Design

Contents

[Output 2](#_Toc348706875)

[Input 4](#_Toc348706876)

[Form design 5](#_Toc348706877)

[frmStart 5](#_Toc348706878)

[frmMain 5](#_Toc348706879)

[frmSettings 6](#_Toc348706880)

[frmStudent 7](#_Toc348706881)

[frmStaff 7](#_Toc348706882)

[frmResults 8](#_Toc348706883)

[frmSeeResults 8](#_Toc348706884)

[Files /Data structures and methods of access 9](#_Toc348706885)

[Data Structures 9](#_Toc348706886)

[Student table 9](#_Toc348706887)

[Teacher table 9](#_Toc348706888)

[Stud av table 10](#_Toc348706889)

[Staff av table 11](#_Toc348706890)

[Day table 11](#_Toc348706891)

[Appointments table 12](#_Toc348706892)

[Lesson table 13](#_Toc348706893)

[Methods of access 13](#_Toc348706894)

[Validation 14](#_Toc348706895)

[Presence of day settings validation 14](#_Toc348706896)

[Basic presence check 14](#_Toc348706897)

[Processing stages 15](#_Toc348706898)

[Evaluation Criteria 16](#_Toc348706899)

# Output

The system will display to students blocks they have declared themselves to be available for. This will up until the sorting algorithm has been run they can change and see their choices. This will be so that they can check what they entered and change it if needs must. Once the deadline for appointment selection is over and the admin trigger the sorting algorithm emails will be sent to each students college email. The emails will contain the names of the teachers the student has an appointment with along with the time and the day, they will also contain a section of text written by an admin with further information to be added at the discretion of the admin.

The system will display blocks the same way as the students. Since like the students they will also want to check what times they have declared themselves available for and also will likely wish to change it closer to the time as circumstances change. The teachers will, the same as the students, receive an email containing the names of the students that they have appointments with along with the times and days in order so that they can print it off giving them a schedule.

Admin will also be able to see the appointments that users have received though a results form which contains links to another form containing the selected user’s appointments. This will be so that they can check the appointments of individuals primarily teachers to see whether they are booked at certain times as they may need to manualy fiddle around with appointments to fit others in after the sorting algorithm has been run since it will not be perfect and students may find that they can find more time in their schedual for an important appointment.

The system will also output the data of appointments to the user who it involves via the send results button on the admin form which will send out a mass mail merge involving the students.

|  |  |
| --- | --- |
| Example of an email to a student containing appointment infromation, this will not differ from the email to teachers. |  |
|  | Subject will be editable in email settings  Appointments list generated from assigned appointments  Conclusion editable in email settings. This may contain anything though primarily designed to contain a message from the event organiser reminding participants about a extraneous information ie parking. |
| Example of the further restults form |  |
|  | See results forms with a student or teacher selected and the populated list box containing the list of appointments the individual has been asigned, shown as and admin would see it. Only the informaiton about timings and te name is being inclueded buecause it is all that is necicary for the processes stated above.  Further description of the form given in the form design section. |

# Input

Data will be collected from the college MIS database as to the lessons and the student and staff data. It will be read by the program and will be line by line be written into the desired format in a set of new DAT files created for and by the program. It is done this way because it is fast and because so long as the code is correct the process will be completed with out error.

Upon loading the program the user will automatically input their area logon which is checked against the system to log the user in if they are a current user. This is done because the user has already logged on proving who they are, it is easier to program and takes up less time.

The users will input data as to their availability on the availability form. This is done by checking checkboxes which refer to blocks of time. Upon checking an option the system data will be altered to reflect this change. Ie the availability file of the student or staff member will be changed for the time in question. It is done this way because it is much more simple to handle than time in large blocks but also because it makes sure that people sign up to the minimum time required for them to have any reasonable chance of getting the appointments they want.

# Form design

## frmStart

|  |  |
| --- | --- |
|  | btnEnter  btnAdmin |
| First form opened  when opened – checks the network profile and from that derives weather they are a teacher or a student and if they are a member of staff weather they are an admin the form then acts acordingly  btnEnter – sends the user to the student form if the user is a student and the staff av form if they are a member of staff.  btnAdmin – only visible if the memer staff is also admin, when clicked opens the Admin form. | |

## frmMain

|  |  |
| --- | --- |
|  | btnImport  btnSettings  btnAlgorithm  btnSeeResults  btnSendResults  btnBack |
| btnImport – a button that when clicked will import data from the colleges csv files that contain student data and teacher data into the csv’s that will be used for the consultation evening.  btnsettings – a button that when clicked will open up the settings form  btnAlgorithm – a button that will execute the algorithm that will organise the appointments  btnSeeResults – a button that will open up the results form  btnSendResults – a button that will seen the lists of appointmetns to each student and teacher via college e-mail  btnBack – a button that will open the Start form | |

## frmSettings

|  |  |
| --- | --- |
|  | cmbNOofDays  cmbDayNO  chk5min  chk10min  cmbStart  cmbEnd  btnBack |
| cmbNOofDays - drop down menu from which the user will select the number of days they wish to have the consultaition evening across  cmbDayNO – drop down menu from which the user will select the day by number they wish to edit the settings for and will only go up to the number of days selected  chk5min – when checked changes the information for that day changing it so that it is recorded as being 5 mins. When checked chk10min is un checked and when chk10min is checked chk5min is unchecked. Will be the default checked box  chk10min – when checked changes the information for that day changing it so that it is recorded as being 10 mins. When checked chk10min is un checked and when chk10min is checked chk5min is unchecked.  cmbStart – drop down menu from which the user will select the start time for the consultation evening. The options will be staggered by 30 mins and when changed the change will be set that time as the start time on the day record for that day. Will only display times before cmbEnd.  cmbEnd – drop down menu from which the user will select the end time for the consultation evening. The options will be staggered by 30 mins and when changed the change will be set that time as the end time on the day record for that day. Will only display times after cmbStart  btnBack – button that will be used to exit back to the admin form. Upon clicking the button the day structures will be validated, they will be checked so that the form cant be closed unless all the days have a start and end time. | |

## frmStudent

|  |  |
| --- | --- |
|  | lblName  chklstAvailable |
| lblName – a lable that will cotain the name of the user  chklstAvailable – a checked list box in which the student will check the boxes of the bocks of time he will be available. | |

## frmStaff

|  |  |
| --- | --- |
|  | lblName  chklstAvailabilty |
| lblName – a lable that will cotain the name of the user  chklstAvailable – a checked list box in which the member of staff will check the boxes of the bocks of time he will be available. | |

## frmResults

|  |  |
| --- | --- |
|  | lstStudents  lstStaff  btnBack |
| lstStudents – a list box that contains the names of all the students, when a students name is clicked on it will open up the see results form with the data on the students appointments on it.  lstStaff – a list box that contains the names of all the staff, when a staff member is clicked on it will open up the see results form with the data on the member of staff’s appointments on it  btnBack – closes the form and opens up the main form again. | |

## frmSeeResults

|  |  |
| --- | --- |
|  | lblTitle  lstAppointments  btnBack |
| lblTitle – a lable that contains the name of the student or member of staf that is having their appointments looked at  lstAppointments – a list box that contains the appointments that the student or the member of staff has including the teacher , the student and the time. Also when they are clicked the student or teacher that the appointment is will is selected on the see results form. This is incase after seeing the first users information the admin has a follow question he wishes to resolve.  btnBack – a button that when clicked closes the see results form and opens the results form. | |

# Files /Data structures and methods of access

## Data Structures

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Student table (StudNo, StudID, Forename, Surname, Year, Form, Email, PhoneNO) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | | coments | | Example data |
| Stud NO. | Short | 2 | 0000 |  | Primary key | |  | | 0011 |
| Stud ID. | Integer | 4 | 000000 |  | To store the id of the student for outputs | |  | | 112544 |
| Forename | String | 60 | xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx |  | To store the name of the student for outputs | |  | | Jason |
| Surname | String | 60 | xxxxxxxxxx xxxxxxxxxx xxxxxxxxxx |  | To store the name of the student for outputs | |  | | Scott |
| Year | byte | 1 | 00 |  | Purpose to help divide between the years which will have separate consultation evenings | |  | | 13 |
|  | | | | | | | | | |
| Teacher table (StaffNO, Forename, Surname, Email, Admin) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | coments | | Example data | |
| Staff NO | short | 2 | 000 |  | Primary key |  | | 032 | |
| Forename | string | 60 | xxxxxxxxxx  xxxxxxxxxx  xxxxxxxxxx |  | To store the teachers name for outputs |  | | Jane | |
| Surname | string | 60 | xxxxxxxxxx  xxxxxxxxxx  xxxxxxxxxx |  | To store the teachers name for outputs |  | | Law | |
| Admin | Boolean | 2 | True/false |  | To decide weather the teacher will have access to the admin area |  | | true | |
|  | | | | | | | | | |
| Stud av table (Appointment, *StudNO*, *Day*, Available) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | coments | | Example data | |
| appointment | byte | 1 | 000 | Must be in staff av table | Primary key |  | | 12 | |
| Stud NO. | Short | 2 | 0000 |  | Primary key / Foreign key |  | | 0011 | |
| Day | Byte | 1 | 00 |  | Primary key / Foreign key |  | | 1 | |
| Block | Byte | 1 | 0 |  | To store which block the student is available for |  | | 1 | |
| Available | Boolean | 2 | True / False |  | to store whether the student is available |  | | True | |
|  | | | | | | | | | |
| Staff av table (Appointment, *StaffNO*, *Day*, available) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | coments | | Example data | |
| Appointment | Byte | 1 | 000 | Must be in stud av table | Primary key |  | | 001 | |
| Staff no. | Short | 2 | 000 |  | Primary / Foreign key |  | | 0012 | |
| day | Byte | 1 | 00 |  | Primary / foreign key |  | | 01 | |
| Block | Byte | 1 | 0 |  | To store which block the staff member is available for |  | | 1 | |
| available | Boolean | 2 | True/False |  | Store weather or not they are able to do that appointment |  | | false | |
|  | | | | | | | | | |
| Day table (Day, Start, End, length) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | coments | | Example data | |
| Day | byte | 1 | 00 |  | Primary key |  | | 03 | |
| Start | Date |  | hh/mm |  | Records the start time |  | | 0600 | |
| End | Date |  | hh/mm | Must be atleast an hour and a half after start | Records the end time |  | | 0830 | |
| Length | byte | 1 | 5|10 | It will be from a selection of 5 to 10 mins | It is used to work out how many appointments. If 5 1 hour blocks and if 10 an hour and a half | There will be a drop down | | 10 | |
|  | | | | | | | | | |
| Appointments table (*Day*, *LessonNO*, appointment) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | coments | | Example data | |
| Day | Byte | 1 | 00 |  | Primary key / Foreign key |  | | 01 | |
| Lesson NO. | Short | 2 | 0000 |  | Primary key / Foreign key |  | | 0456 | |
| appointment | byte | 2 | 0000 |  | Stores the time as the slot |  | | 0023 | |
|  | | | | | | | | | |
| Lesson table (LessonNO, *StudNO, StaffNO*) | | | | | | | | | |
| Field name | Data type | Length (bytes) | Format | Validation | Purpose | comments | | Example data | |
| Lesson NO. | Short | 2 | 0000 |  | Primary key |  | | 0124 | |
| Stud NO. | Short | 2 | 0000 |  | Foreign key |  | | 0011 | |
| Staff NO. | Short | 2 | 0000 |  | Foreign key |  | | 024 | |
|  | | | | | | | | | |

## Methods of access

# Validation

## Presence of day settings validation

When btnBack is clicked on the settings form it will check whether there has been a value for both the start and end days have been entered e.g.

For counter = 1 to NDays

If day(counter).start = -1 then

set = false

Else if day(counter).end = -1 then

set = false

Else if Day(counter).Date = -1 OR 0 then

Set = false

end if

if set = false then

msgbox(“you have to enter start and end time for all the days”)

exit sub

end if

next

## Basic presence check

This will be used for checking if there has been a conclusion set for the Results emails set in the Email Settings. It will be triggered when btnEmail is clicked

If email.Conclusion = “” then

Msgbox(“You must add a email conclusion before you are able to send out the emails”)

exit sub

End if

## Error recognition in importing

This will be used while importing data from csv to dat, example used is for students

Try

Student.forename = currentrow(0)

Student.surname = currentrow(1)

Ect

Catch

Message (“Error with the student csv file”)

Exit sub

End try

# Processing stages

## Code to find out who the user is logged on as

Dim parts() As String = Split(My.User.Name, "\")

ID = parts(1)

The code finds what the network logon is and then chops of the username which will just be their ID for both teachers and studnets

## Import from csv to dat

Fileopen(selectedcsv)

While not document over

Current row = filereader.currentrow

Variable 1 = currentrow(0)

Variable 2 = currentrow(1)

Variable 3 = currentrow(2)

Variable 4 = currentrow(3)

putSelected(selected, selecte.selectedNO)

End while

The code opens the csv and line by line reads it in where it is split up by the deliminating character and the fields are put into the correct variabeles of the corresponding structure where they are then put into their corresponding dat file.

## Get function

getRecord (record number)

file open (record.dat , record length)

file get ( getrecord, record number)

file close

the code opens the desired dat file, finds the desired record using the parameter and then returnes it to be stored in a record structure

## Put function

Putrecord(editedrecord , record number)

Fileopen( record.dat , record length)

Fileput( edited record, record number)

Fileclose

The code opens the desired dat file finds the desired record via the recordnumber parameter and then overwrites the record there with the editedrecord perameter and then closes the file.

## Military time

Militarytime( timeNO)

Hours as string

Minuets as string

Hours = (timeNO \ 12) as string

If length of hours = 1 then

Hours = “0” + hours

Else if length of hours = 0 then

Hours = “00”

Minuets = timeNO – ((timeNO \ 12 ) \* 12

Minuets = minuetes \* 5 as string

If length of minuets = 1 then

minuets = “0” + hours

Else if length of minuets = 0 then

minuets = “00”

militarytime = hours + minuets

the code takes a time in the form of a number from 0 to 287 and turnins it into a 24 hour clock time . with hours being the truncated number divied by 12. And minuets being the remainder times 5. It also fills up the empty characters with zeros

## the sorting algorithm

For counter1 = 0 To Number of students

student = GetStudent(counter1)

For counter2 = 0 To Number of lesson

Lesson = Getlesson(counter2)

If Lesson.StudNO = student.StudNO Then

Staff = GetStaff(counter2)

For counter3 = 0 To Numbeer Studav records

StudAv = GetStudAV(counter3)

If StudAv.StudNo = student.StudNO And StudAv.available = True Then

For counter4 = 0 To Number of StaffAv

StaffAv = GetStaffAV(counter4)

If StaffAv.StaffNO = Staff.StaffNO And StaffAv.Available = True Then

Appointment dat is put into appointment varable

Sets appoinments in different blcoks to unavailable

The code goes student by student and for each student lesson by lesson. When it finds a teacher a student has in lesson it looks for times when they are mutualy available and when it finds this it makes the appointment it then makes all times for the student that are in different boxes unavailable.

# Evaluation Criteria

The project must

* Produce an easily printable list of appointments for each student and teacher
* Must score a unanimous vote as simple to use in the beta test.
* Must never generate two consecutive appointments for a student.
* Must handle multiple lessons with multiple teachers
* Must be multi user multi access
* Must allow users to log on just by their network area
* Must keep the appointments for each student within a reasonable distance of each other
* Loads within 10 seconds
* Must be less than 10 mb
* There is a consistent house style that keeps colour to a minimum
* Text boxes and buttons are clearly labelled
* All inputs must be validated so that errors don’t often arise
* Must not cost more than £100
* Must not be in breach of the data protection act
* Must import student, staff and lesson information from csv to dat
* Students must be able to
  + Input when they are available
  + Change their availability settings
* Staff must be able to
  + Input when they are available
  + Change when they are available
* Admin must be able to
  + Decide set the number of days
  + Set the start and finish times for the consultation evenings
  + Be able to view the appointments for each student
  + Must be able to email students and teachers their appointments
  + Must be able to reset the dat files
* must satisfy the customer