:

Get connection to database

Search table for username

Begin if

If username exists

Display “Username found. Answer security question”

Disable username field

Disable Search button

Display security question

User enters answer and clicks enter:

Begin if

If answer is correct

Disable answer field

Enable password fields

Display “Correct Answer. Enter new password”

User enters password in password fields:

Begin if

If passwords match

Display “Password changed successfully”

Else

Display “Ensure passwords match”

End if

Else

Display “Incorrect Answer”

Display Security question

End if

Else

Display “Username not found”

End if

Close connection

* + The user clicks the ‘return’ button:
    - Close current screen

Reopen login screen

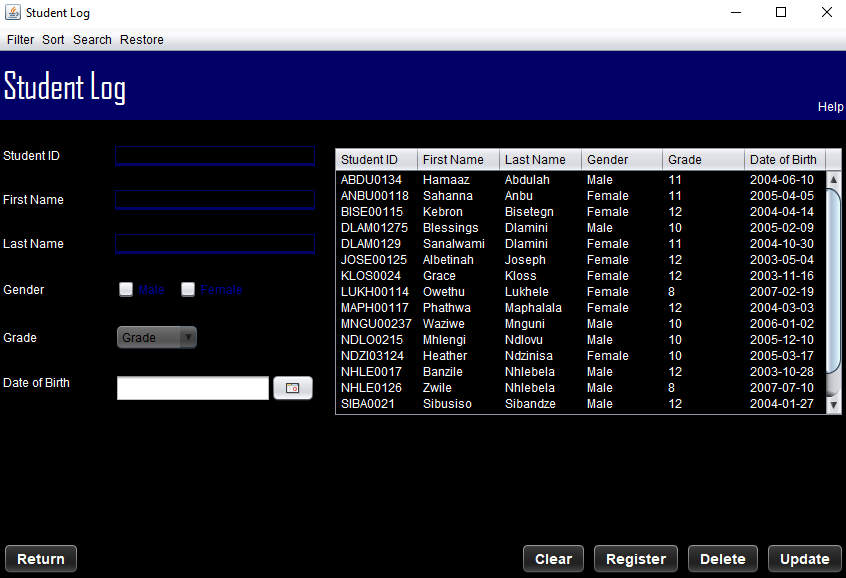
* After successful login, Main Menu screen is displayed:



* The user clicks one of the 7 available buttons:
  + The user clicks the ‘Student Log’ button:

Close current screen

Display student log screen



* + Begin if:

If user is Head Prefect or Prefect

Disable Register and Delete buttons

Disable Student ID, Gender and Grade Fields

Else

Leave all enabled

End if

* The user enters student details:
  + The user clicks ‘Register’:

Validate all fields

Begin if

if data is valid

display green stars under the fields  
 get connection to database  
 input data into table

display data on the table

clear all fields

display “Registered successfully”

else

display appropriate message indicating error

display message under field in red informing user of error

end if

* The user selects a row on the table:
  + The user clicks ‘delete’:

Display a message asking for confirmation

Begin if

If confirmation is given

Set foreground of labels and fields to red

Get connection to database

Delete data from table

Reflect action on table of the screen

Clear all data from fields

Else

Display “Delete action aborted”

Set foreground of fields and labels to blue

Clear all data from fields

End if

* + The user changes data in the fields and clicks ‘update’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Validate data

Begin if

If all data is valid

Display green stars under all fields

Execute update in database

Reflect action on table of the screen

Clear all data from fields

Else

Display appropriate messages in red.

End if

Else

Display “Update action aborted”

Clear all data from fields

End if

* + - The user clicks ‘Filter’
      * The user clicks ‘Gender’
        + The user clicks ‘Male’:

Display all male students on table

* + - * + The user clicks ‘Female’:

Display all female students on table

* + - * The user clicks ‘Grade’
        + The user clicks ‘8’:

Display all grade 8 students on table

* + - * + The user clicks ‘9’:

Display all grade 9 students on table

* + - * + The user clicks ‘10’:

Display all grade 10 students on the table

* + - * + The user clicks ‘11’:

Display all grade 11 students on the table

* + - * + The user clicks ‘12’:

Display all grade 12 students on the table

* + - The user clicks ‘Sort’
      * The user clicks ‘Ascending Order’
        + The user clicks ‘Student ID’:

Sort all data according to Student ID in Ascending order

* + - * + The user clicks ‘Grade’:

Sort all data according to Grade in Ascending order (8 - 12)

* + - * The user clicks ‘Descending order’
        + The user clicks ‘Student ID’:

Sort all data according to Student ID in Ascending order

* + - * + The user clicks ‘Grade’:

Sort all data according to Grade in Ascending order (8 - 12)

* + - The user clicks ‘Search’:
      * The user clicks ‘First Name’
        + Display input dialog asking for first name user would like to search for

Display results of search on table

* + - * The user clicks ‘Last Name’
        + Display input dialog asking for last name user would like to search for

Display results of search on table

* + - The user clicks ‘Restore’
      * Display all data on the table
* The user clicks ‘Clear’:
  + Set all data fields to null
* The user clicks ‘Return’:
  + Close current screen

Reopen Main Menu screen

* Or the user clicks the ‘Offence Reg’ button in the main menu:
  + The ‘Student Offence Register’ screen is displayed:



* + Begin if:

If user is Head of School or Deputy Head of School

Disable Register and Update Buttons

Else

Leave buttons enabled

End if

* The user enters the student offence details:
  + The user clicks ‘Register’:

Validate all fields

Begin if

if data is valid

display green stars under the fields  
 get connection to database  
 input data into table

display data on the table

clear all fields

display “Data input successfully”

else

display appropriate message

display message under field in red informing user of error

end if

* The user selects a row on the table:
  + The user clicks ‘delete’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Delete data from table

Reflect action on table of the screen

Clear all data from fields

Else

Display “Delete action aborted”

Clear all data from fields

End if

* + The user changes data in the fields and clicks ‘update’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Validate data

Begin if

If all data is valid

Display green stars under all fields

Execute update in database

Reflect action on table of the screen

Clear all data from fields

Else

Display appropriate messages in red.

End if

Else

Display “Update action aborted”

Clear all data from fields

End if

* + - The user clicks ‘Filter’
      * The user clicks ‘Offence ID’
        + The user clicks ‘Level 1’:

Display all data where student has committed Level 1 offences

* + - * + The user clicks ‘Level 2’:

Display all data where student has committed Level 2 offences

* + - * + The user clicks ‘Level 3’:

Display all data where student has committed Level 3 offences

* + The user clicks ‘Level 4’:

Display all data where student has committed Level 4 offences

* + - * The user clicks ‘Sanction’
        + The user clicks ‘None’:

Display all data where sanction has not been given

* + - * + The user clicks ‘Verbal Reprimand’:

Display all data where student was given verbal reprimand

* + - * + The user clicks ‘Awarded Zero’:

Display all data where student was awarded zero

* + - * + The user clicks ‘Break Detention’:

Display all data where student was given break detention

* + - * + The user clicks ‘Friday Detention’:

Display all data where student was given Friday detention

* + - * + The user clicks ‘Written Warning’:

Display all data where student was given written warning

* + - * + The user clicks ‘Hearing’:

Display all data where a hearing for the student was held

* + - * + The user clicks ‘Suspension’:

Display all data where student was suspended.

* + - * + The user clicks ‘Expulsion’:

Display all data where student was expelled.

* + - * The user clicks ‘Parents Told’
        + The user clicks ‘No’:

Display all data where parents have not been told of offence

* + - * + The user clicks ‘Yes’:

Display all data where parents have been told of offence

* + - * The user clicks ‘Student ID’
        + The user is asked for the student ID they would like to use as a filter

Display all data using specified student ID

* + - The user clicks ‘Sort’
      * The user clicks ‘Student ID’
        + The user clicks ‘Ascending order’:

Sort data according to student ID in ascending order

* + - * + The user clicks ‘Descending order’:

Sort data according to student ID in descending order

* + - * The user clicks ‘Offence ID’
        + The user clicks ‘Ascending order’:

Sort data according to offence ID in ascending order

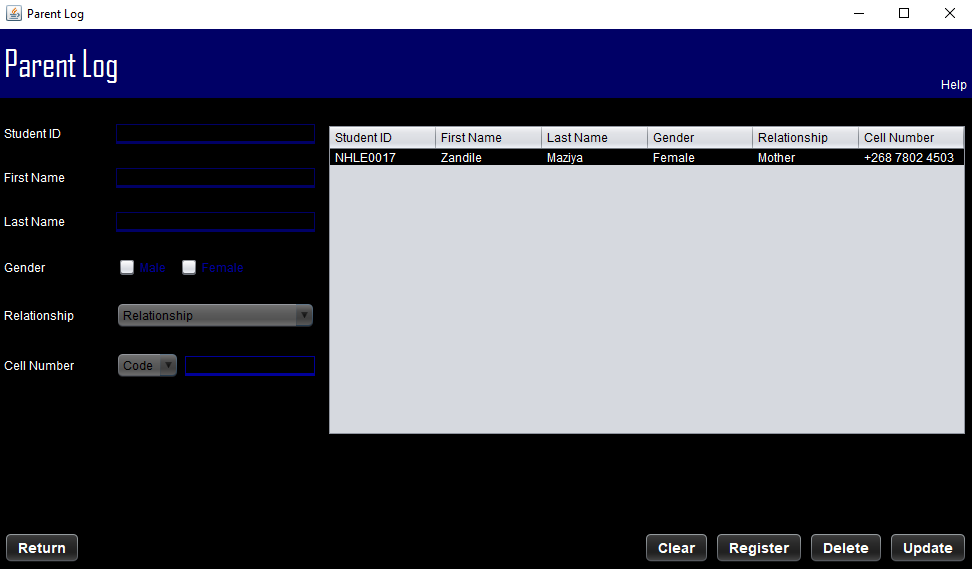
* + - * + The user clicks ‘Descending order’:

Sort data according to offence ID in descending order

* + - * The user clicks ‘Sanction’
        + Sort all data by sanction in ascending order
      * The user clicks ‘Subject’
        + Sort all data by subject in ascending order
      * The user clicks ‘Parents Told’
        + Sort all data by parents told in descending order
    - The user clicks ‘Restore’
      * Display all data on the table
* The user clicks ‘Clear’:
  + Set all data fields to null
* The user clicks ‘Return’:
  + Close current screen

Reopen Main Menu screen

* Or the user clicks the ‘Parent Log’ button in the main menu:
  + The ‘Parent Log’ screen is displayed:



* The user enters the student parent details:
  + The user clicks ‘Register’:

Validate all fields

Begin if

if data is valid

display green stars under the fields  
 get connection to database  
 input data into table

display data on the table

clear all fields

display “Data input successfully”

else

display appropriate message

display message under field in red informing user of error

end if

* The user selects a row on the table:
  + The user clicks ‘delete’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Delete data from table

Reflect action on table of the screen

Clear all data from fields

Else

Display “Delete action aborted”

Clear all data from fields

End if

* + The user changes data in the fields and clicks ‘update’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Validate data

Begin if

If all data is valid

Display green stars under all fields

Execute update in database

Reflect action on table of the screen

Clear all data from fields

Else

Display appropriate messages in red.

End if

Else

Display “Update action aborted”

Clear all data from fields

End if

* The user clicks ‘Clear’:
  + Set all data fields to null
* The user clicks ‘Return’:
  + Close current screen  
    Reopen Main Menu screen
* Or the user clicks the ‘Offences’ button in the main menu:
  + The ‘Offence’ screen is displayed:



* The user enters the offence details:
  + The user clicks ‘Register’:

Validate all fields

Begin if

if data is valid

display green stars under the fields  
 get connection to database  
 input data into table

display data on the table

clear all fields

display “Data input successfully”

else

display appropriate message

display message under field in red informing user of error

end if

* The user selects a row on the table:
  + The user clicks ‘delete’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Delete data from table

Reflect action on table of the screen

Clear all data from fields

Else

Display “Delete action aborted”

Clear all data from fields

End if

* + The user changes data in the fields and clicks ‘update’:

Display a message asking for confirmation

Begin if

If confirmation is given

Get connection to database

Validate data

Begin if

If all data is valid

Display green stars under all fields

Execute update in database

Reflect action on table of the screen

Clear all data from fields

Else

Display appropriate messages in red.

End if

Else

Display “Update action aborted”

Clear all data from fields

End if

* + - The user clicks ‘Filter’
      * The user clicks ‘Offence ID’
        + The user clicks ‘Level 1’:

Display all Level 1 offences

* + - * + The user clicks ‘Level 2’:

Display all Level 2 offences

* + - * + The user clicks ‘Level 3’:

Display all Level 3 offences

* + The user clicks ‘Level 4’:

Display all Level 4 offences

* + - The user clicks ‘Sort’
      * The user clicks ‘Ascending Order’
        + The user clicks ‘Level’:

Sort data according to level of offence in ascending order

* + - * + The user clicks ‘Offence ID’:

Sort data according to offence ID in ascending order

* + - * The user clicks ‘Descending Order’
        + The user clicks ‘Level’:

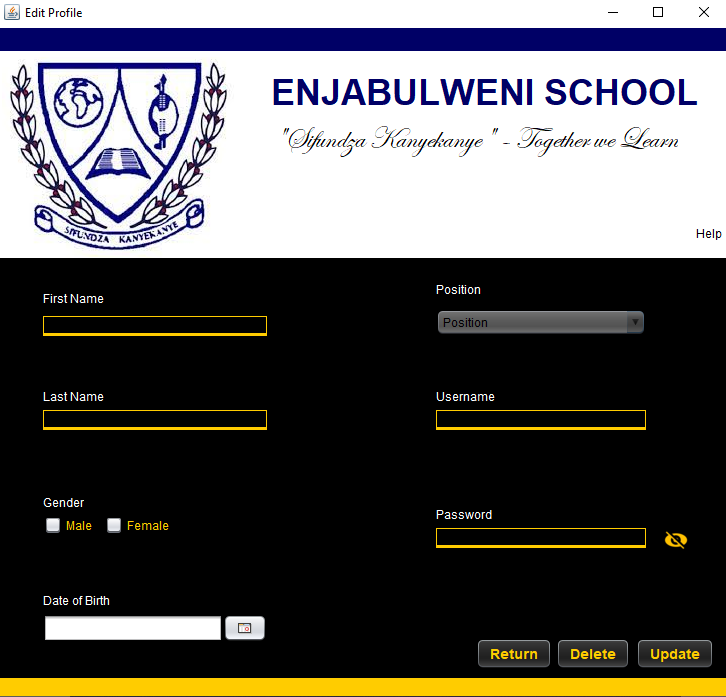
Sort data according to level of offence in descending order

* + - * + The user clicks ‘Offence ID’:

Sort data according to offence ID in descending order

* + - The user clicks ‘Restore’
      * Display all data on the table
* The user clicks ‘Clear’:
  + Set all data fields to null
* The user clicks ‘Return’:
  + Close current screen  
    Reopen Main Menu screen
* Or the user clicks the ‘Edit profile’ button on the main menu:
  + Display ‘edit profile’ screen

Close Main menu screen



* + The user changes their data and clicks ‘Change’:

Ask for confirmation to update

Begin if

If confirmation given

Validate all data

Begin if

If all data valid

Begin try-catch

Get connection to database

Update data

Display “Updated successfully”

Close current screen

Reopen Main Menu

End of try-catch

Else

Display messages informing the user of the error

End if

Else

Display “Update action aborted”

End if

* + The user clicks “Delete”:

Ask for confirmation

Begin if

If confirmation given

Begin try-catch

Get connection to database

Search table for username

Delete from table

Close current screen

Open Login screen

End of try-catch

Else

Display “Delete action aborted”

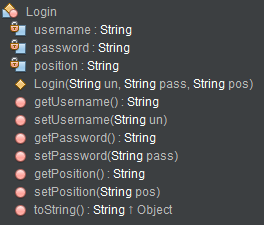
End if

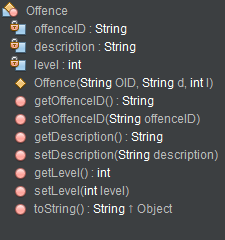
* + The user clicks ‘Return’:

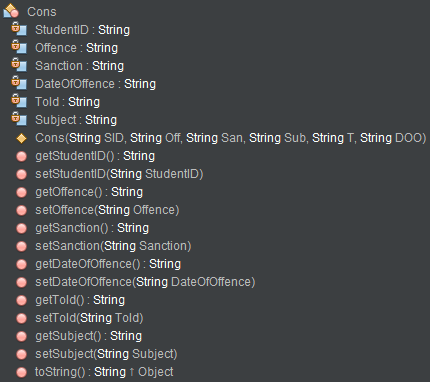
Close current screen

Reopen Main Menu screen

# **Class Diagrams Design**







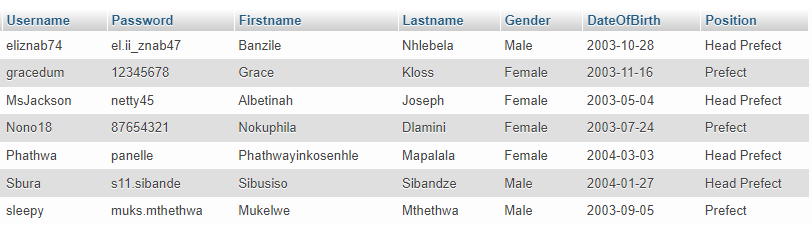


# **Secondary Storage Design**

## **User Data**

The user data will be stored in a database named “consultations” in a table named “accountdetails”. The table will store the user’s ‘password’, ’username’, ’first name’, ’last name’, ’gender’, ’date of birth’ and ’position’. The username, password, first name, last name, position and gender fields are of String type. Date of Birth is of type date.

Example:



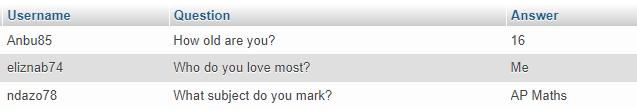
## **Help Data**

The help data will be stored on different text files. Each text file will store all information regarding a certain topic that the user can select. These files will be read from and the data will be displayed on the help page.

## **Security Data**

Security data such as the username (of type String), the security question (of type String) and the answer to said question (of type String) is stored in a table named ‘Security’. This table is used when the user wants to reset their password having forgotten it.

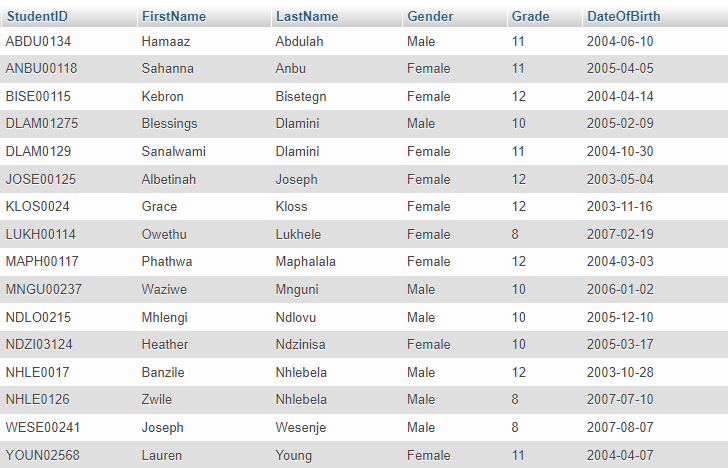
Example:



## **Student Data**

The student data will be stored in the database “consultations” in a table named “studentlog”. The table will store each student’s ‘Student ID’ (of Type String), ’first name’ (of Type String), ’last name’ (Of type String), ’gender’ (of Type String), ’grade’ (of Type int) and ’date of birth’ (of Type Date).

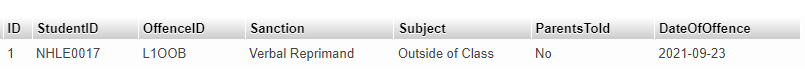
Example:



## **Student Offence Data**

This data will be stored in the database “consultations” in a table named “conlog”. The table will store the student’s ‘student ID’ (of type String), the ‘offence ID’ (of type String) representing the offence committed by the student, the ‘sanction’ (of type String) for said offence, the ‘subject’ (of type String) in which it was committed, the ‘date of offence’(of type date) and whether or not the parents have been told (of type Boolean) of this offence.

Example:



## **Parent Data**

This data will be stored in the database “consultations” in a table named “parents”. The table will store the student’s ‘student ID’ (of type String), the first and last name of the parent (of type String), the ‘gender’ (of type String) of the parent, the ‘relationship’ (of type String) that the parent has with the child and their ‘Cell number’(of type String)

Example:



## **Offence Data**

This data will be stored in the database “consultations” in a table named “offences”. The table will store the offence ID (of type String), the Description of the offence (of type String) and the level of the offence (of type integer)

Example:



## **Temporary User Data**

The password, username and position of the user currently logged into the program will be stored in a text file named “temp.txt”

Format: username#password#position

Example: eliznab74#el.ii\_znab47#Head Prefect

# **Explanation of Secondary Storage Design**

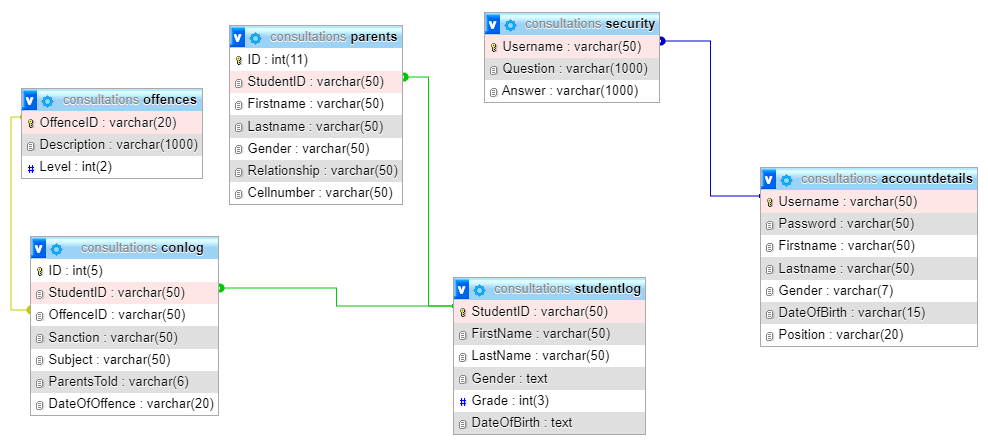
A database consisting of four different tables as well as multiple text files have been used. The database has been used to store the user data, the student log data, the security data as well as the consultations data. The text files have been used to store the username, password and position of any user currently using the program and every help topic that can be accessed using the help screen.

A database was more suitable than the text files because databases are faster when retrieving information as they extract the value the user is looking for whereas text files have to read through the whole file in an attempt to find the desired value. Databases also take up less space when normalized to 3NF as compared to text files.

Text files were used in areas that did not need a search algorithm. The text file used for storing temporary user data only stores one line that contains two fields. This will make displaying user details in the ‘edit profile’ screen much easier. Accessing the data in the text file is also faster as there is no searching needed. They were also used to store each help topic. They are only to be displayed in the help screen thus no value is needed and this ensures speed and efficiency.

Databases were used where values needed to be extracted and where speed was required thus their selection over text files and text files were used where no search algorithm was needed thus also ensuring speed and efficiency in the program.

# **Table Relationships**



# **Explanation/Description of Primary Data Structure**

## **Student Class**

The user class is used to access and change all the student details and data for the purposes of display. When the user inputs student details on the student log screen, the student class is then used to display those details using the getter/accessor methods on the table beside the input fields. Each accessor method is allocated its own field, for example, the getFirstName method is used to retrieve the first name of the student and display it on the form. When the user updates any field of data in any row of the table, the class is used to display those updates. The updates are reflected using the setter/mutator methods in the class. Each mutator method has its own field that it is assigned to, for example, the setStudentID method will be used to change the student ID of a student. The class contains a constructor method that initializes the variables of the class as well as a toString method that returns a string value of all the student details with every field being of String type.

## **Cons Class**

The cons class is used to access and change all the student offence details for the purposes of display and tracking. When the user inputs student details on the Student Offence Register screen, the cons class is then used to display those details using the getter/accessor methods on the table beside the input fields. Each accessor method is allocated its own field, for example, the getOffence method is used to retrieve the offence ID relating to the offence committed by the student and display it on the form. When the user updates any field of data in any row of the table, the class is used to display those updates. The updates are reflected using the setter/mutator methods in the class. Each mutator method has its own field that it is assigned to, for example, the setSanction method will be used to change the sanction to be enacted on the student. The class contains a constructor method that initializes the variables of the class as well as a toString method that returns a string value of all the student offence details with every field being of String type.

## **Login Class**

The login class is used to store the username, password and position of any user currently using the program in a text file. This serves the purpose of allowing all the user details to be displayed in the Edit profile screen and to control the functions that each user can access based on their position. The accessor methods are used to obtain the username of the user and use it to search for the user details in the database and this display them on the screen mentioned above. The mutator methods are used for changing any details that the user changes on the screen also changing them in the database. The class contains a constructor method that initializes all the variables of the class as well as a toString method that returns a string value of all the variables in the class. All the fields present in the class are of type string.

## **Parents Class**

The ‘parents’ class is used to access and change all the student parent details and data for the purposes of display. When the user inputs student parent details on the parent log screen, the ‘parents’ class is then used to display those details using the getter/accessor methods on the table beside the input fields. Each accessor method is allocated its own field, for example, the getFirstname method is used to retrieve the first name of the parent and display it on the form. When the user updates any field of data in any row of the table, the class is used to display those updates. The updates are reflected using the setter/mutator methods in the class. Each mutator method has its own field that it is assigned to, for example, the setStudentID method will be used to change the student ID of a student who is related to the parent. The class contains a constructor method that initializes the variables of the class as well as a toString method that returns a string value of all the parent details with every field being of String type.

## **Offence Class**

The offence class is used to access and change all the offence details and data for the purposes of display. When the user inputs offence details on the student log screen, the offence class is then used to display those details using the getter/accessor methods on the table beside the input fields. Each accessor method is allocated its own field, for example, the getOffence method is used to retrieve the offence ID of the offence and display it on the form. When the user updates any field of data in any row of the table, the class is used to display those updates. The updates are reflected using the setter/mutator methods in the class. Each mutator method has its own field that it is assigned to, for example, the setDescription method will be used to change the description of an offence. The class contains a constructor method that initializes the variables of the class as well as a toString method that returns a string value of all the offence details with every field being of String type.