

Online education

Hillcrest high school

PAT 2022 phase 1

Job ko 12ld 13741

2022

Table of Contents

[Scope/Scenario 2](#_Toc110026113)

[USER REQUIREMENTS 3](#_Toc110026114)

[Navagation/Flow diagram 6](#_Toc110026115)

[Students 6](#_Toc110026116)

[Teachers 7](#_Toc110026117)

[Administrator 8](#_Toc110026118)

[Data structures 9](#_Toc110026119)

[Database Table relationship 10](#_Toc110026120)

[Class Description and class diagram 11](#_Toc110026121)

[Class description 12](#_Toc110026122)

[DATA DICTIONARY 12](#_Toc110026123)

[DATABASE 12](#_Toc110026124)

[Text files 12](#_Toc110026125)

[Arrays 13](#_Toc110026126)

[CLASSES AND OBJECTS 13](#_Toc110026127)

[ADVANCED PROGRAMMING 13](#_Toc110026128)

[GUI Design 14](#_Toc110026129)

[DATA INPUT 21](#_Toc110026130)

[Sign up/adding validation 21](#_Toc110026131)

[How will processing be done 23](#_Toc110026132)

[What processing will need to be done? 23](#_Toc110026133)

[Checking if password is valid 23](#_Toc110026134)

[Calculate average marks 24](#_Toc110026135)

[Instantiating an object using an admin number and database 26](#_Toc110026136)

[BIBLIOGRAPHY 27](#_Toc110026137)

Topic: online education

# Scope/Scenario

During the recent years there has been an increase on online education. Online education has grown in popularity as a convenient method to study without having to leave the house. Many teachers/educators and educational institutions have utilized these strategies to instruct pupils who are unable to attend school or those who are facing difficulties. Online education is a flexible instructional delivery method that covers any learning that occurs through the internet. Online learning allows instructors to communicate with students who may not be able to enroll in a typical classroom course, as well as students who need to work on their own time and at their own pace. Sources are shared through these communications, and an open-ended synergy emerges through a learning process. When each person shares their point of view or opinion through debates and comments on others' work, it helps the student learn more effectively. This distinct advantage manifests itself in a student-centered virtual learning environment, which only online learning can provide. Software tools are required for ease of use and reliability in order to make online education a viable option. It is necessary to ensure that processes such as student registration, educational resource administration, and financial management are simple, efficient, accurate, and relevant.

The program will have three users: Student, teacher and Admin. So I will make an application using Delphi that allows students to register for online courses at a school and keep track of courses, teacher/lecturer and keeping track of student progress in a registered course with the aid of a database to help keep the school organized and students aware.

TASK:

Design and build a software system that is able to assist with students registering for an online course.

PURPOSE:

The purpose of the software is to assist students who are struggling and need extra courses. The software’s purpose is to make it easier for students ,teachers and administrators work easier by integrating a database to make recording much easier through the use of code and GUI design.

SOLUTION:

The proposed solution to assist with online courses is to design a software that will allow:

* Allow students to create or login to an account.
* Allow students to see their course and their marks.
* Allow logged in users to see the available courses.
* Allow logged in users to register or quit a course.
* Allow logged in teachers to mark/edit a students mark.
* Allow teachers to keep track of their students mark.
* Allow administrator to edit, delete and view large amounts of data where appropriate.

# USER REQUIREMENTS

Students cannot see other students student number and ID number and passwords. Students also cannot see who is in the classes.

Teacher/lecturer must be able to see all students registered for their course, as well as their email addresses and phone numbers. Also able to see student grades/marks and if the student has passed or not indicated by colours. Teacher/lecturer cannot see students passwords and other courses that they have registered to.

Admin must be able to see all students registered to each class and how many courses each student has. Admin must be able to see how much a student has paid on what date and time.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Students | Teacher | Administrator |
| Role: | Students can register for a course  and see what courses they are enrolled in. Students can see the amount needed to be paid, how much they have paid. Able to see who teaches the course. | -log in  - see how many students are enrolled in their course  -what the students marks/scores currently are.  -see their email and ‎ ‎ ‎ number details  - able to change their marks  - able to add new ‎ ‎ ‎ ‎ ‎ ‎ ‎ ‎ marks | -log in  -see which students are enrolled in which course  -see all the teachers classes  -see how much a student has paid  -able to edit students details  -can search for students in a database |
| Activity | - log on  - register for a ‎ ‎ ‎ ‎ ‎ ‎ course  - able to edit their ‎ ‎ ‎ ‎ details  - able to leave/quit a ‎ course  -able to change passwords  -see their marks | -log on  -see who is in their ‎ ‎ ‎ ‎ ‎ course  - edit/input the ‎ ‎ ‎ ‎ ‎ ‎ ‎ ‎ ‎ ‎students marks  - see a chart of the  ‎‎‎‎‎‎‎‎‎‎ whole classes progress | - log on  - see all the students  - see all the teachers  - able to see all the ‎ ‎ ‎tables and can edit ‎ ‎ ‎the tables  - see the students ‎ ‎ ‎ ‎ ‎ ‎paid amount  - do SQL queries and search for students  -can change who is enrolled in each course  -add students  -remove students  -add teachers |
| Limitations | -Cannot change their own mark  -cannot see other students details  -cannot see which students are enrolled in which course.  -cannot access the database | -cannot change students details  -cannot remove students from their course  -cannot access the database  -cannot see students passwords |  |

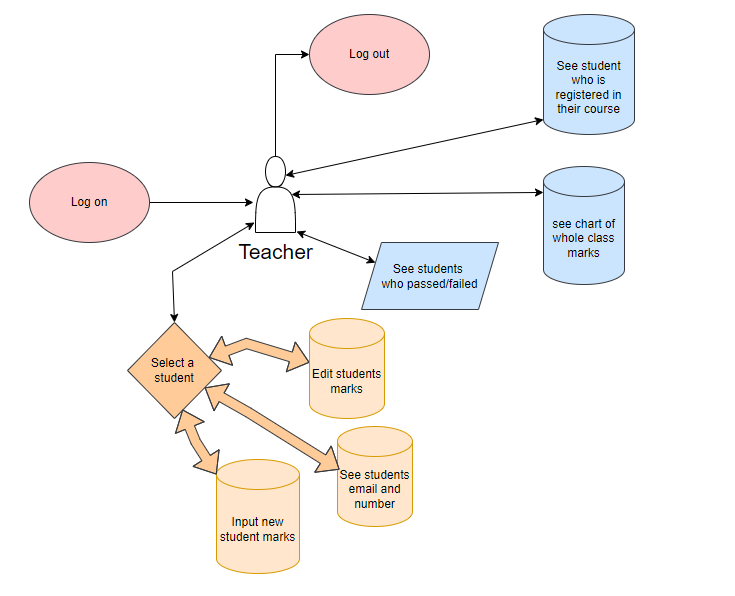
# Navagation/Flow diagram

## Students

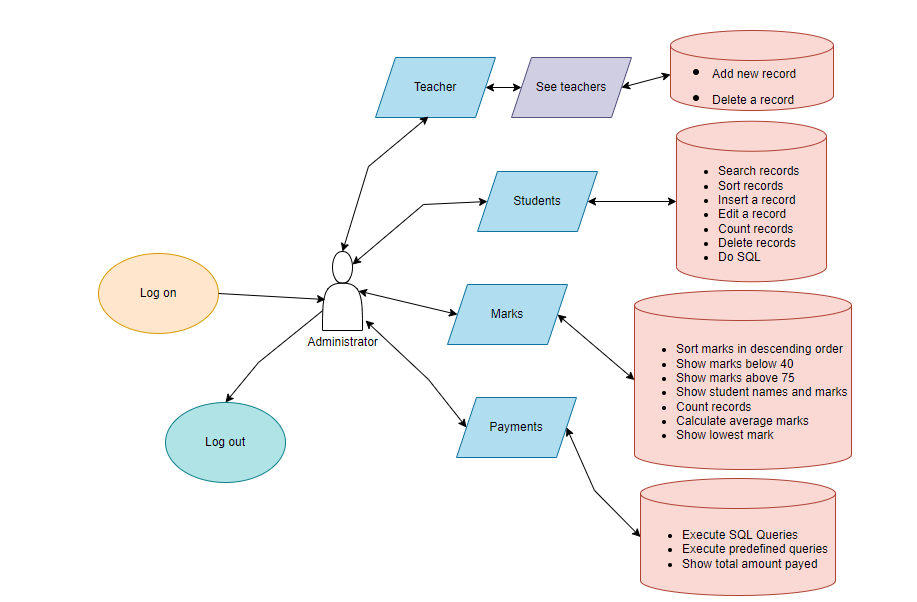
Diagram

Description automatically generated

## Teachers



## Administrator



# Data structures

tblStudents

|  |  |  |
| --- | --- | --- |
| **Field** | **Data type** | **Field size** |
| ID | AutoNumber | Long integer |
| Admin | Number | Double |
| Firstname | Short Text | 50 |
| Lastname | Short Text | 50 |
| Courses | Short Text | 100 |
| Email | Short Text | 50 |
| Gender | Short Text | 8 |
| DateOfBirth | Date/Time | Short Date (YYYY/MM/DD) |
| Number | Short Text | 10 |
| Password | Short Text | 50 |

tblTeachers

|  |  |  |
| --- | --- | --- |
| **Field** | **Data type** | **Field size** |
| ID | AutoNumber | Long integer |
| Teacher | Short Text | 50 |
| Course | Short Text | 50 |
| Email | Short Text | 50 |
| DateOfWorked | Date/Time | Short Date (YYYY/MM/DD) |
| password | Short Text | 50 |

tblMarks

|  |  |  |
| --- | --- | --- |
| **Field** | **Data type** | **Field size** |
| **ID** | **AutoNumber** | **Long integer** |
| Student | Number | Integer |
| Course | Short Text | 5 |
| Marks | Number | Double |
| TestNo | Number | Double |

tblPayments

|  |  |  |
| --- | --- | --- |
| **Field** | **Data type** | **Field size** |
| **ID** | **AutoNumber** | **Long integer** |
| studentID | Number | Double |
| Processed | Yes/no | 3 |
| payamount | Number | Double |
| paydate | Date/Time | Short Date (YYYY/MM/DD) |

# Database Table relationship

Graphical user interface

Description automatically generated

# Class Description and class diagram

|  |
| --- |
| **Student details** |
| -admin: String;  - Lastname: String;  - Firstname: String;  - gender: String;  - email: String;  - Number: String;  - Courses: String;  - arrcourses: array[1..11] of string;  - errors: String;  - numberofcourses:integer; |
| + constructor Create(fadmin: integer);  + getAdminNO: String;  + getLastname: String;  + getFirstname: String;  + getGender: String;  + getEmail: String;  + getdateofbirth: string;  + getNumber: String;  + getcourses: string;  + getcoursestostring:string;  + getallcourses:string;  + getpayment:integer;  + getmarks:string;  + getmarksasint:integer;  + geterrors:string;  + determinecolor(fmarks:integer):Tcolor;  + checkFieldSize(data: String; min: Byte; max: Byte): Boolean;  + getnumofcourses:integer;  + getcoursemark(course:string):integer;  + iscourseempty:boolean;  + setLastname(pLastname: String);  + setFirstName(pFirstname: String);  + setGender(pGender: String);  + setEmail(pEmail: String);  + setNumber(pnumber: String);  + setAdminNO(pAdminNO: String);  + Addcourses(course:string);  + updatecourses(scoursei:string);  + insertnewmarks(acourse:string; marks,testno:integer);  + updatemarks(acourse:string; marks,testno:integer); |

# Class description

The class is used to create a student in memory by using the admin number. When an admin number is found in the database the class will create all the relevant fields example it will set the student name and surname with all their courses. The class has functions to get all the values like name and surname. The class can also update values by using procedures or functions. The class can also determine the colour of a students marks. If a students marks are 40 and less the output colour will be red if the marks are 40-70 the marks will be yellow else the marks will be green. The class can also validate fields for example it validates phone numbers to check if there is 10 digits.

The above class/object will be used across multiple forms to ensure that the form data is consistent in both format and meets all presence check requirements.

# DATA DICTIONARY

## DATABASE

The database will be used to organize and store a lot of vital data required by the software. Using the database and the software, users are able to create, update, view and delete the data. Most data that is inputted will first be validated through a set of strict validation rules that ensure only valid data types are entered into the database and should limit the number of errors.

## Text files

Text files allow for information to be stored without any special formatting or requirements and is great for storing in permanent memory. When a student logs in they have an option to remember their first name and password, when a user clicks “remember me” and successfully logged in, it will store their first name and password in a text file. So when the user logs in they don’t need to put in name and password.

Use of text file to store the help messages. So when the user clicks on help it will display the contents of the text file. And an admin can change the help messages.

## Arrays

One dimensional array to read the contents of a text file and store it in each line into an array.

Array of objects is used to create each student thus it will be easier to get the values from every student easily and allows for more efficient code.

## CLASSES AND OBJECTS

Classes and Objects are part of object-orientated programming (OOP) and will be used in the program in a variety of suitable ways. One of the planned ways is to create an object that is stored in memory so when a user requests for information it is easier to display the relevant information in a neat and efficient way. To do this I will be using a lot of functions and procedures.

## ADVANCED PROGRAMMING

Advanced programming is a requirement for the PAT. The program is planned to use the following advanced programming methods such as:

* The use of Array of Objects
* Inputting values into the TChart to create a graph for the marks

Array of objects:

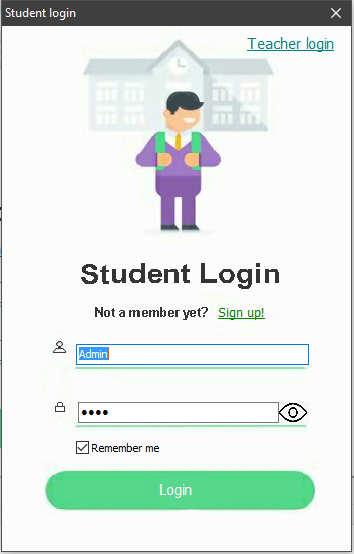


The object is instantiated when the teacher logs in and how ever many students are in that course will be created into an array of Tstudent objects.  It simplifies access to the information.

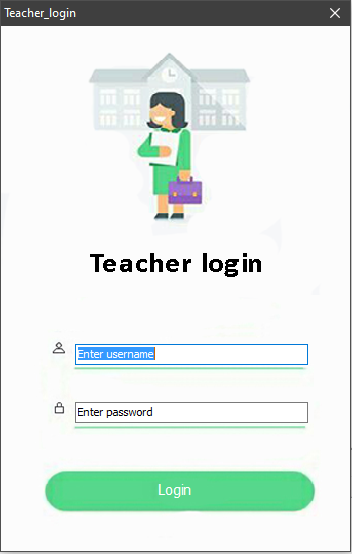
Making the coding easier to read and less redundant thus making the application more efficient.

# GUI Design

NOTE: The program will have human readable error messages which will be displayed in the event of a validation error or other issues that may occur during the GUI use.



|  |  |
| --- | --- |
| **Component** | **Use** |
| TImage | Used to display the login screens defined by each user. |
| TLabel | Used to Login a student when the firstname and password is found in the database and show the main form. |
| TLabel | Used to show the regisration form to allow users to input their information. |
| TImage | Allow users to see or unsee their passwords |
| TCheckbox | Allows users Login information to be saved to a text file |

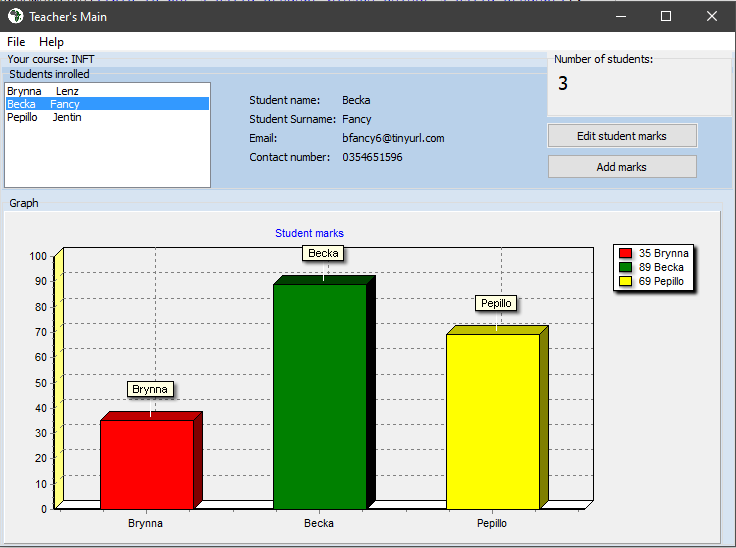


|  |  |
| --- | --- |
| **Component** | **Use** |
| TImage | Used to display the login screens defined by each user. |
| TLabel | Used to Login a teacher when the Username and password is found in the database and show the teachers main form. |

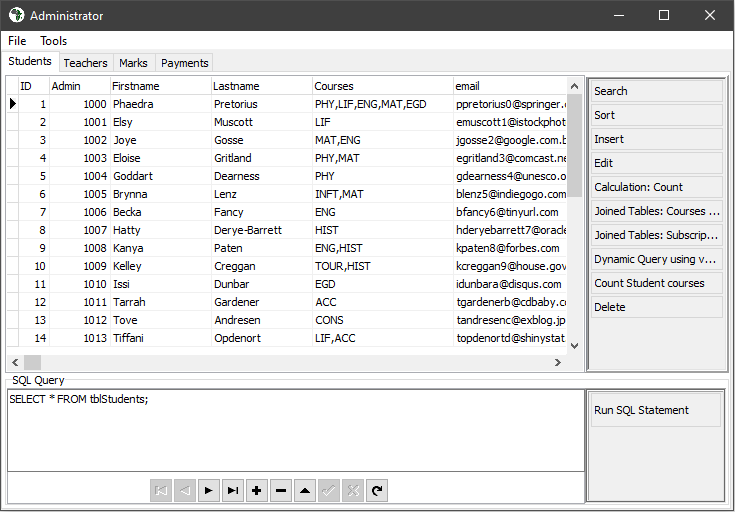
Graphical user interface, application

Description automatically generated

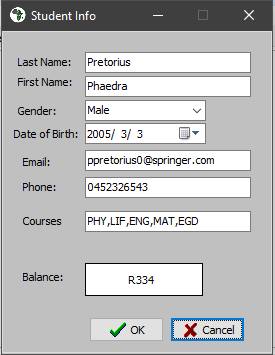
|  |  |
| --- | --- |
| **Component** | **Use** |
| TListbox | Used to display the available courses so that users can click on it. |
| TButton | Allows users to select a course and register for the course. |
| TButton | Allows users to select a course and quit the course. |
| TRichedit | Allow users to see their course marks |
| TMainMenu | Allow users to change their emails, number and password. See help messages. Log out. Exit. |
| TLabel | Allow users to see their registered courses and the amount they have paid. |



|  |  |
| --- | --- |
| **Component** | **Use** |
| TChart | Used to display the students marks and determines the colour. If a students mark is less than 40 then the colour will be red. If the students mark is more than 40 but less than 80 then the colour will be yellow else the colour will be green. |
| TLabel | Allows the teacher to see each students details like email, name, surname and number. Determines how many students are in the class |
| TButton | Allows the teacher to add/edit students marks by entering their marks and mark number. |
| TMainMenu | See help messages. Log out. Exit. |
| TListbox | Allow the teacher to select each student and display their releveant details. Allows to edit student marks/add marks |

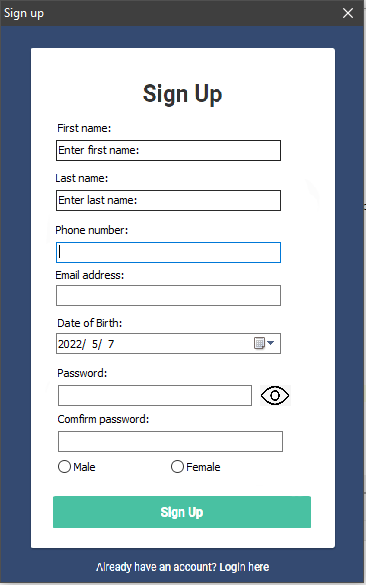


|  |  |
| --- | --- |
| **Component** | **Use** |
| TPageControl | Able to switch between pages quickly and easily. Each page has a dbgrid that displays different tables |
| TDBGrid | Allows administrator to see the contents of a table. |
| TEdit | Allows administrator to write SQL and execute specific queries |
| TButton | After the administrator has written the SQL the administrator can execute those queries. Can add a teacher. Can Sort the marks in descending order. Can display the amount of marks below 40.  Can display the amount of marks above 75. Can join tables to show student names and marks. Can calculate the average marks. |
| TButtonGroup | Allows administrator to search for specific requirements on the table. Allows the admin to sort the table in alphabetical order. Allows administrator to add a new student. Allows administrator to edit a students details using their admin number.  Can count how many records there are in the table. Can delete a record |
| TDBNavigator | Allow administrator to navigave between the records easily and able to edit as well. |



|  |  |
| --- | --- |
| **Component** | **Use** |
| TEdit | Allows populates all relevant values and allows the user to edit the values. |
| TButton | Allows the user to comfirm all the changes made to the data. |
| TDateTimePicker | Allows users to select a date at which the student was born. |
| TComboBox | Allow user to select a gender with relative ease. |

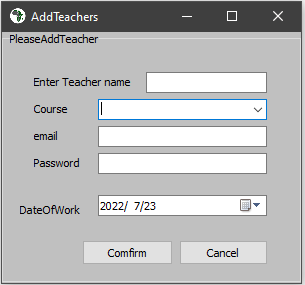
Adding a student



# DATA INPUT

### Sign up/adding validation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | **Source** | **DataType/ Format** | **GUI**  **Component** | **Validation** | **Error message** |
| Firstname | Keyboard | String | TEdit | Check if the field is not empty  And needs a minimum of 2 characters | Firstname size must be at least 2 characters and no more than 50 characters. |
| Lastname | Keyboard | String | TEdit | Check if the field is not empty  And needs a minimum of 2 characters | Lastname size must be at least 2 characters and no more than 50 characters. |
| Phone number | Keyboard | String | TEdit | Check if not empty  There must be 10 digits and the first digit must be 0 | Phone number incorrect. |
| Email address | Keyboard | String | TEdit | Checks if there is a ‘@’ symbol and that there is a ‘.’ After the @ symbol. | Invalid email address. |
| Date of Birth | Keyboard | DateTime- shortdate (YYYY/MM/DD) | TDateTimePicker | Checks if the age is greater than 14. | Need to be at least 14 years old to sign up. |
| Gender | Keyboard | String | TRadioGroup | Check If one was selected | Please select a gender. |
| Password | Keyboard | String | TEdit | Check if the password has the following criteria:  Has a symbol  Has a capital letter  Has a number  Has a lower case letter.  Length of password at least 6 characters | (Will show what ever criteria is missing.)  Example  Length of password needs to be at least 6 characters. |
| Comfirm password | Keyboard | String | TEdit | Check if the password matches confirm password. | Password does not match |
| Sign up | Mouse |  | TLabel | Will validate all the above requirements and will check if the firstname is used in the database. If the firstname is used in the database then error will occur. | User name has been taken please try another one |



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | **Source** | **DataType/ Format** | **GUI**  **Component** | **Validation** | **Error message** |
| Teacher name | Keyboard | String | TEdit | Check that the fields are not empty | Error please fill in fields correctly |
| Course | Keyboard | String | TComboBox | Check that a field is selected | Please select a course. |
| Password | Keyboard | String | TEdit | Check if the password has the following criteria:  Has a symbol  Has a capital letter  Has a number  Has a lower case letter.  Length of password at least 6 characters | (Will show what ever criteria is missing.)  Example  Length of password needs to be at least 6 characters. |
| Email address | Keyboard | String | TEdit | Checks if there is a ‘@’ symbol and that there is a ‘.’ After the @ symbol. If the ‘.’ Is before the @ then email is not valid | Invalid email address. |
| Date of work | Keyboard | DateTime- shortdate (YYYY/MM/DD) | TDateTimePicker | Cannot be validated because the start of work can be in the past, present or future. | No error message |

# How will processing be done

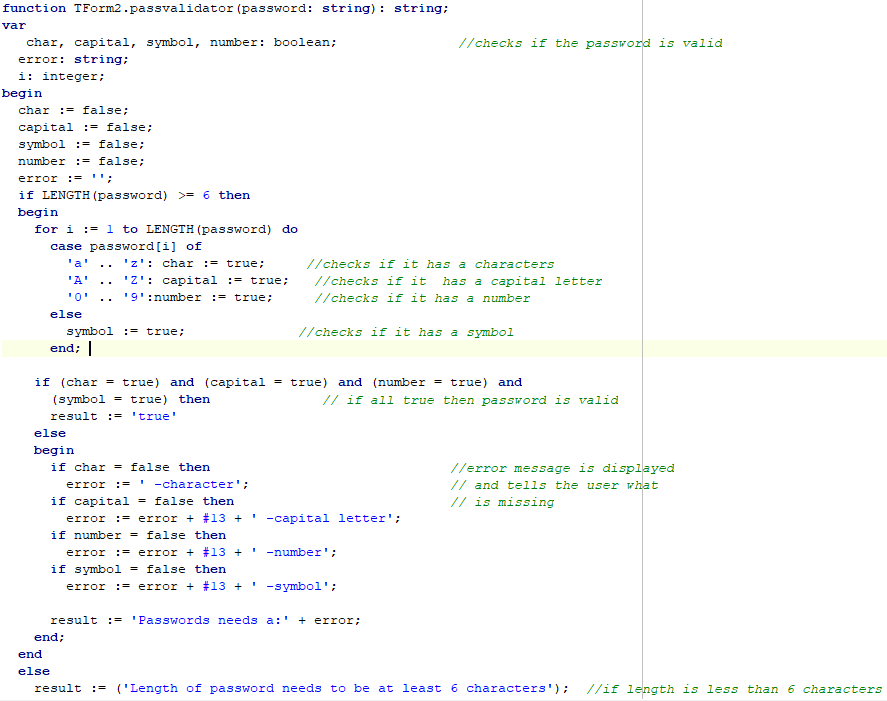
The software will utilize data validation to maintain data integrity and prevent any unforeseen failures that might have an impact on the database or other crucial parts of the program. To ensure the program's continued integrity, all inputs, including those made through components that ordinarily wouldn't require it, will be checked whenever possible. Additionally, all inputs will be checked for NULL or empty values. An appropriate error message will appear when necessary.

## What processing will need to be done?

* Checking if a password is strong enough.
  + Needs to have a symbol
  + Needs to have a capital letter
  + Needs to have a lower case letter
  + Needs to have a number
* Need to be able to calculate the sum of a table.
* Able to read from a database and instantiating a object using the database.
* All input fields that are required will be validated to ensure that there is no NULL values and all other validation rules are met.
* Check if someone is above 14 years old with the Tdatetimepicker component.
* The inputted email address will be validated by checking if email address contains a ‘@’ sign followed by a ‘.’ After the ‘@’.
* The inputted mobile number will be validated to ensure it consists of 10 digits and begins with a 0.
* Capable of editing a specific database record when the administrator requires itAble to append a database.
* Able to search for the correct record and delete the record.
* The use of arrays to turn the course code into the full name of the course.
* The use of text files to read and write data in permanent memory. This will read the stored username and password, populated when appropriate.
* When the student pays, the amount must be at least R10 and less than R1100.
* When the teacher selects a student the names will be populated. This will be done by checking the Itemindex value and using array of objects.

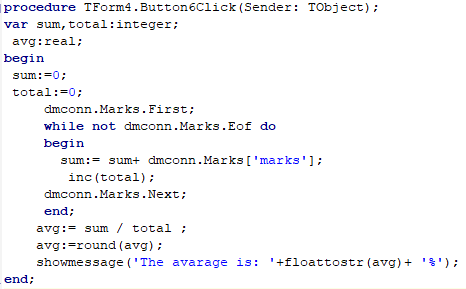
## Checking if password is valid

|  |
| --- |
| char 🡨 false  capital 🡨false  symbol 🡨false  number 🡨false  if length of password > 6 then  begin  for I = 1 🡪 length of password  begin  case password of  a to z then char 🡨true  A to Z then capital 🡨true  0 to 0 then number 🡨true  Else symbol 🡨true  End  End  If char and capital and number and symbol = true then password is valid else the password is not valid. |



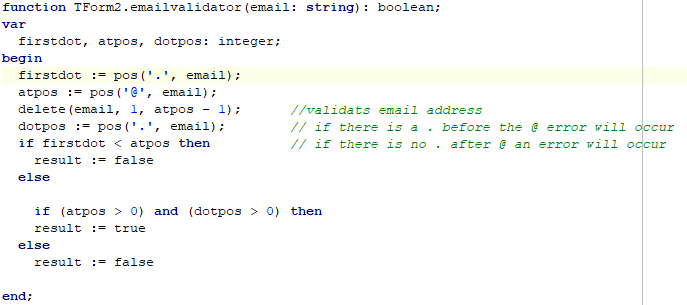
## Calculate average marks

|  |
| --- |
| sum 🡨0  total 🡨0  open tblMarks  go to first record  while not end of file tblMarks  begin  total 🡨total+ tblMarks[‘marks’]  inc(sum)  go to next record  end  average 🡨total/ sum  output average |



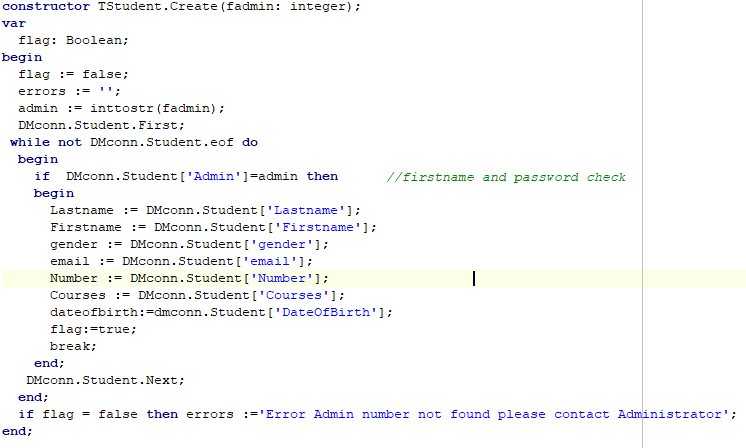
Validate an email address

|  |
| --- |
| firstdot🡨Position of ‘.’  atpos🡨position of ‘@’  delete(email from position 1 to atpos-1)  dotpos🡨position of dot in email  if firstdot < atpos then result will be false else  if atpos greater than 0 and dotpos greater than 0 then result = true else result=false |



## Instantiating an object using an admin number and database

|  |
| --- |
| flag🡨false  errors🡨’’  go to first record in student table  while not end of file begin  if student[admin] = admin then  begin  lastname🡨Student['Lastname']  Firstname 🡨 Student['Firstname']  gender 🡨Student['gender']  email 🡨 Student['email']  Number 🡨 Student['Number']  Courses 🡨 Student['Courses']  dateofbirth🡨 Student['DateOfBirth']  flag🡨true  break  end  go to next record  if flag = false then error error message= ‘Error Admin number not found please contact Admnistrator’; |



# BIBLIOGRAPHY

Bowman, W. (2022) *Designing database applications - Delphi Guide* *Delphi Power*.3 June 2022 [online]. Available from: https://www.delphipower.xyz/guide\_6/designing\_database\_applications\_1.html [Accessed 20 July 2022].

Olivier, JAK. “TeachITza.com - Teaching and Learning Resource for IT Educators and Learners in South Africa Using Delphi.” *Teachitza.com*, 2008, teachitza.com/delphi/index.htm. Accessed 24 Mar. 2022.

StackOverFlow. “Stack Overflow - Where Developers Learn, Share, & Build Careers.” *Stack Overflow*, 2019, stackoverflow.com. Accessed 21 July 2022.

usability.gov. “User Interface Design Basics | Usability.gov.” *Usability.gov*, 2019, www.usability.gov/what-and-why/user-interface-design.html. Accessed 27 Apr. 2022.

Mr long. “Mr Long Education - IT & CAT - YouTube.” *Www.youtube.com*, www.youtube.com/channel/UCPQG0dcSGvOP20NLcjnYTPQ/playlists. Accessed 10 July 2022.