{TITLE}

Name: James Cole

Candidate Number: 3021

Centre Name: The King’s School Grantham

Centre Number: 26220

Contents

[1. Analysis 3](#_Toc473791896)

[1.1 Background 3](#_Toc473791897)

[1.2 Investigation 4](#_Toc473791898)

[1.3 Program Requirements 7](#_Toc473791899)

[1.4 Data Volumes 7](#_Toc473791900)

[1.5 Data Dictionary 8](#_Toc473791901)

[1.6 Entity Relationship diagram 9](#_Toc473791902)

[1.7 Data Flow Diagram 9](#_Toc473791903)

[1.8 Prototype Designs 11](#_Toc473791904)

[1.9 Use of Language 11](#_Toc473791905)

[2Design 11](#_Toc473791906)

[2.1 Design Overview 11](#_Toc473791907)

[2.2 Key Algorithms 11](#_Toc473791908)

[2.3 Forms/Modules 11](#_Toc473791909)

[2.4 Data Structures 11](#_Toc473791910)

[2.5 Data requirements 11](#_Toc473791911)

[2.6 File organisation and processing 11](#_Toc473791912)

[2.7 Table/Record Structures 11](#_Toc473791913)

[2.8 Data Model 11](#_Toc473791914)

[2.9 SQL commands 11](#_Toc473791915)

[2.10 Classes and Objects 11](#_Toc473791916)

[2.11 UI Design 11](#_Toc473791917)

[3. Technical solution 12](#_Toc473791918)

[4. Testing 12](#_Toc473791919)

[4.1 Introduction 12](#_Toc473791920)

[4.2 Normal Data 12](#_Toc473791921)

[4.3 Extreme Data 12](#_Toc473791922)

[4.4 Erroneous Data 12](#_Toc473791923)

[5. Evaluation 12](#_Toc473791924)

[5.1 Overview 12](#_Toc473791925)

[5.2 Review of Objects 12](#_Toc473791926)

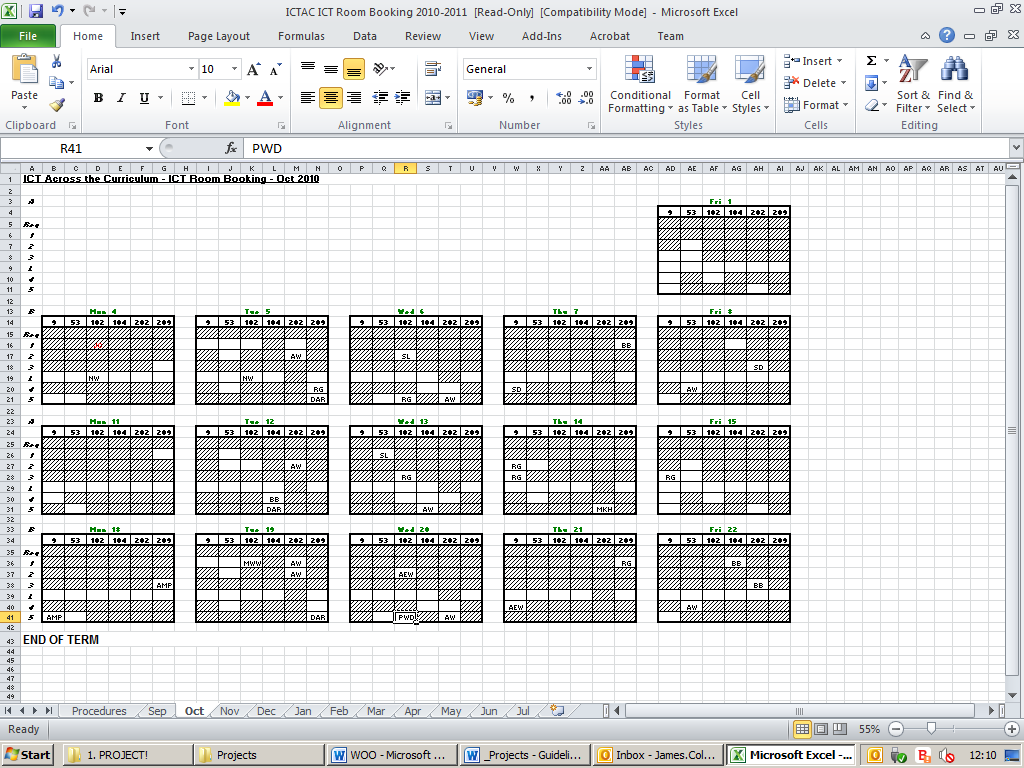
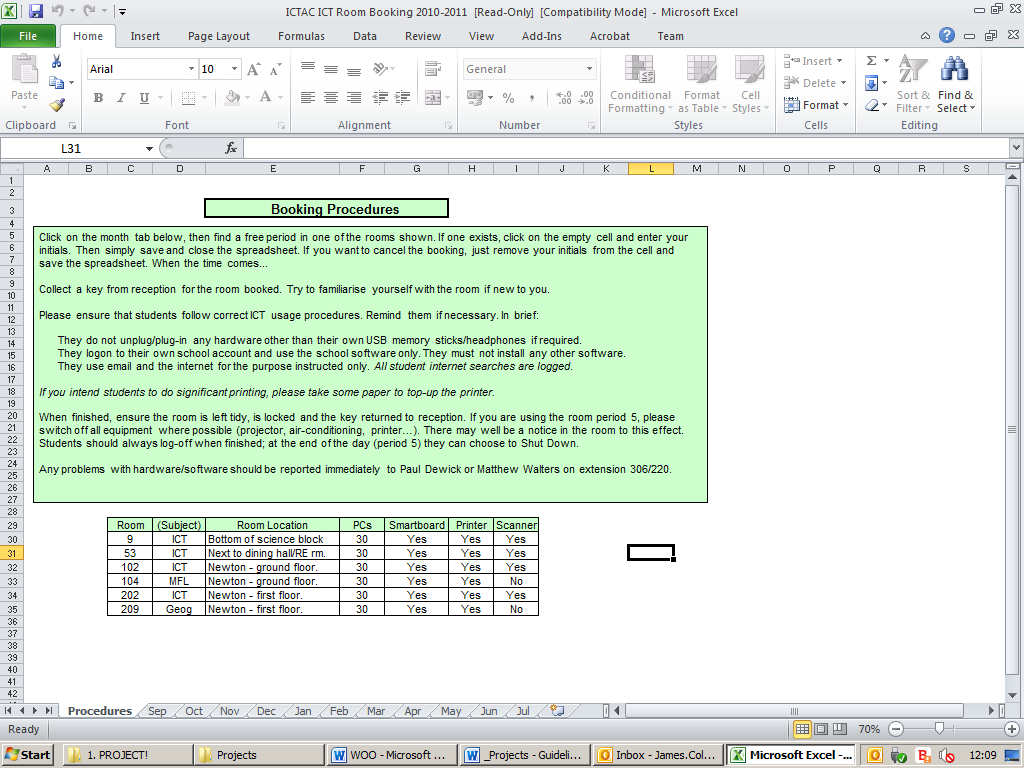
[5.3 User Feedback 12](#_Toc473791927)

[5.4 Improvements 12](#_Toc473791928)

# 1. Analysis

## Background

The Program I am planning on writing is a computer room booking/timetabling program for The King’s school Grantham. The school currently has about one thousand pupils and 70 teachers. The school was the school which Isaac Newton attended and has since expanded. This is the main reason for creating the program, the school doesn’t currently have a decent system to book rooms and as the school expands the current way they reserve rooms will become inadequate. The current system they use it just a simple spreadsheet. This works to a certain degree however it does have a fair amount of problems; firstly, the users have to manually search through each month to find their bookings themselves. The system also means that only one person can access it at any one time. Furthermore, any repeat booking like a club at lunchtime has to manually be entered on all their slots throughout a year. The spreadsheet also means that if a person closes it without saving their booking is lost and then this could lead to double bookings. Finally one rather large problem with the current system is that one person may leave the spreadsheet open over a lunchtime while their pc is locked and therefore stopping any other teacher making a booking. I am making one to allow teachers to see when or if a computer room is free. The school currently has nine it equipped rooms. Each room has a slightly different number of machines and other facilities (printer, Scanner, Whiteboard) and therefore the other facilities should be listed in the room booking so the teachers know if they can perform their desired lesson in this room or not.

The current system the school uses it is a simple spreadsheet that a teacher opens, finds a free cell on a confusing book system and puts their initials in the box. However this means that someone can change the bookings after someone has already reserved the room. The current system is fairly unfriendly and the instructions could be improved.

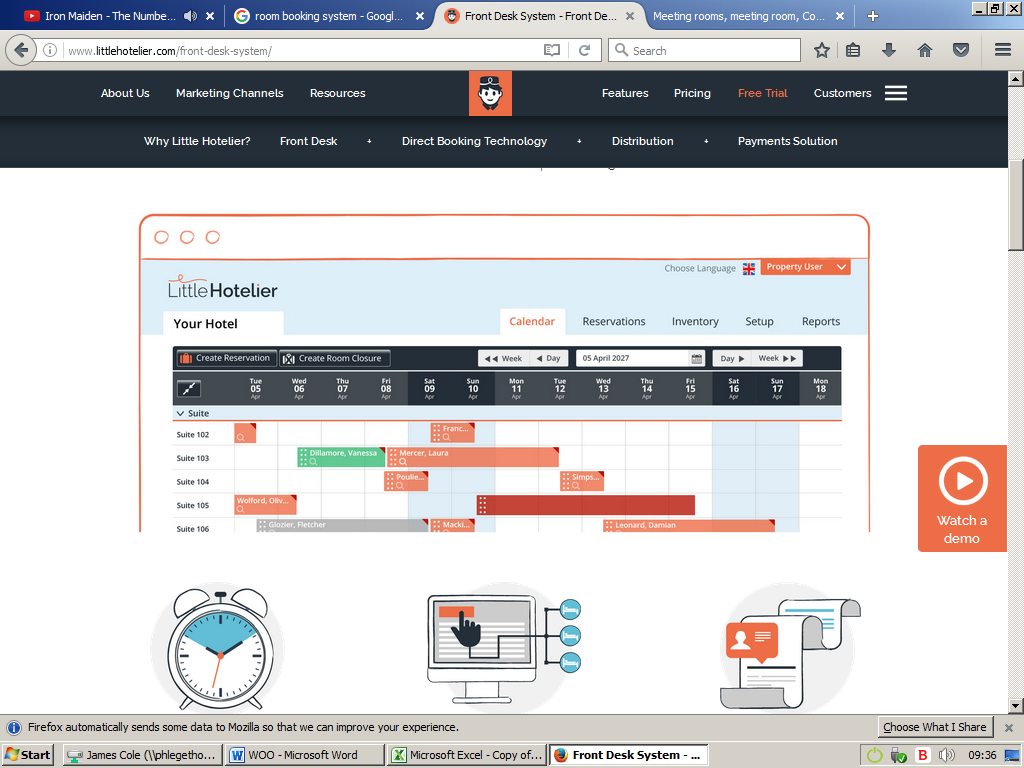
## 1.2 Investigation

The current program is just an excel spreadsheet. The spreadsheet is saved in a shared “T:” drive for all the teachers to be able to access it. The main problem with the main system is that it doesn’t have a very friendly/clear UI and way to actually book the rooms. Therefore I was going to use VB.net to create a nice UI and it also allows me to use the SQL commands to link it to an access file with tables linked to each room. Vb.net would also allow me to still save the program to the shared drive for ease of access. Also I could add multiple logins to grant privileges to certain users and only allow certain accounts to remove/ change bookings directly. Furthermore, I could use the school’s Email network to connect the teachers directly from the program and notify teachers when their room booked and by who.

There are certain programs already like this, however they are used in an office environment and each of the rooms have to run as separate version of the software. They also don’t tell you who has the room booked.

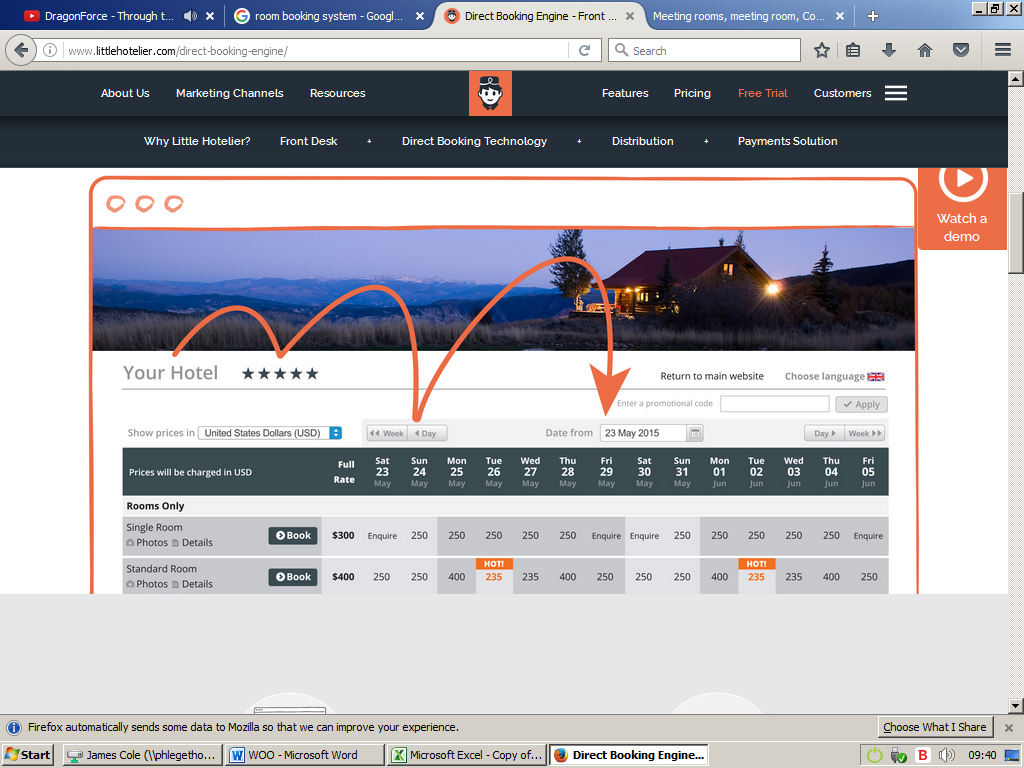
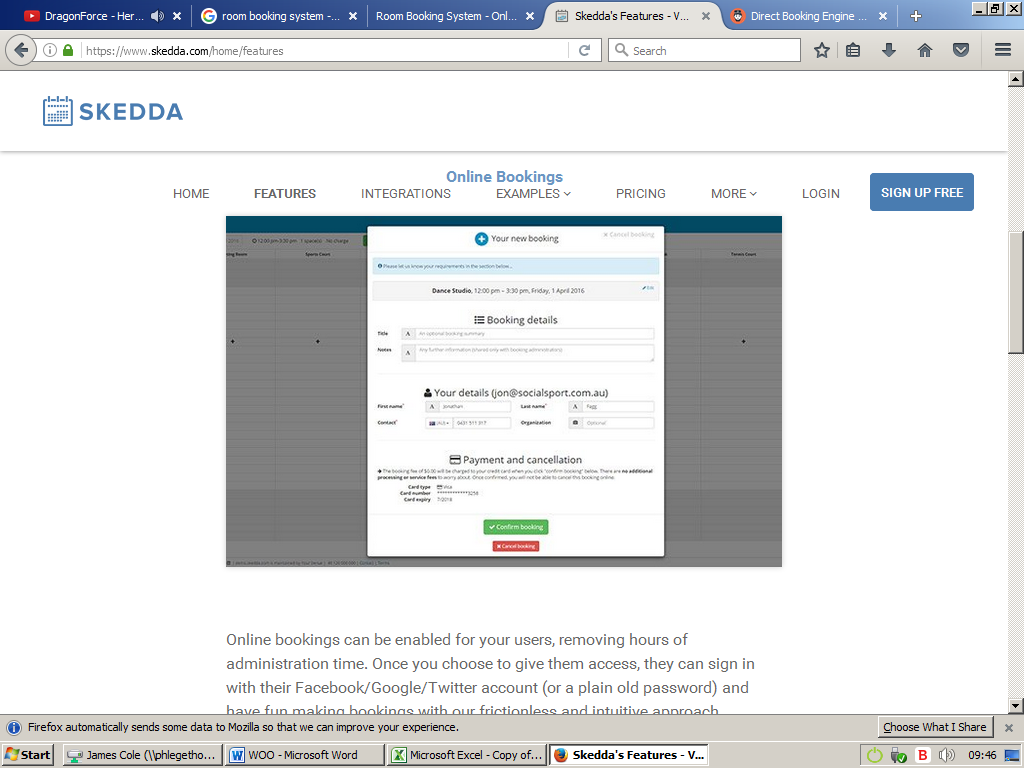
I conducted an interview with Mr.Greenhalgh on the 9th of January 2017 at about 1:00. In the interveiw we covered the main points of the old system and ways in which it should be improved. The points we came up with are**; *To make sure it was used for specifically ict room bookings as teachers likely using this system already have acces to a normal room it just lacks the facilities, A more friendly user experience , remove problems such as removeing a teacher’s current it suit booking to replace it with your own, an affective method of organising a room change, and an easy way of checking the room has all the additional hardware required for the lesson***. We also ***talked about having a system that multipul people could use at the same time to eliminate the possibility of one user using the spread sheet for a long time and stopping other people using it***. Mr. Greenhalgh was also fairly adiment that the ***schools colours and logo were a prominent feature of the program to make it more personalised.***

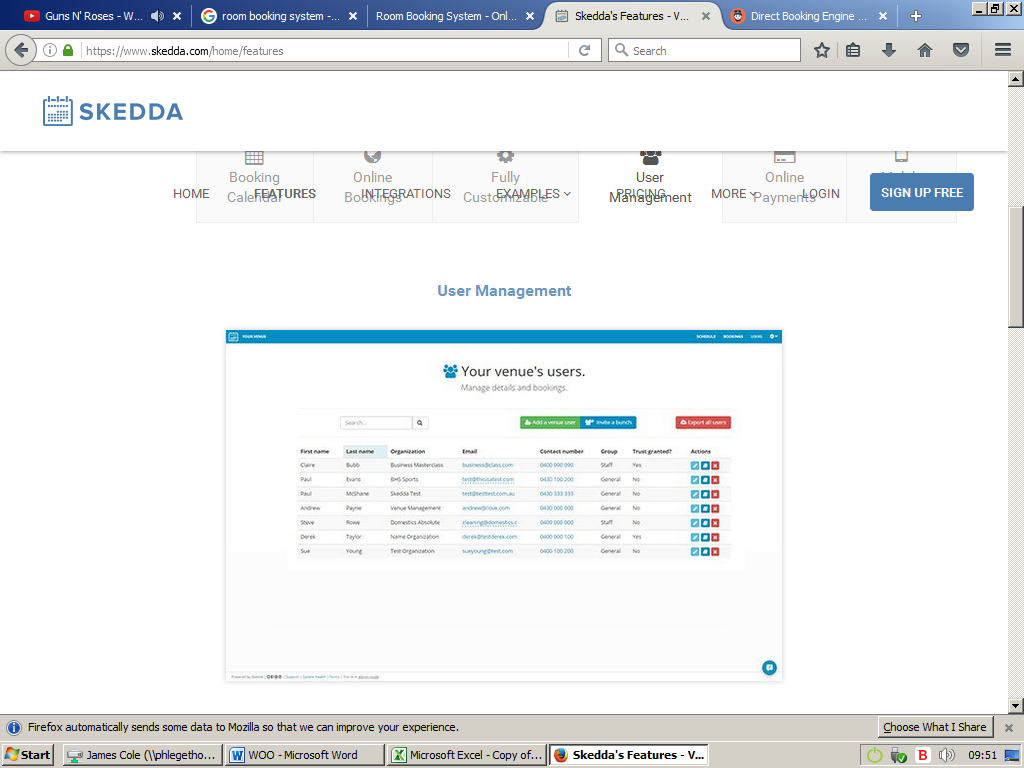
One of the main reasons the school want a new system is to accomidate new rooms being added to the school, and an easier way of adding/removing the rooms to the program rather than manually adding/removing them across the whole calendar.

**Examples**

Some examples of when computers are used to book things is a seat booking program on planes. Tables in a restaurant and maybe the most relevant to me are a room booking system for a hotel.

In all examples, there are a limited number of places, this is very important to do to prevent double booking. Also a very important thing to make sure is that the program is in real time, so it processes any bookings immediately so a room is booked to the first person who requested to book, rather than allowing the room to be double booked and having to cancel one booking.

This is a good example of what I would like my final program to look like. This layout makes it easy to manage which room is booked and when. This is an example of a hotel booking system, the first screenshot is what the program looks like for anyone working behind the desk, so therefore customers see a different screen. However I don’t feel this would be required as the bookings are internal. Furthermore I am a fan of the gant chart style however the pure table look is easier to achieve in visual basic.net.



Equally along side this layout of program, I like the idea of filling out a booking form so if anyone needs to find out who has booked a room, they click the full boooking and the person’s details come up. This is what a form could look like, however I would like to include the King’s school logo on the program somewhere and to use the school colours on the program to make it feel a little bit more like the schools’ program rather than just a 3rd party software they purchased. But that’s just a cosmetic thing.

Annother aspect the program could have would be a users page which manages who is using the program and what their role on the program is, so this would allow admin to manage who is using the system and allow them to create/ remove users as they require. Again this is a page I would like to change slightly but keep the main parts as they are. Adding the logo and changing the colours.

This next program is perhaps the closest thing I could find to the program I would like to create. It manages It room bookings within a school. However you can change it so it displays any free room in the school. The only thing that this program doesn’t do as much as I would like it to would be the room’s information before a person books the room. This program looks a lot nicer than the others yet they all have similarities in how they look, they are all very minimalist, so I need to make sure that when I design my program I keep it very simple and not to cluttered with buttons and other features. Annother feature this program offers is reoccuring room bookings, so this would be something I would like to add as then an individual doesn’t have to spend loads of time booking their room every week for the whole year, instead (using a club as an example) whist club could book their room for the whole year on a Wednesday lunchtime.

 However then this could turn into a room booking system rather than just an it suite booking sustem. However this could be faily easy to manage as all I would have to do is add more rooms and their specifications to the system and add filters to filter out the types of room you would like to book or even a filter that allowed you to select which building in the school you would like the room. One feature that I havent seen is allowing teachers to communicate through the application directly to eachother to request a room change/ booking change.

There are some programs which are a bit more cluttered but yet they still are easy to navigate. However I don’t think this look is as proffesional and it looks faily basic.

**Second interview**

I then conducted a follow up interview with mr.Greenhalgh to see if I was heading in the right direction, he said to me that there were a few more things he wanted in the final program, for example ***a feature which automatically told the office that there was a room change so they were also aware of it***. We ironed out the final main points that he considered the most important, these were that the ***program has to be able to save bookings and cancel bookings ( he added the fact this should also be extended to multipule bookings). The ability to add/ remove rooms from the program if a new room is built to cater for the expanding school. Look nice and be easy to use. Have a reminder system to notify staff when they have a booking coming up.*** We also drew up a few ideas for the design of the program and we both agreed on one that was clear and fairly minimalist.

## 1.3 Program Requirements

The new program must:

1. Save who booked the room.
2. A user must be able to Cancel individual bookings made else a room may appear booked when it’s not.
3. The Admin need the ability to Add/remove rooms if the school expands or an IT room is repurposed.
4. The program should Store a room’s timetable which contains information about when it is booked or when it is free.
5. The program needs the ability to Add/remove users.
6. Make multiple bookings
7. Have a Friendly / easy to use UI
8. Show each user when their next reservation is
9. Have a reminder system which reminds users when their booking is “A day away”
10. Have different user accounts I.e. - an admin account and a few normal ones which are individually different.
11. Be updated in real time
12. Allow for term dates to be entered and used during booking
13. Make it so only admins/ the person who made the booking can cancel bookings

## 1.4 Data Volumes

|  |  |  |
| --- | --- | --- |
| Record | Currently Stored | Input Weekly |
| Rooms | 7 | 0 |
| Bookings | 250 | 20 |
| Teachers | 70 | 0 |
| Week | A/B | 0 |
| Time | 8 | 0 |
| Information about room | 5 | 0 |

\*If a new room is added, it is added at the start of a new year over the summer term

## 1.5 Data Dictionary

|  |  |  |
| --- | --- | --- |
| Data To be stored | Description | Example |
| Teacher name | First and last name of teachers who wish to book the room. | Andrew Payne |
| Teacher ID | A three digit number assigned to each teacher. | 001 |
| Teacher Subject | Save the subject the teacher teaches so if a person wants to see which department is most active or has the most current active bookings they can do. | ICT,Geography,Computing,MFL; ETC |
| Room Timetable | Which lessons are free for each specific room. |  |
| Teacher timetable | A timetable of two weeks that each teacher has and shows them which lesson they would wish to apply for a computer room. |  |
| A teacher’s specific bookings | Store the booking a teacher has made and join the bookings tables and the teacher id in an access table. | Period 1 Monday week a, period 4 Thursday week B |
| Room information | Does the room have a scanner, printer, smart board, and how many PCs does the room have. | Yes, Yes , No, 30 |
| When is the booking | Store room number, period, date and which week the room is booked in. |  |
| Where in the school is the booked room | Store where in the school the booked room is so new teachers can find their newly booked room. | Bottom floor of the Newton Block |

## 1.6 Entity Relationship diagram

1

M

M

1

Active bookings

Rooms

Teacher s

## Data Flow Diagram

**Level 0 data flow diagram for both the Old System and the New System**

**Level 1 Data flow diagram for the old spreadsheet system**

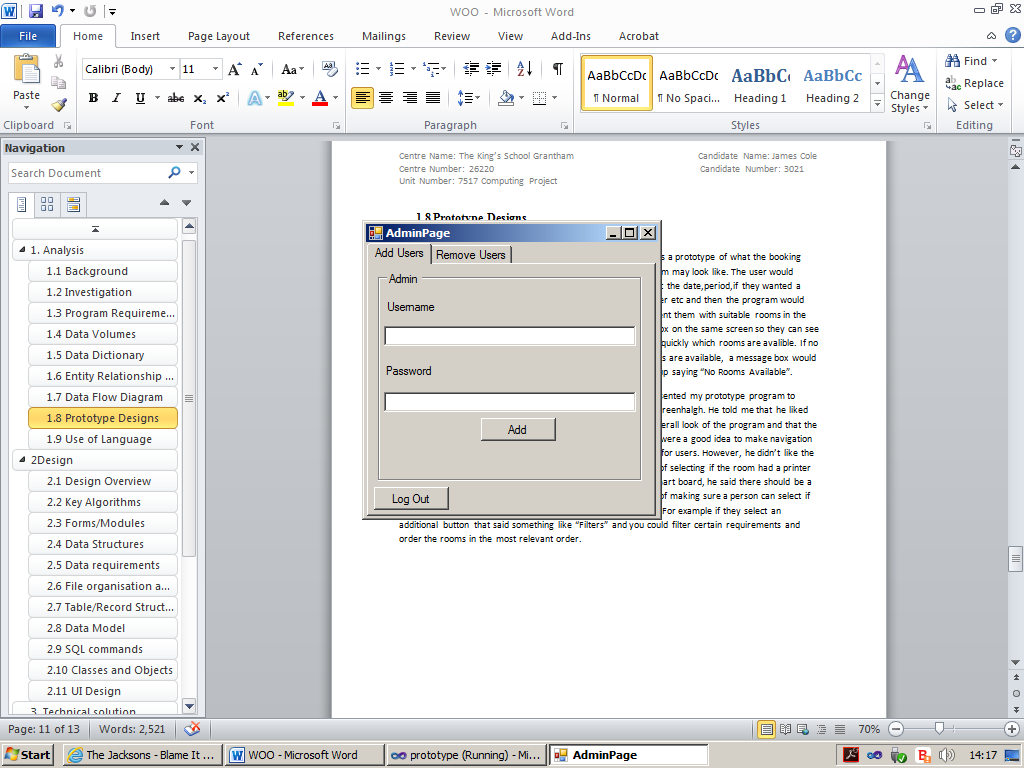


**Level1 Data flow diagram for the new system**

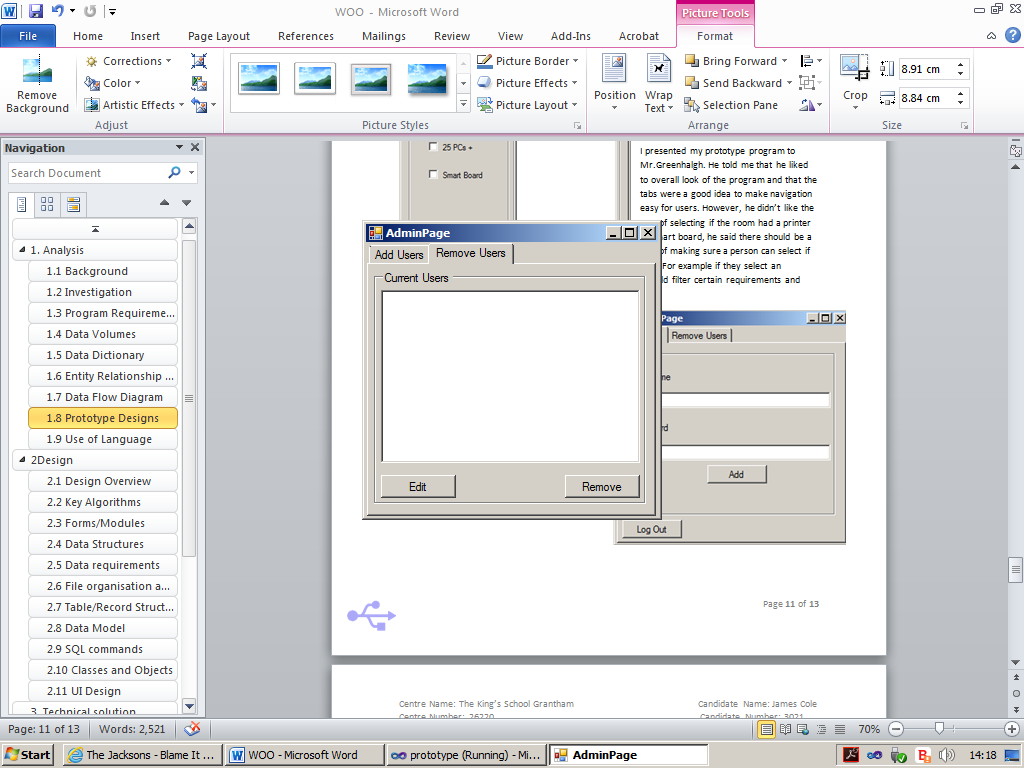


## Prototype Designs

This is a prototype of what the booking system may look like. The user would select the date,period,if they wanted a printer etc and then the program would present them with suitable rooms in the list box on the same screen so they can see very quickly which rooms are avalible. If no rooms are available, a message box would pop up saying “No Rooms Available”.

I presented my prototype program to Mr.Greenhalgh. He told me that he liked to overall look of the program and that the tabs were a good idea to make navigation easy for users. However, he didn’t like the way of selecting if the room had a printer or smart board, he said there should be a way of making sure a person can select if they want to be more picky with the room’s specific hardware. For example if they select an additional button that said something like “Filters” and you could filter certain requirements and order the rooms in the most relevant order.

I added the Admin page function that isnt accessable to normal users as it requires a different login. From this page admin users can add new users and monitor the current users, i.e change names if someone gets married or remove users if a member of staff leaves.

Mr.Greenhalgh liked this idea as it was an eligant way of organising the staff who use the program, he did however say that I need to make sure that it is only the admin who can use this page and not teachers by mistake, or even students as they could mess a lot up within the school.

## Use of Language

For this project I will be using VB.net as it enables me to visually design my program so it is easy to use for the users. The language is also high level and “English like”. Furthermore I can create Csv files that contain user information and Room information. It allows me to filter results into a list if I need to. In addition the program has in built encryption on data saved to files, this will be useful for saving usernames and passwords of users anonymously.

The school use fairly standard PCs so I will have to make sure the program isn’t too demanding. This is another reason for using Vb.net as it creates easy to run programs that are well optimised. Overall the reason for using this language is because it is easy to code in and doesn’t take very long to create a fairly substantial and versatile program. It also has a lot of inbuilt features that I can utilise to save making them and potentially having to do a lot of troubleshooting to find and potential problems.

# 2. Design

## 2.1 Design Overview

The program will be broken down into a top down structure where any clear inputs and outputs are used to help me create a well-structured program. I am using VB.net as it is very good with file handling and it allows me to create a very visually accurate program because it allows me to see the look of the program as I am working on it. It also allows me to tweak it quickly if the client isn’t happy with how it looks. In addition, it allows the client to have their input listened to as designing the UI doesn’t need a lot of computing knowledge and can therefore be done very easily to a high standard.

I will use a few menus as it allows the program to look very clean when they are not being used but adds functionality when the user clicks on one. However, I must be careful not to use to many as to make the screen feel cluttered and untidy.

I will be using random access files and user defined types(records) as these will help to structure my own records with the correct fields/ types. In addition, random access files are faster and can store a huge number of records. Furthermore, they can be edited more easily.

My program is Split into forms and then within those the key functions are split into subroutines. Any frequently accessed subroutines are then placed into a module so can be accessed program wide. This means that my program is easy to change and or update in the future if new features need adding.



**System Flow Charts**

Rooms

Rooms

Disk

Disk

Check and Save

Collect and resolve mistakes

Staff

Collect and resolve mistakes

Staff

Disk

Disk

Check and Save

Timetable

Disk

Disk

Timetable

Check and Save

Collect and resolve mistakes

Booking

Book Room

Booking

Disk

Disk

Check room Availability

Check and Save

Collect and resolve mistakes

Booking

Disk

Disk

Booking

Email

Email

Check and Save

Cancel

Confirmation

Email

Email

## 2.2 Key Algorithms

**Algorithm for Adding/Updating information in a file (Teachers file)**

Open Teacher file

TeacherID = TextBox1.Text

TeacherName = TextBox2.Text

Subject =TextBox3.Text

If currentrecord > 0 Then

Write teacher record to file at current record

Else

Calculate new rec pos as length of file / rec len

Write teacher record to file at new record pos

End If

**Algorithm for Adding/Updating information in a file (Teachers file)**

Open Teacher file

TeacherID = TextBox1.Text

TeacherName = TextBox2.Text

Subject =TextBox3.Text

If currentrecord > 0 Then

Write teacher record to file at current record

Else

Calculate new rec pos as length of file / rec len

Write teacher record to file at new record pos

End If

**Algorithm for Deleting a record in a file**

open File

Primary key = ""

Write the data at the current record

Close file

MsgBox("Deleted")

End If

**Algorithm for blanking out the table**

Dim TestDate As Date

TestDate = Label1.Text

For row = 1 To 5

If TestDate < Today.Date Then

Grid3.Row = row

For i = 1 To 8

Grid3.Col = i

Grid3.CellBackColor = Color.Gray

Next

End If

TestDate = TestDate.AddDays(1)

Next

**Algorithm for Searching for a record in a file**

Open File

currentrecord = 0

calculate the number of records Length of the file / length of record

found = False

For 1 To numrecords

Read data from the file and current record

Output left most data

Output the next part of the data

Output the last part of the data

currentrecord = recordcounter

found = True

Exit For

End If

Next

Close file

If Not found Then MsgBox("Item Not Found. Try Again")

## 2.3 Forms/Modules

|  |  |
| --- | --- |
| Forms/Modules | Description |
| FrmLogin | The first page to open up when you start the program. It contains username and password boxes for specific users to log into the program. |
| FrmStartup | The first page a normal user comes to Which allows them to navigate through the program. This screen is going to be split into tabs. Some of the tabs that are currently there are the welcome page and the make a booking tab, the names of the tabs basically explain the function of the form that the user would access by clicking on it. |
| FrmAdmin | This is the page that will only be accessed when an admin username and password it inserted. This form is again spilt into tabs. The first tab the admin will show the admin a welcome page with not a lot in the way of functionality on the start tab. There will then be tabs dedicated to inserting a timetable for a year, adding/removing users and any other tabs that I feel the admin should be able to change within the program. |

## 2.4 Data requirements

The files I would need are:

**Teachers** (**Teacher Name**, *TeacherID*,Subject,NormalRoomNo.)

**Rooms** (*Room No.*, Location, **Printer**, **NumberOfPCs**)

**Bookings** (*Booking Reference*, **Teacher ID**, **Room**, **date**, **Period**, Repeat (Y or N))

**Logins** (*Username*, **Password** (Hopefully encrypted))

**Timetable** [specific room number] (When the room is Free)

## 2.5 File organisation and processing

|  |  |  |  |
| --- | --- | --- | --- |
| **File** | **Access** | **Description** | **Master/Transaction** |
| Teachers | Random | Stores teachers’ information, Their ID,their name,Their subject and their normal room number | Master |
| Rooms | Text | Stores room information such as the rooms location within the school. | Master |
| Bookings | Text | Stores the information surrounding a specific book a teacher makes | Master |
| Logins | Random | Stores teachers’ specific and individual logins. | Master |

I am aiming to make an online program so it is all updated in real time when there is an input the master files are changed directly. So there aren’t any times when the program does mass processing of data at none peak times.

## 2.6 Table/Record Structures

Adding a teacher

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Feild No. | Data Item | Data Type | Length | Format | Source | Validation Required | Remarks |
| 1 | Name | Text | 30 | First name surname | Admin Page | Admin login |  |
| 2 | Subject | Text | 18 | Subject name | Admin Page | Admin login |  |
| 3 | Username | Text | 10 | First initial. Surname | Admin Page | Admin login |  |
| 4 | Password | text | 50 |  | Admin Page | Admin login | No set format for teacher password so it can be anything they want. |

Adding a Room

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Field No. | Data Item | Data Type | Length | Format | Source | Validation Required | Remarks |
| 1 | Room number | Number | 3 | 000 | Admin Page | Admin Login | Anything under three digits is stored as 0XX just to make it a unifom size |
| 2 | Printer? | Boolean | 8 | YPrinter/Nprinter | Admin Page | Admin Login |  |
| 3 | Number of PCs | Number | 2 | 00 | Admin Page | Admin Login |  |

An example of a Booking File

Saved in a text file and the different letter correspond to whether or not the room is Free. “F” is used when the room is free and it alerts the program to this fact when it is loaded into the flex grid.

FXXFXXXFXXXXXXXXXFXXXXXXXXXXXXFXX

The availability is shown by using a different colour in the Grid.

## 2.9 Data Model

1

M

M

1

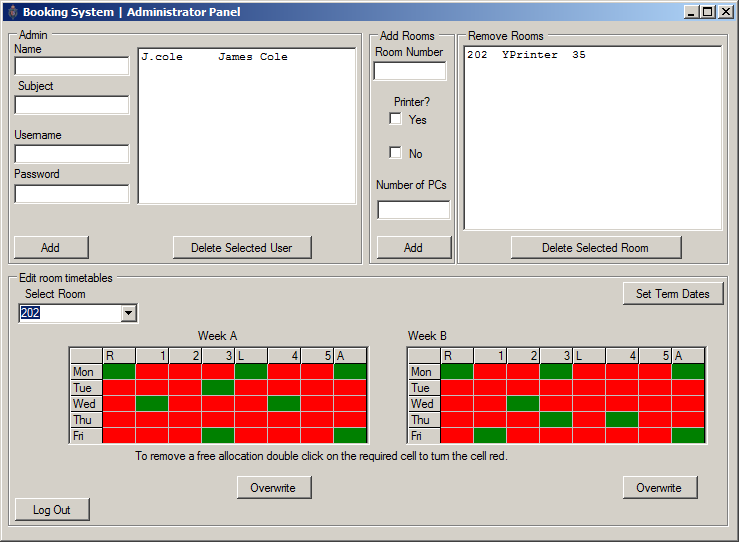
Active bookings

Rooms

Teacher s

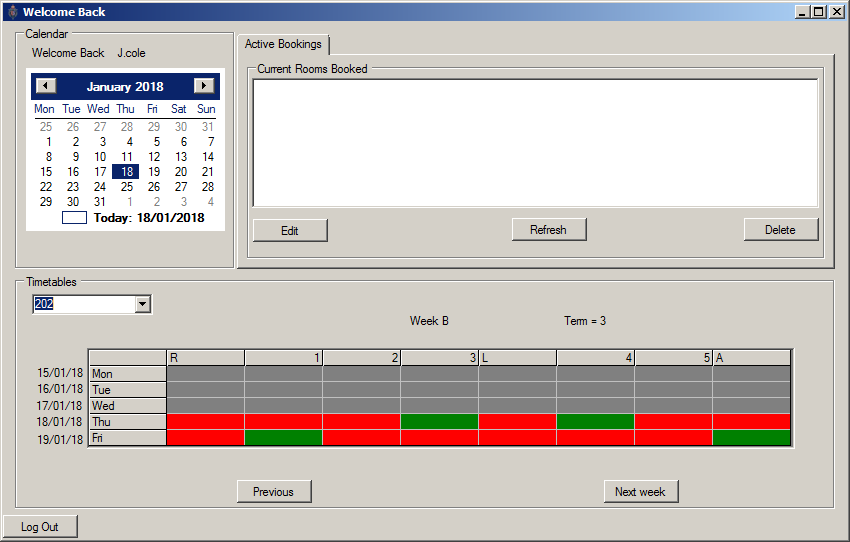
## 2.10 UI Design

This is the Admin page. You can only get onto this page with the admin login. This means that normal users can’t change information which is important to the operation of the program.



This is the Administrator page and it is used for adding and removing both users and rooms. You do this by using the three group boxes at the top.

From this page you can also change the term dates which dictates when you can book rooms throughout the year.

The large section at the bottom of the page is for setting the timetables for both weeks A and B. this is the largest and most important part of the program so I thought it was only right that it was the largest section on the page. Each week is separate and colour coded into the periods that are free (GREEN) and those which aren’t (RED). This makes it easy to see at a glance which room is free on which period.

This is true on both pages as both the admin and the normal users need to be able to see the availability at a glance. The main page for users has all the information they will need on a daily basis. The calendar in the top left of the screen allows users to set appointments for the day and they are allowed to set as many as they need for the future. Furthermore, the list box on the right of the screen displays any live bookings which the specific user has. The grid control with the buttons to change weeks allows for a seamless transition between weeks. In addition to this I tried to streamline booking by making the process of booking a room take less than 10 seconds. Simply click on the period you want. Press yes and its done.

# Technical solution

**Term Dates Page:**

Declaring all the variables for the Class.

Option Explicit On

Public Class Term\_Dates

Dim TermOneStart, TermOneEnd, HOLIDAY1, HOLIDAY1END, TermTwoStart, TermTwoEnd, TermThreeStart, TermThreeEnd, TermFourStart, TermFourEnd, TermFiveStart, TermFiveEnd, TermSixStart, TermSixEnd As Date

Private Sub Button1\_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click

Me.Hide()

Hides the current window. Same as a “Back” Button.

End Sub

Public Sub SetWeeks()

Dim TermOneFirst As String = "A"

Dim first As String = ""

Dim last As String = ""

Dim Previous As String = ""

Sets the weeks as A or B depending on term dates and the weeks in the year.

Dim NXT As String = ""

If last = "B" Then first = "A"

If last = "A" Then first = "B"

If Previous = "B" Then NXT = "A"

If Previous = "A" Then NXT = "B"

Weeks = DateDiff(DateInterval.WeekOfYear, TermOneStart, TermOneEnd)

MsgBox(Weeks, TermOneFirst)

End Sub

Public Sub Setdates()

TermOneStart = DateTimePicker1.Text

TermOneEnd = DateTimePicker2.Text

TermTwoStart = DateTimePicker3.Text

TermTwoEnd = DateTimePicker4.Text

Assigns the dates on the form to the right term. If it’s the start of term one or the end of term three ETC.

TermThreeStart = DateTimePicker5.Text

TermThreeEnd = DateTimePicker6.Text

TermFourStart = DateTimePicker7.Text

TermFourEnd = DateTimePicker8.Text

TermFiveStart = DateTimePicker9.Text

TermFiveEnd = DateTimePicker10.Text

TermSixStart = DateTimePicker11.Text

TermSixEnd = DateTimePicker12.Text

If TermOneEnd < TermOneStart Then

MsgBox("Please enter a valid date for the end of term one")

ElseIf TermTwoEnd < TermTwoStart Then

MsgBox("Please enter a valid date for the end of term two")

ElseIf TermThreeEnd < TermThreeStart Then

MsgBox("Please enter a valid date for the end of term three")

ElseIf TermFourEnd < TermFourStart Then

MsgBox("Please enter a valid date for the end of term three")

ElseIf TermFiveEnd < TermFiveStart Then

MsgBox("Please enter a valid date for the end of term four")

ElseIf TermSixEnd < TermSixStart Then

MsgBox("Please enter a valid date for the end of term six")

Saves these dates to a file.

Else

Dim myfile As New System.IO.StreamWriter("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\termdates.txt")

myfile.Write(TermOneStart & "." & TermOneEnd & "." & TermTwoStart & "." & TermTwoEnd & "." & TermThreeStart & "." & TermThreeEnd & "." & TermFourStart & "." & TermFourEnd & "." & TermFiveStart & "." & TermFiveEnd & "." & TermSixStart & "." & TermSixEnd)

myfile.Close()

myfile.Dispose()

End If

End Sub

Private Sub Button3\_Click(sender As System.Object, e As System.EventArgs) Handles Button3.Click

Runs the sub routine which saves the term dates.

Setdates()

End Sub

End Class

**Admin Page Code:**

Public Class AdminPage

Dim currentrecord As Integer

Dim reclen As Integer = Len(Room)

Dim Room As Room

Declaring all the variables for the Class.

Dim Roomcodes(0 To 100) As String

Dim recnums(0 To 100) As Integer

Dim activeroomitems As Integer

Public Sub LoadTeachers(listbox2 As ListBox)

listbox2.Items.Clear()

Loading the teachers able to use the program into a listbox using a file which stores the teachers’ information.

Dim Teacher As New login

Dim RecLen As Integer = Len(Teacher)

Dim NumRecords As Integer

Dim RecordCounter As Integer

Dim activeitems As Integer = 0

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\logins.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, RecLen)

NumRecords = LOF(1) / RecLen

For RecordCounter = 1 To NumRecords

FileGet(1, Teacher, RecordCounter)

If Trim(Teacher.username) <> "" Then

listbox2.Items.Add(Teacher.username & " " & Teacher.Name & " ")

End If

Next

FileClose(1)

End Sub

Public Sub Logins()

Dim logins As New login

Dim currentrecord As Integer

Dim Reclen As Integer = Len(logins)

Dim NewRecordPos As Integer

Dim FileNum As Integer = FreeFile()

FileOpen(FileNum, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\logins.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, Reclen)

logins.username = TextBox1.Text

logins.password = TextBox2.Text

logins.Name = TextBox5.Text

logins.Subject = TextBox6.Text

Code for adding users to the program and assigning them a username and password.

If currentrecord > 0 Then

FilePut(FileNum, logins, currentrecord)

Else

NewRecordPos = LOF(FileNum) / Reclen + 1

FilePut(FileNum, logins, NewRecordPos)

End If

FileClose(FileNum)

TextBox5.Text = ""

TextBox6.Text = ""

TextBox1.Text = ""

TextBox2.Text = ""

End Sub

Private Sub Button1\_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click

Me.Hide()

Another back button.

StartUpScreen.Show()

StartUpScreen.Button3.Visible = False

End Sub

Private Sub Button3\_Click(sender As System.Object, e As System.EventArgs) Handles Button3.Click

If TextBox4.Text = "" Then

MsgBox("please enter a room number")

ElseIf CheckboxNo.Checked = False And CheckboxYes.Checked = False Then

MsgBox("Please State if the room has a printer")

Validating that the correct room information is filled in.

ElseIf Val(TextBox3.Text) < 1 Then

MsgBox("Enter no. of pcs")

Else

Dim currentrecord As Integer

If everything is ok. Add the room to the Room file.

Dim Room As New Room

Dim Reclen As Integer = Len(Room)

Dim NewRecordPos As Integer

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Rooms.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, Reclen)

Room.Number = TextBox4.Text

Room.pcs = TextBox3.Text

If CheckboxYes.Checked = True Then Room.printer = "YPrinter"

If CheckboxNo.Checked = True Then Room.printer = "NPrinter"

If currentrecord > 0 Then

FilePut(FreeFile, Room, currentrecord)

Else

NewRecordPos = LOF(1) / Reclen + 1

FilePut(1, Room, NewRecordPos)

FileClose(1)

End If

FileClose(1)

FileClose(1)

This makes sure the new room can be used to be assigned a time table by refreshing the rooms displayed in the combobox.

PopulateComboBox(ComboBox1)

TextBox4.Text = ""

TextBox3.Text = ""

LoadRooms(ListBox1, Roomcodes, recnums, activeroomitems)

FileClose(1)

ComboBox1.Items.Clear()

LoadRooms(ListBox1, Roomcodes, recnums, activeroomitems)

LoadTeachers(ListBox2)

PopulateComboBox(ComboBox1)

SetGrid()

End If

End Sub

Public Sub LoadRooms(listbox1 As ListBox, codes() As String, recnos() As Integer, ByRef itemsactive As Integer)

listbox1.Items.Clear()

Dim Room As New Room

Dim RecLen As Integer = Len(Room)

Dim NumRecords As Integer

Dim RecordCounter As Integer

Dim activeitems As Integer = 0

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Rooms.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, RecLen)

NumRecords = LOF(1) / RecLen

Loads all the rooms into the listbox next to the place the room is entered.

For RecordCounter = 1 To NumRecords

FileGet(1, Room, RecordCounter)

If Trim(Room.Number) <> "" Then

listbox1.Items.Add(Trim(Room.Number) & " " & " " & Trim(Room.printer) & " " & Trim(Room.pcs))

codes(activeitems) = Trim(Room.Number)

recnos(activeitems) = RecordCounter

activeitems += 1 ' the same as activeitems= activeitems+1

End If

Next

For i As Integer = 0 To listbox1.Items.Count - 1

Removes the room that is selected in the listbox.

If CStr(listbox1.Items(i)) = String.Empty Then

listbox1.Items.RemoveAt(i)

i -= 1

End If

Next

FileClose(1)

itemsactive = activeitems

End Sub

Private Sub AdminPage\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Loads the rooms into listbox by passing the room codes. Number of records and activerooms.

ComboBox1.Items.Clear()

LoadRooms(ListBox1, Roomcodes, recnums, activeroomitems)

LoadTeachers(ListBox2)

PopulateComboBox(ComboBox1)

Loads teaches and rooms. Also sets the grids on the page. Ready for timetables.

SetGrid()

End Sub

Private Sub Button6\_Click(sender As System.Object, e As System.EventArgs) Handles Button6.Click

Dim TempRoomNum As String

Dim response As Integer

Selects a room and then deletes all data associated with the room.

Dim RoomNo As String

Dim RecordNumber As Integer

RoomNo = ListBox1.SelectedItem

If ListBox1.SelectedItem = "" Then

MsgBox("Please Select a room to delete")

Else

response = MsgBox("This will delete the room from the file and remove any timetabled reservations.Are you Sure?", MsgBoxStyle.Question + MsgBoxStyle.YesNo, "Delete Record")

If response = vbYes Then

TempRoomNum = Microsoft.VisualBasic.Left(ListBox1.SelectedItem, InStr(ListBox1.SelectedItem, " ") - 1)

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Rooms.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, reclen)

For RecordNumber = 1 To LOF(1) / Len(Room)

FileGet(1, Room, RecordNumber)

If Trim(Room.Number) = TempRoomNum Then

Room.Number = " "

FilePut(1, Room, RecordNumber)

End If

Blanks the room number out in the file and removes it from the listbox.

Next

ListBox1.Items.Remove(RoomNo)

FileClose(1)

MsgBox("Deleted")

Runs the sub routine called deleteroom which removes the timetables and active bookings.

DeleteRoom()

End If

End If

PopulateComboBox(ComboBox1)

End Sub

Private Sub Button2\_Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click

If TextBox1.Text <> "" And TextBox2.Text <> "" Then

Makes sure that there is data being entered for new users.

Call Logins()

LoadTeachers(ListBox2)

Else

MsgBox("Please enter username and password for the new User")

End If

End Sub

Private Sub Button7\_Click(sender As System.Object, e As System.EventArgs) Handles Button7.Click

Dim response As Integer

Dim username As String

Deletes a user from the program by selecting them from the listbox.

Dim RecordNumber As Integer

Dim found As Boolean = False

Dim record As login

reclen = Len(record)

response = MsgBox("Are you Sure?", MsgBoxStyle.Question + MsgBoxStyle.YesNo, "Delete Record")

If response = vbYes Then

username = Trim(Microsoft.VisualBasic.Left(ListBox2.SelectedItem, 10))

ListBox2.Items.Remove(username)

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Logins.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, reclen)

For RecordNumber = 1 To LOF(1) / Len(record)

FileGet(1, record, RecordNumber)

If record.username.Trim = username Then

record.username = " "

FilePut(1, record, RecordNumber)

MsgBox("Deleted")

found = True

Exit For

End If

Next

End If

If found = False Then MsgBox("not found")

FileClose(1)

ListBox2.Items.Clear()

Refreshes the list of teachers.

LoadTeachers(ListBox2)

End Sub

Public Sub SetGrid()

GridA.Clear()

GridA.BackColor = Color.Red

GridA.set\_TextMatrix(0, 1, "R")

GridA.set\_TextMatrix(0, 2, "1")

GridA.set\_TextMatrix(0, 3, "2")

GridA.set\_TextMatrix(0, 4, "3")

Holds the properties for the grid on the page which represents week A. Sets the width of each cell, the colour of each cell and whether it has text in that cell.

GridA.set\_TextMatrix(0, 5, "L")

GridA.set\_TextMatrix(0, 6, "4")

GridA.set\_TextMatrix(0, 7, "5")

GridA.set\_TextMatrix(0, 8, "A")

GridA.set\_TextMatrix(1, 0, "Mon")

GridA.set\_TextMatrix(2, 0, "Tue")

GridA.set\_TextMatrix(3, 0, "Wed")

GridA.set\_TextMatrix(4, 0, "Thu")

GridA.set\_TextMatrix(5, 0, "Fri")

GridA.set\_ColWidth(0, 500)

GridA.set\_ColWidth(1, 500)

GridA.set\_ColWidth(2, 500)

GridA.set\_ColWidth(3, 500)

GridA.set\_ColWidth(4, 500)

GridA.set\_ColWidth(5, 500)

GridA.set\_ColWidth(6, 500)

GridA.set\_ColWidth(7, 500)

GridA.set\_ColWidth(8, 500)

GridB.Clear()

GridB.BackColor = Color.Red

GridB.set\_ColWidth(0, 500)

GridB.set\_ColWidth(1, 500)

GridB.set\_ColWidth(2, 500)

GridB.set\_ColWidth(3, 500)

GridB.set\_ColWidth(4, 500)

GridB.set\_ColWidth(5, 500)

GridB.set\_ColWidth(6, 500)

GridB.set\_ColWidth(7, 500)

GridB.set\_ColWidth(8, 500)

GridB.set\_TextMatrix(0, 1, "R")

GridB.set\_TextMatrix(0, 2, "1")

GridB.set\_TextMatrix(0, 3, "2")

GridB.set\_TextMatrix(0, 4, "3")

GridB.set\_TextMatrix(0, 5, "L")

GridB.set\_TextMatrix(0, 6, "4")

GridB.set\_TextMatrix(0, 7, "5")

GridB.set\_TextMatrix(0, 8, "A")

GridB.set\_TextMatrix(1, 0, "Mon")

GridB.set\_TextMatrix(2, 0, "Tue")

GridB.set\_TextMatrix(3, 0, "Wed")

GridB.set\_TextMatrix(4, 0, "Thu")

GridB.set\_TextMatrix(5, 0, "Fri")

End Sub

Public Sub PopulateComboBox(ComboBox As ComboBox)

ComboBox1.Items.Clear()

LoginScreen.ComboBox2.Items.Clear()

Dim Room As Room

Takes the rooms and loads them into a combobox which allows the user to select which timetable to display on the admin page.

Dim RecLen As Integer = Len(Room)

Dim NumRecords As Integer

Dim RecordCounter As Integer

Dim activeitems As Integer = 0

Dim FileNum As Integer = FreeFile()

FileOpen(FileNum, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Rooms.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, RecLen)

NumRecords = LOF(FileNum) / RecLen

For RecordCounter = 1 To NumRecords

FileGet(FileNum, Room, RecordCounter)

If Trim(Room.Number) <> "" Then

ComboBox.Items.Add(Trim(Room.Number))

End If

Next

FileClose(FileNum)

End Sub

Private Sub GridA\_ClickEvent(sender As Object, e As System.EventArgs) Handles GridA.ClickEvent

Makes the Cell colour change from red to green when the cell is clicked. Because that’s how to program decides if the room is free or not. In Week A

GridA.CellBackColor = Color.Green

End Sub

Private Sub GridB\_ClickEvent(sender As Object, e As System.EventArgs) Handles GridB.ClickEvent

GridB.CellBackColor = Color.Green

Does the same as above but for week B

End Sub

Private Sub GridA\_DblClick(sender As Object, e As System.EventArgs) Handles GridA.DblClick

Changes the colour back when double clicked. Week A

If GridA.CellBackColor = Color.Green Then

GridA.CellBackColor = Color.Red

End If

End Sub

Private Sub GridB\_DblClick(sender As Object, e As System.EventArgs) Handles GridB.DblClick

If GridB.CellBackColor = Color.Green Then

Changes the colour back when double clicked. Week B

GridB.CellBackColor = Color.Red

End If

End Sub

Public Sub GridBSave()

Dim myfile As New System.IO.StreamWriter("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & ComboBox1.SelectedItem & Label11.Text & ".txt", False)

Dim rowNum As Integer

Dim colNum As Integer

Saves Week B’s timetable for a specific room. By recording which colour is present in which cell and assigning it a letter. This is how the program knows which is free when it is loaded back into the program.

If ComboBox1.SelectedItem = "" Then

MsgBox("Please select a room")

ElseIf ComboBox1.SelectedItem <> "" Then

For rowNum = 1 To 5

For colNum = 1 To 8

GridB.Row = rowNum

GridB.Col = colNum

If GridB.CellBackColor = Color.Green Then

myfile.Write("F")

Else

myfile.Write("X")

End If

Next

Next

myfile.Close()

myfile.Dispose()

MsgBox("Timetable Updated")

End If

LGD1()

LGD2()

End Sub

Private Sub GridASave()

Dim myfile As New System.IO.StreamWriter("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & ComboBox1.SelectedItem & Label10.Text & ".txt", False)

Dim rowNum As Integer

Dim colNum As Integer

If ComboBox1.SelectedItem = "" Then

MsgBox("Please select a room")

ElseIf ComboBox1.SelectedItem <> "" Then

For rowNum = 1 To 5

Saves the timetable for Week A

For colNum = 1 To 8

GridA.Row = rowNum

GridA.Col = colNum

If GridA.CellBackColor = Color.Green Then

myfile.Write("F")

Else

myfile.Write("X")

End If

Next

Next

myfile.Close()

myfile.Dispose()

MsgBox("Timetable Updated")

End If

End Sub

Private Sub TabControl1\_MouseClick(sender As System.Object, e As System.Windows.Forms.MouseEventArgs)

PopulateComboBox(ComboBox1)

End Sub

Private Sub CheckboxYes\_MouseClick(sender As System.Object, e As System.Windows.Forms.MouseEventArgs) Handles CheckboxYes.MouseClick

If CheckboxYes.Checked = True Then

Makes it so you can’t select that there is and isn’t a printer in a room.

CheckboxNo.Checked = False

End If

End Sub

Private Sub CheckboxNo\_MouseClick(sender As System.Object, e As System.Windows.Forms.MouseEventArgs) Handles CheckboxNo.MouseClick

If CheckboxNo.Checked = True Then

Makes it so you can’t select that there is and isn’t a printer in a room.

CheckboxYes.Checked = False

End If

End Sub

Public Sub DeleteRoom()

Deleted the room and the timetables permanently.

Dim Room123 As String

If ListBox1.SelectedIndex <> -1 Then

Room123 = Microsoft.VisualBasic.Left(ListBox1.SelectedItem, InStr(ListBox1.SelectedItem, " ") - 1)

My.Computer.FileSystem.DeleteFile("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & Room123 & ".txt",

Microsoft.VisualBasic.FileIO.UIOption.AllDialogs,

Microsoft.VisualBasic.FileIO.RecycleOption.DeletePermanently,

Microsoft.VisualBasic.FileIO.UICancelOption.DoNothing)

End If

End Sub

Private Sub LGD2()

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & ComboBox1.SelectedItem & "Week B.txt")

Dim stringReader As String

stringReader = fileReader.ReadLine()

Dim pos As Integer

Dim row As Integer

Loads a timetable into the grid using the Week B table for that room.

Dim col As Integer

pos = 1

For row = 1 To 5

For col = 1 To 8

GridB.Row = row

GridB.Col = col

If Mid(stringReader, pos, 1) = "x" Then

GridB.CellBackColor = Color.Red

ElseIf Mid(stringReader, pos, 1) = "F" Then

GridB.CellBackColor = Color.Green

End If

pos = pos + 1

Next col

Next

fileReader.Close()

fileReader.Dispose()

End Sub

Private Sub LGD1()

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & ComboBox1.SelectedItem & "Week A.txt")

Dim stringReader As String

stringReader = fileReader.ReadLine()

Dim pos As Integer

Again. Loads the rooms timetable into the week A grid based on the information saved in the timetable file.

Dim row As Integer

Dim col As Integer

pos = 1

For row = 1 To 5

For col = 1 To 8

GridA.Row = row

GridA.Col = col

If Mid(stringReader, pos, 1) = "x" Then

GridA.CellBackColor = Color.Red

ElseIf Mid(stringReader, pos, 1) = "F" Then

GridA.CellBackColor = Color.Green

End If

pos = pos + 1

Next col

Next

fileReader.Close()

fileReader.Dispose()

End Sub

Private Sub ComboBox1\_SelectedIndexChanged(sender As System.Object, e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged

SetGrid()

Calls the subroutines to load timetables when selected room changes.

LGD1()

LGD2()

End Sub

Private Sub Button4\_Click(sender As System.Object, e As System.EventArgs) Handles Button4.Click

Runs the Subroutine which saves Week A’s timetable.

GridASave()

End Sub

Private Sub Button5\_Click(sender As System.Object, e As System.EventArgs) Handles Button5.Click

GridBSave()

Runs the Subroutine which saves Week B’s timetable.

End Sub

Private Sub Button8\_Click\_1(sender As System.Object, e As System.EventArgs) Handles Button8.Click

Opens the termdates page.

Term\_Dates.Show()

End Sub

End Class

Main user page:

Holds the variables for this form.

Imports System.IO

Public Class LoginScreen

Dim Term As String

Dim TermOneStart, TermOneEnd, TermTwoStart, TermTwoEnd, TermThreeStart, TermThreeEnd, TermFourStart, TermFourEnd, TermFiveStart, TermFiveEnd, TermSixStart, TermSixEnd As Date

Private Sub Button1\_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click

Me.Hide()

Another back button.

StartUpScreen.Show()

End Sub

Dim m As MsgBoxResult

Dim t As String

Private Sub MonthCalendar1\_DateSelected(sender As System.Object, e As System.Windows.Forms.DateRangeEventArgs) Handles MonthCalendar1.DateSelected

t = MonthCalendar1.SelectionRange.Start.Month.ToString & MonthCalendar1.SelectionRange.Start.Day.ToString

Try

If File.Exists(t & Label3.Text & ".txt") = True Then

MonthCalendar1.Enabled = False

MonthCalendar1.Hide()

TextBox1.Enabled = True

TextBox1.Show()

Button4.Enabled = True

Button4.Show()

Button5.Enabled = True

Button5.Show()

TextBox1.Text = File.ReadAllText(t & Label3.Text & ".txt")

Else

m = MsgBox("Would you like to enter appointments for this date?", MsgBoxStyle.YesNo)

If m = MsgBoxResult.Yes Then

MonthCalendar1.Enabled = False

MonthCalendar1.Hide()

TextBox1.Enabled = True

TextBox1.Show()

TextBox1.Text = ""

Button4.Enabled = True

Allows the user to select a date on a calendar and enter /save appointments for this date. (unique to each user)

Button4.Show()

Button5.Enabled = True

Button5.Show()

End If

End If

Catch ex As Exception

MsgBox(ex.Message)

End Try

End Sub

Private Sub Button5\_Click(sender As System.Object, e As System.EventArgs) Handles Button5.Click

TextBox1.Enabled = False

TextBox1.Hide()

Hides/Shows the relevant bits for the option they select. (as to whether they want to create an appointment.)

Button4.Enabled = False

Button4.Hide()

Button5.Enabled = False

Button5.Hide()

MonthCalendar1.Enabled = True

MonthCalendar1.Show()

End Sub

Private Sub Button4\_Click(sender As System.Object, e As System.EventArgs) Handles Button4.Click

Try

If TextBox1.Text = "" Then

If File.Exists(t & Label3.Text & ".txt") = True Then

File.Delete(t & Label3.Text & ".txt")

End If

Saves any appointments the user makes as text files.

End If

If TextBox1.Text.Length > 0 Then

File.WriteAllText(t & Label3.Text & ".txt", TextBox1.Text)

End If

Catch ex As Exception

MsgBox(ex.Message)

End Try

End Sub

Private Sub LoginScreen\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

AdminPage.PopulateComboBox(Me.ComboBox2)

DATELABELS()

Dim m1 As MsgBoxResult

t = MonthCalendar1.SelectionRange.Start.Month.ToString & MonthCalendar1.SelectionRange.Start.Day.ToString

If Date.Today = MonthCalendar1.TodayDate And File.Exists(t & Label3.Text & ".txt") = True Then

m1 = MsgBox("You have appointments set for today would you like to view them?", MsgBoxStyle.YesNo)

Sets the page up. Loads information from the rooms into combo boxes and sets the labels up depending on what date it is today.

This bit checks if the user has an appointment today and if they do it asks if they wish to view it.

If m1 = MsgBoxResult.Yes Then

MonthCalendar1.Enabled = False

MonthCalendar1.Hide()

TextBox1.Enabled = True

TextBox1.Show()

Button4.Enabled = True

Button4.Show()

Button5.Enabled = True

Button5.Show()

TextBox1.Text = File.ReadAllText(t & Label3.Text & ".txt")

End If

End If

SETGrid3Beta()

Calls all the subroutines which load data onto the page such as the bookings for that user and what term it is.

BlankTable()

TermTime()

CheckDates()

Listboxload()

WhichTerm()

End Sub

Public Sub SETGrid3Beta()

Grid3.Clear()

Dim matrixText As String() = {"R", "1", "2", "3", "L", "4", "5", "A", "Mon", "Tue", "Wed", "Thu", "Fri"}

Grid3.BackColor = Color.Red

For i As Integer = 1 To 8

Grid3.set\_TextMatrix(0, i, matrixText(i - 1))

Next

Sets the properties of the only grid on this page. This shows one week at a time and is slightly larger than the ones on the admin page.

For i As Integer = 0 To 8

Grid3.set\_ColWidth(i, 1163)

Next

For i As Integer = 1 To 5

Grid3.set\_TextMatrix(i, 0, matrixText(i + 7))

Next

End Sub

Public Sub Listboxload()

bookedlist.Items.Clear()

Dim recnum As Integer = 0

Dim stringReader As String

Dim BookingVals(0 To 4) As String

Dim Row As Integer

Dim col As Integer

Dim Name As String

Dim RoomNO As Integer

Dim date123 As Date

Dim matrixText As String() = {Name}

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt")

Do While Not fileReader.EndOfStream

recnum += 1

stringReader = fileReader.ReadLine

BookingVals = stringReader.Split(",")

Row = Val(BookingVals(0))

Reads the bookings and changes the format from two numbers to a date and a period. This is then loaded into a listbox.

col = Val(BookingVals(1))

Name = BookingVals(2)

RoomNO = Val(BookingVals(3))

date123 = (BookingVals(4))

Dim period As String = 0

If col = 1 Then period = "Morning "

If col = 2 Then period = "First "

If col = 3 Then period = "Second "

If col = 4 Then period = "third "

If col = 5 Then period = "Lunch "

If col = 6 Then period = "Fourth "

If col = 7 Then period = "Fith "

If col = 8 Then period = "AfterSchool"

Dim Day As String

If Row = 1 Then Day = "Monday "

If Row = 2 Then Day = "Tuesday "

If Row = 3 Then Day = "Wednesday"

If Row = 4 Then Day = "Thursday "

If Row = 5 Then Day = "Friday "

If Name = Label3.Text Then

Me.bookedlist.Items.Add(Name & " " & RoomNO & " " & period & " " & Day & " " & date123 & " " & Format(recnum, "000"))

End If

Each booking is assigned a number this makes it easier to delete the booking as through program searches through via the numbers.

Loop

fileReader.Close()

fileReader.Dispose()

End Sub

Private Sub ComboBox1\_SelectedIndexChanged\_1(sender As System.Object, e As System.EventArgs) Handles ComboBox2.SelectedIndexChanged

Calls the subroutine which loads the right week into the grid. Depending on what week the user is looking for.

If Label10.Text = "Week A" Then LGDA()

If Label10.Text = "Week B" Then LGDB()

BookedRooms()

Calls the subroutine which checks for active bookings.

BlankTable()

End Sub

Private Sub LGDA()

Grid3.Clear()

SETGrid3Beta()

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & ComboBox2.SelectedItem & "Week A.txt")

Dim stringReader As String

stringReader = fileReader.ReadLine()

Dim pos As Integer

Dim row As Integer

Loads week a’s timetable using the combo box to select the room’s timetable to load.

Dim col As Integer

pos = 1

For row = 1 To 5

For col = 1 To 8

Grid3.Row = row

Grid3.Col = col

If Mid(stringReader, pos, 1) = "x" Then

Grid3.CellBackColor = Color.Red

ElseIf Mid(stringReader, pos, 1) = "F" Then

Grid3.CellBackColor = Color.Green

End If

pos = pos + 1

Next col

Next

fileReader.Close()

fileReader.Dispose()

End Sub

Public Sub BookedRooms()

Dim TestDate As Date

Checks for active bookings by using the booking file.

Dim stringReader As String

Dim BookingVals(0 To 4) As String

Dim Row As Integer

Dim col As Integer

Dim Name As String

Dim RoomNO As Integer

Dim date123 As Date

Dim matrixText As String() = {Name}

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt")

Do While Not fileReader.EndOfStream

stringReader = fileReader.ReadLine

It selects which bookings to display based on the date in the file and the date on the program. It also only displays the bookings for the selected room. However it does show all users bookings for that room and in that date range so you can see the availability.

BookingVals = stringReader.Split(",")

Row = Val(BookingVals(0))

col = Val(BookingVals(1))

Name = BookingVals(2)

RoomNO = Val(BookingVals(3))

date123 = (BookingVals(4))

TestDate = Label1.Text

For Row = 1 To 5

If RoomNO = Val(ComboBox2.SelectedItem) And TestDate = date123 Then

Me.Grid3.Row = Row

Me.Grid3.Col = col

Me.Grid3.CellBackColor = Color.Gray

Me.Grid3.Text = Name

This greys out any days that have passed in the week. So if we are on Wednesday, it would grey out Monday and Tuesday so the user couldn’t book a room.

End If

TestDate = TestDate.AddDays(1)

Next

Loop

fileReader.Close()

fileReader.Dispose()

End Sub

Private Sub LGDB()

Grid3.Clear()

SETGrid3Beta()

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Timetable" & ComboBox2.SelectedItem & "Week B.txt")

Dim stringReader As String

stringReader = fileReader.ReadLine()

Dim pos As Integer

Loads week B’s timetable for a specific room when the user requires the week B timetable.

Dim row As Integer

Dim col As Integer

pos = 1

For row = 1 To 5

For col = 1 To 8

Grid3.Row = row

Grid3.Col = col

If Mid(stringReader, pos, 1) = "x" Then

Grid3.CellBackColor = Color.Red

ElseIf Mid(stringReader, pos, 1) = "F" Then

Grid3.CellBackColor = Color.Green

End If

pos = pos + 1

Next col

Next

fileReader.Close()

fileReader.Dispose()

End Sub

Private Sub DATELABELS()

Puts values in each label. Depending on the date. It takes todays date then changes the dates in the other labels so a week Monday to Friday is always displayed correctly.

Dim date1 As Date = Now

Select Case Now.Date.DayOfWeek

Case DayOfWeek.Monday

Label1.Text = Format(date1.Date, "dd/MM/yy")

Label2.Text = Format(date1.AddDays(1).Date, "dd/MM/yy")

Label13.Text = Format(date1.AddDays(2).Date, "dd/MM/yy")

Label14.Text = Format(date1.AddDays(3).Date, "dd/MM/yy")

Label5.Text = Format(date1.AddDays(4).Date, "dd/MM/yy")

Case DayOfWeek.Tuesday

Label2.Text = Format(date1.Date, "dd/MM/yy")

Label13.Text = Format(date1.AddDays(1).Date, "dd/MM/yy")

Label14.Text = Format(date1.AddDays(2).Date, "dd/MM/yy")

Label5.Text = Format(date1.AddDays(3).Date, "dd/MM/yy")

Label1.Text = Format(date1.AddDays(-1).Date, "dd/MM/yy")

Case DayOfWeek.Wednesday

Label13.Text = Format(date1.Date, "dd/MM/yy")

Label1.Text = Format(date1.AddDays(-2).Date, "dd/MM/yy")

Label2.Text = Format(date1.AddDays(-1).Date, "dd/MM/yy")

Label14.Text = Format(date1.AddDays(1).Date, "dd/MM/YY")

Label5.Text = Format(date1.AddDays(2).Date, "dd/MM/yy")

Case DayOfWeek.Thursday

Label14.Text = Format(date1.Date, "dd/MM/yy")

Label1.Text = Format(date1.AddDays(-3).Date, "dd/MM/yy")

Label2.Text = Format(date1.AddDays(-2).Date, "dd/MM/yy")

Label13.Text = Format(date1.AddDays(-1).Date, "dd/MM/yy")

Label5.Text = Format(date1.AddDays(1).Date, "dd/MM/yy")

Case DayOfWeek.Friday

Label5.Text = Format(date1.Date, "dd/MM/yy")

Label13.Text = Format(date1.AddDays(-2).Date, "dd/MM/yy")

Label1.Text = Format(date1.AddDays(-4).Date, "dd/MM/yy")

Label2.Text = Format(date1.AddDays(-3).Date, "dd/MM/yy")

Label14.Text = Format(date1.AddDays(-1).Date, "dd/MM/yy")

End Select

End Sub

Private Sub Button2\_Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click

Dim WEEK As String

Dim THEDateM As Date = Label1.Text

Dim THEDateTu As Date = Label2.Text

Dim THEDateW As Date = Label13.Text

Dim THEDateTh As Date = Label14.Text

Dim THEDateF As Date = Label5.Text

WEEK = Label10.Text

If ComboBox2.SelectedItem = "" Then

MsgBox("please Select a room to view the time table")

elseIf Label10.Text = "Week B" Then

LGDB()

THEDateM = THEDateM.AddDays(7)

Adds 7 days when the week is cycled forwards. It also loads the next weeks timetable for that room.

THEDateTu = THEDateTu.AddDays(7)

THEDateW = THEDateW.AddDays(7)

THEDateTh = THEDateTh.AddDays(7)

THEDateF = THEDateF.AddDays(7)

Label1.Text = Format(THEDateM, "dd/MM/yy")

Label2.Text = Format(THEDateTu, "dd/MM/yy")

Label13.Text = Format(THEDateW, "dd/MM/yy")

Label14.Text = Format(THEDateTh, "dd/MM/yy")

Label5.Text = Format(THEDateF, "dd/MM/yy")

ElseIf Label10.Text = "Week A" Then

LGDA()

THEDateM = THEDateM.AddDays(7)

THEDateTu = THEDateTu.AddDays(7)

THEDateW = THEDateW.AddDays(7)

THEDateTh = THEDateTh.AddDays(7)

THEDateF = THEDateF.AddDays(7)

Label1.Text = Format(THEDateM, "dd/MM/yy")

Label2.Text = Format(THEDateTu, "dd/MM/yy")

Label13.Text = Format(THEDateW, "dd/MM/yy")

Label14.Text = Format(THEDateTh, "dd/MM/yy")

Label5.Text = Format(THEDateF, "dd/MM/yy")

BlankTable()

End If

Calls the subroutines to check for live bookings, to check that it is in an active term and to determine which term is live.

BookedRooms()

TermTime()

WhichTerm()

End Sub

Private Sub Grid3\_ClickEvent(sender As System.Object, e As System.EventArgs) Handles Grid3.ClickEvent, Grid3.Enter

Dim M As MsgBoxResult

The main booking Sub routine. This allows the user to click on the green cells and reserve a booking. It is done by asking if they are sure then if they select yes, they have booked the room.

Dim Date1 As String

Dim N As MsgBoxResult

If Grid3.Row = 1 Then

Date1 = Label1.Text

End If

If Grid3.Row = 2 Then

Date1 = Label2.Text

End If

If Grid3.Row = 3 Then Date1 = Label13.Text

If Grid3.Row = 4 Then Date1 = Label14.Text

If Grid3.Row = 5 Then Date1 = Label5.Text

Dim x As System.Drawing.Color = Grid3.CellBackColor

If x = Color.Green Then

M = MsgBox("Would you like to book this room?", MsgBoxStyle.YesNo)

If M = MsgBoxResult.Yes Then

N = MsgBox("Room booked.", MsgBoxStyle.OkOnly)

If N = MsgBoxResult.Ok Then

'Listboxload()

End If

Dim myfile As New System.IO.StreamWriter("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt", True)

If Grid3.Col = 1 Then

End If

If ComboBox2.SelectedItem = "" Then

MsgBox("Please select a room")

ElseIf ComboBox2.SelectedItem <> "" Then

periods()

myfile.Write(Grid3.Row & "," & Grid3.Col & "," & Label3.Text & "," & ComboBox2.SelectedItem & "," & Date1)

myfile.WriteLine()

It saves the row, column, the week, the room and the date.

myfile.Close()

myfile.Dispose()

End If

End If

ElseIf x <> Color.Gray Then

MsgBox("This room cannot be booked.")

Else

Greys the square and adds the users name to the square. To show it is booked and who by.

End If

BookedRooms()

End Sub

Private Sub TabControl1\_SelectedIndexChanged(sender As System.Object, e As System.EventArgs) Handles TabControl1.SelectedIndexChanged

If TabControl1.SelectedTab.Text = "Active Bookings" Then

Listboxload()

When the tab is changed run the subroutines to check the dates are ok and load the bookings into a listbox.

CheckDates()

End If

End Sub

Private Sub Button3\_Click(sender As System.Object, e As System.EventArgs) Handles Button3.Click

Refreshes the live bookings table. “Refresh” button.

Listboxload()

End Sub

Public Sub CheckDates()

Dim Today As Date

Today = Date.Now

Dim date123 As Date

Dim Newfile As New System.IO.StreamWriter("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\NewBookings.txt", True)

Dim stringReader As String

Dim BookingVals(0 To 4) As String

Checks the current date against live bookings. If the date has past it removes the bookings.

Dim Row As Integer

Dim col As Integer

Dim Name As String

Dim RoomNO As Integer

Dim matrixText As String() = {Name}

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt")

Do While Not fileReader.EndOfStream

stringReader = fileReader.ReadLine

BookingVals = stringReader.Split(",")

Row = Val(BookingVals(0))

col = Val(BookingVals(1))

Name = BookingVals(2)

RoomNO = Val(BookingVals(3))

date123 = (BookingVals(4))

If date123 >= Today.Date Then

Newfile.WriteLine(Row & "," & col & "," & Name & "," & RoomNO & "," & date123)

End If

Loop

Newfile.Close()

Newfile.Dispose()

fileReader.Close()

fileReader.Dispose()

Kill("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt")

My.Computer.FileSystem.RenameFile("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\NewBookings.txt", "Bookings.txt") 'this sub checks the current date and will stop loading bookigns from dates that have passed

End Sub

Public Sub BlankTable()

Checks the date today and blanks out any dates that have passed so the user can’t book a room for a date that has already passed. It physically greys out the cells to they cannot be selected.

Dim TestDate As Date

TestDate = Label1.Text

For row = 1 To 5

If TestDate < Today.Date Then

Grid3.Row = row

For i = 1 To 8

Grid3.Col = i

Grid3.CellBackColor = Color.Gray

Next

End If

TestDate = TestDate.AddDays(1)

Next

End Sub

Public Sub TermTime()

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\termdates.txt")

Dim stringReader As String

Dim Dates(0 To 11) As String

Do While Not fileReader.EndOfStream

stringReader = fileReader.ReadLine

Dates = stringReader.Split(".")

Sees and displays the current term the user is in. It determines what term the user is currently in.

TermOneStart = Dates(0)

TermOneEnd = Dates(1)

TermTwoStart = Dates(2)

TermTwoEnd = Dates(3)

TermThreeStart = Dates(4)

TermThreeEnd = Dates(5)

TermFourStart = Dates(6)

TermFourEnd = Dates(7)

TermFiveStart = Dates(8)

TermFiveEnd = Dates(9)

TermSixStart = Dates(10)

TermSixEnd = Dates(11)

If TermOneStart <= Today.Date And Today.Date <= TermOneEnd Then Term = "1"

If TermTwoStart < Today.Date And Today.Date > TermTwoEnd Then Term = "2"

If TermThreeStart < Today.Date And Today.Date > TermThreeEnd Then Term = "3"

If TermFourStart < Today.Date And Today.Date > TermFourEnd Then Term = "4"

If TermFiveStart < Today.Date And Today.Date > TermFiveEnd Then Term = "5"

If TermSixStart < Today.Date And Today.Date > TermSixEnd Then Term = "6"

Loop

Dim testdate As Date = Label1.Text

fileReader.Close()

fileReader.Dispose()

getWEEK(testdate)

End Sub

Public Sub WhichTerm()

Dim currentterm As String = ""

Dim Check(0 To 4)

Dim day As Date

Dim IsTerm As Boolean

Updates the term label depending on which date the user is looking at in the table. Does this by referencing a date and sees if it is within a certain date range.

Dim placeholder(4) As Label

placeholder(0) = Label1

placeholder(1) = Label2

placeholder(2) = Label13

placeholder(3) = Label14

placeholder(4) = Label5

For i As Integer = 0 To placeholder.GetUpperBound(0)

If placeholder(i).Text > TermOneStart And placeholder(i).Text < TermOneEnd Then

currentterm = "1"

Exit For

End If

If placeholder(i).Text > TermTwoStart And placeholder(i).Text < TermTwoEnd Then

currentterm = "2"

Exit For

End If

If placeholder(i).Text > TermThreeStart And placeholder(i).Text < TermThreeEnd Then

currentterm = "3"

Exit For

End If

If placeholder(i).Text > TermFourStart And placeholder(i).Text < TermFourEnd Then

currentterm = "4"

Exit For

End If

If placeholder(i).Text > TermFiveStart And placeholder(i).Text < TermFiveEnd Then

currentterm = "5"

Exit For

End If

If placeholder(i).Text > TermSixStart And placeholder(i).Text < TermSixEnd Then

currentterm = "6"

Exit For

End If

Next

Label7.Text = currentterm

End Sub

Function getWEEK(date2check As Date) As String

Dim week As String = "A"

Dim DayOfWeek As Integer = TermOneStart.DayOfWeek

Dim startdate As Date = TermOneStart.AddDays(-(DayOfWeek - 1))

Do

If date2check >= startdate And date2check <= startdate.AddDays(+6) Then

If week = "B" Then

Label10.Text = "Week B"

Sees if the week is a Week A or B depending on the term times entered at the start of the year.

Else

Label10.Text = "Week A"

End If

Return week

Else

startdate = startdate.AddDays(7)

If week = "A" Then week = "B" Else week = "A"

If startdate > TermOneEnd Then

Exit Do

End If

Each loop determines what week each new term starts on depending on what the last one ended on.

End If

Loop

DayOfWeek = TermTwoStart.DayOfWeek

startdate = TermTwoStart.AddDays(-(DayOfWeek - 1))

Do

If date2check >= startdate And date2check <= startdate.AddDays(+6) Then

If week = "B" Then

Label10.Text = "Week B"

Else

Label10.Text = "Week A"

End If

Return week

Else

startdate = startdate.AddDays(+7)

If week = "A" Then week = "B" Else week = "A"

If startdate > TermTwoEnd Then

Exit Do

End If

End If

Checks start week for term 3.

Loop

DayOfWeek = TermThreeStart.DayOfWeek

startdate = TermThreeStart.AddDays(-(DayOfWeek - 1))

Do

If date2check >= startdate And date2check <= startdate.AddDays(+6) Then

If week = "B" Then

Label10.Text = "Week B"

Else

Label10.Text = "Week A"

End If

Return week

Else

startdate = startdate.AddDays(+7)

If week = "A" Then week = "B" Else week = "A"

If startdate > TermThreeEnd Then

Exit Do

End If

End If

Checks start week for term 4.

Loop

DayOfWeek = TermFourStart.DayOfWeek

startdate = TermFourStart.AddDays(-(DayOfWeek - 1))

Do

If date2check >= startdate And date2check <= startdate.AddDays(+6) Then

If week = "B" Then

Label10.Text = "Week B"

Else

Label10.Text = "Week A"

End If

Return week

Else

startdate = startdate.AddDays(+7)

If week = "A" Then week = "B" Else week = "A"

If startdate > TermFourEnd Then

Exit Do

End If

End If

Checks start week for term 5.

Loop

DayOfWeek = TermFiveStart.DayOfWeek

startdate = TermFiveStart.AddDays(-(DayOfWeek - 1))

Do

If date2check >= startdate And date2check <= startdate.AddDays(+6) Then

If week = "B" Then

Label10.Text = "Week B"

Else

Label10.Text = "Week A"

End If

Return week

Else

startdate = startdate.AddDays(+7)

If week = "A" Then week = "B" Else week = "A"

If startdate > TermFiveEnd Then

Exit Do

End If

Checks start week for term 6.

End If

Loop

DayOfWeek = TermSixStart.DayOfWeek

startdate = TermSixStart.AddDays(-(DayOfWeek - 1))

Do

If date2check >= startdate And date2check <= startdate.AddDays(+6) Then

If week = "B" Then

Label10.Text = "Week B"

Else

Label10.Text = "Week A"

End If

Return week

Else

startdate = startdate.AddDays(+7)

If week = "A" Then week = "B" Else week = "A"

If startdate > TermSixEnd Then

Exit Do

End If

End If

Loop

If week = "B" Then

Label10.Text = "Week B"

End If

If week = "A" Then

Label10.Text = "Week A"

End If

Because it’s a function it simply returns A or B

End Function

Private Sub Button8\_Click\_1(sender As System.Object, e As System.EventArgs) Handles Button8.Click

Dim WEEK As String

Dim THEDateM As Date = Label1.Text

Dim THEDateTu As Date = Label2.Text

Moves the week back one. So rather than adding 7 days it takes 7 days.

Dim THEDateW As Date = Label13.Text

Dim THEDateTh As Date = Label14.Text

Dim THEDateF As Date = Label5.Text

WEEK = Label10.Text

If ComboBox2.SelectedItem = "" Then

MsgBox("please Select a room to view the time table")

ElseIf WEEK = "Week B" Then

LGDB()

THEDateM = THEDateM.AddDays(-7)

THEDateTu = THEDateTu.AddDays(-7)

THEDateW = THEDateW.AddDays(-7)

THEDateTh = THEDateTh.AddDays(-7)

THEDateF = THEDateF.AddDays(-7)

Label1.Text = Format(THEDateM, "dd/MM/yy")

Label2.Text = Format(THEDateTu, "dd/MM/yy")

Label13.Text = Format(THEDateW, "dd/MM/yy")

Label14.Text = Format(THEDateTh, "dd/MM/yy")

Label5.Text = Format(THEDateF, "dd/MM/yy")

BlankTable()

ElseIf WEEK = "Week A" Then

Also checks which week to load. It changes the label from week a to week b and loads the correct timetable and cross references the bookings and whether the date has passed.

LGDA()

THEDateM = THEDateM.AddDays(-7)

THEDateTu = THEDateTu.AddDays(-7)

THEDateW = THEDateW.AddDays(-7)

THEDateTh = THEDateTh.AddDays(-7)

THEDateF = THEDateF.AddDays(-7)

Label1.Text = Format(THEDateM, "dd/MM/yy")

Label2.Text = Format(THEDateTu, "dd/MM/yy")

Label13.Text = Format(THEDateW, "dd/MM/yy")

Label14.Text = Format(THEDateTh, "dd/MM/yy")

Label5.Text = Format(THEDateF, "dd/MM/yy")

BlankTable()

End If

BookedRooms()

TermTime()

WhichTerm()

End Sub

Private Sub Button6\_Click(sender As System.Object, e As System.EventArgs) Handles Button6.Click

Dim Y As MsgBoxResult

Y = MsgBox("Are you sure you want to remove this booking?", MsgBoxStyle.YesNo)

If Y = MsgBoxResult.Yes Then

Dim period As String

Dim date123 As Date

Dim Newfile As New System.IO.StreamWriter("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\NewBookings.txt", True)

Dim stringReader As String

Dim BookingVals(0 To 4) As String

Dim Row As Integer

Dim col As Integer

Removes bookings selected by as user.

Dim Name As String

Dim RoomNO As Integer

Dim ok As Boolean

Dim recnum As Integer = 0

Dim matrixText As String() = {Name}

Dim fileReader As System.IO.StreamReader

fileReader =

My.Computer.FileSystem.OpenTextFileReader("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt")

Do While Not fileReader.EndOfStream

stringReader = fileReader.ReadLine

BookingVals = stringReader.Split(",")

Row = Val(BookingVals(0))

Dim Day As String

If Row = 1 Then Day = "Monday"

If Row = 2 Then Day = "Tuesday"

If Row = 3 Then Day = "Wednesday"

If Row = 4 Then Day = "Thursday"

If Row = 5 Then Day = "Friday"

col = Val(BookingVals(1))

If col = 1 Then period = "Morning"

If col = 2 Then period = "First"

If col = 3 Then period = "Second"

If col = 4 Then period = "third"

If col = 5 Then period = "Lunch"

If col = 6 Then period = "Fourth"

If col = 7 Then period = "Fith"

If col = 8 Then period = "AfterSchool"

Name = BookingVals(2)

RoomNO = Val(BookingVals(3))

date123 = (BookingVals(4))

ok = False

recnum += 1

If recnum <> Val(Microsoft.VisualBasic.Right(bookedlist.SelectedItem, 2)) Then

Newfile.WriteLine(Row & "," & col & "," & Name & "," & RoomNO & "," & date123)

End If

Loop

Deletes the bookings by copying the file without and blank records. Killing the old file, then renaming the copy.

Newfile.Close()

Newfile.Dispose()

fileReader.Close()

fileReader.Dispose()

Kill("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Bookings.txt")

My.Computer.FileSystem.RenameFile("M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\NewBookings.txt", "Bookings.txt")

bookedlist.Items.Remove(bookedlist.SelectedItem)

SETGrid3Beta()

BlankTable()

ComboBox2.SelectedItem = ""

MsgBox("Booking deleted")

End If

End Sub

End Class

**Login Page:**

Public Class StartUpScreen

Private Sub Button2\_Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click

If TxtUser.Text = "Admin" And TxtPass.Text = "password123" Then

Button3.Visible = True

If the admin login is typed in, display a secret button to get into the admin page.

End If

Dim logins As New login

Dim reclen As Integer = Len(logins)

Dim numrecords As Integer

Dim found As Boolean

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\logins.dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, reclen)

numrecords = LOF(1) / reclen

found = False

For recordcounter = 1 To numrecords

FileGet(1, logins, recordcounter)

If Trim(logins.username) = TxtUser.Text And Trim(logins.password) = TxtPass.Text Then

Me.Hide()

Checks the login is valid.

LoginScreen.Show()

found = True

Exit For

End If

Next

FileClose(1)

If Not found Then MsgBox("incorrect login. Try Again")

LoginScreen.Label3.Text = TxtUser.Text

TxtUser.Text = ""

Incorrect login results in message box and clears the fields of data.

TxtPass.Text = ""

End Sub

Private Sub StartUpScreen\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

TxtUser.Text = ""

Clears the textboxes when the page loads.

TxtPass.Text = ""

End Sub

Private Sub Button3\_Click(sender As System.Object, e As System.EventArgs) Handles Button3.Click

Secret button that shows admin page.

Me.Hide()

AdminPage.Show()

End Sub

Private Sub Label3\_Click\_1(sender As System.Object, e As System.EventArgs) Handles Label3.Click

Me.Hide()

“Forgot password” button.

FrmPassword.Show()

End Sub

End Class

**FORGOT PASSWORD? PAGE**

Public Class FrmPassword

Declares variables for that page.

Dim currentrecord As Integer

Dim reclen As Integer = Len(login)

Dim Login As login

Private Sub Button1\_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click

if the correct information is entered, show the users password so they can log in.

Dim response As Integer

If TextBox4.Text = "password123" Then

response = MsgBox("Show User information. Warning this information will be in plain text. ", MsgBoxStyle.Question + MsgBoxStyle.YesNo, "Delete Record")

Else

MsgBox("Please Enter the Admin Password(Call Datcom 0845 521 2405)")

End If

If response = vbYes Then

Calls the subroutine called findUser.

Call FindUser()

End If

End Sub

Public Sub FindUser()

Dim Login As New logi

Dim numrecords As Integer

Dim recordcounter As Integer

Dim found As Boolean

FileOpen(1, "M:\A Levels\Computing\COmputing CCat test\Project prototype\prototype\Logins.Dat", OpenMode.Random, OpenAccess.ReadWrite, OpenShare.Shared, reclen)

currentrecord = 0

numrecords = LOF(1) / reclen

Uses the username to find the user and their password in the login file.

found = False

For recordcounter = 1 To numrecords

FileGet(1, Login, recordcounter)

If Trim(Login.username) = TextBox1.Text Then

TextBox2.Text = Trim(Login.password)

currentrecord = recordcounter

found = True

Exit For

End If

Next

FileClose(1)

If Not found Then MsgBox("User Not Found. Try Again")

End Sub

Private Sub Button2\_Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click

TextBox1.Text = ""

TextBox2.Text = ""

“Back” Button

TextBox4.Text = ""

Me.Hide()

StartUpScreen.Show()

End Sub

End Class

# 4. Testing

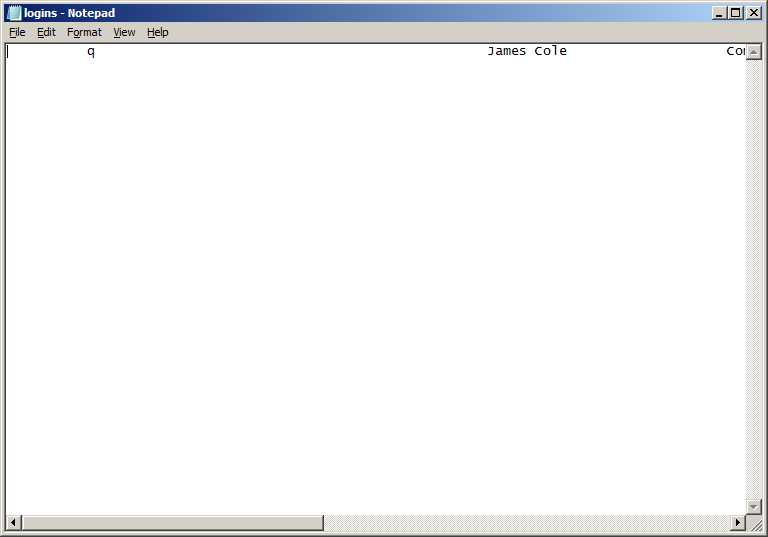
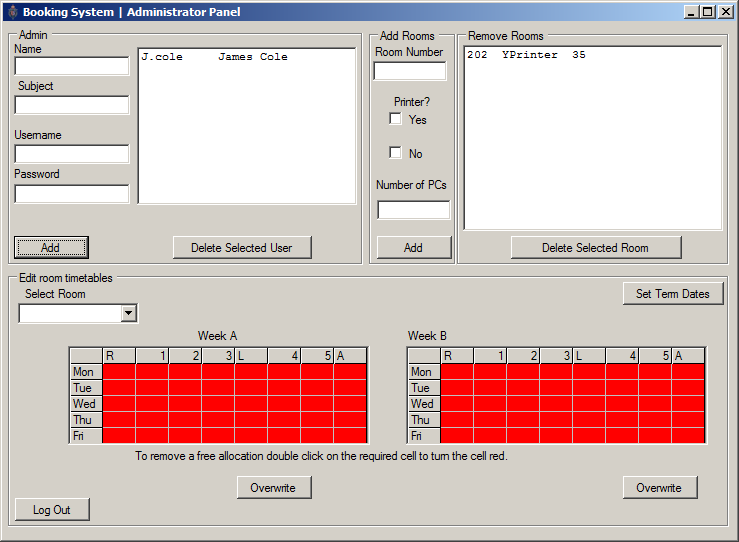
## 4.1 Introduction

There are a few main types of testing. Top-down testing. This is when the whole system is tested with individual functions being in place but doing nothing other than displaying a message or writing something to a file to show the overall logic is correct. Bottom-up which is when Each main function written is fully tested as soon as complete. This is in contrast to top-down testing where many functions are not completed. White box testing is when each function is tested by slowly stepping through the code line by line. This means if there is an error it is easier to find the problem because you can isolate the bug to one line. In contrast, Black box testing is when each main function is called and then checked after the process is finished. This is harder to find a bug because you have to search through the code and then isolate the problem without knowing where it is.

In my testing I will mostly use black box testing as it is easy to see what my program does and each function is relatively small and easy to fix.

## 4.2 Normal Data

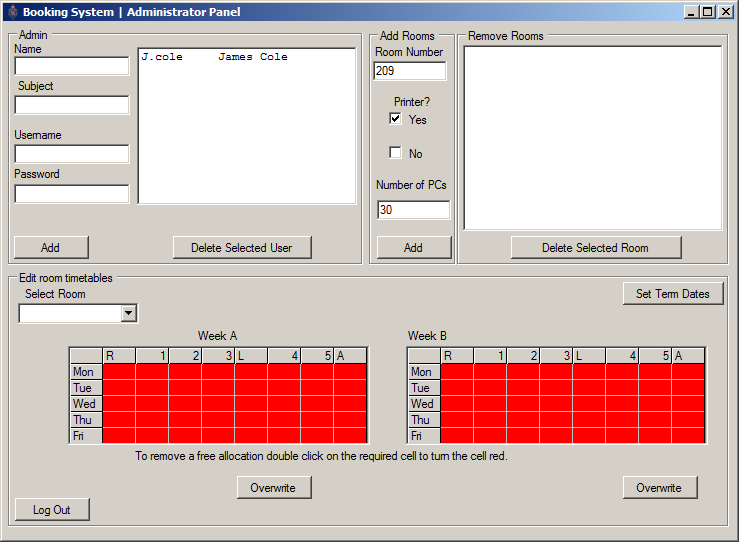
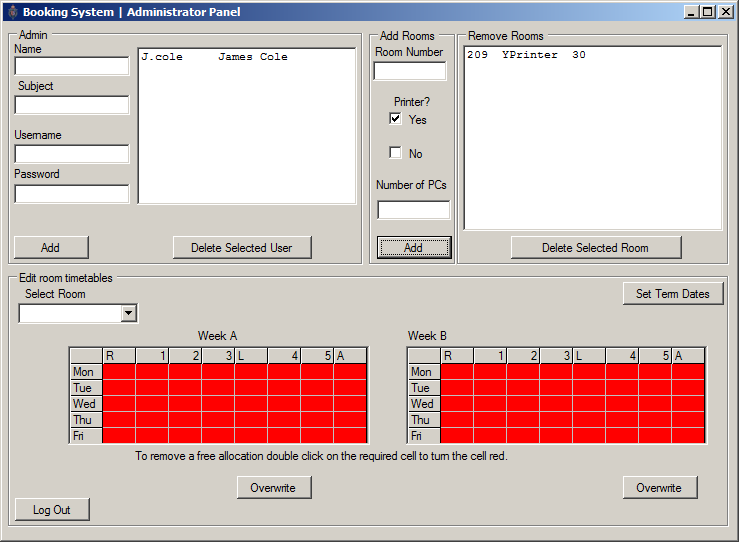
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test No.** | **Description / Purpose of test** | **Input Data** | **Test Method** | **Expected Results** | **Actual Results** |
| **1** | Add a new user | James Cole, computing , J.cole, Q | Black box | The user would be added to the end of the user file and it will be displayed in the listbox. | See screenshot 1 and 2 below. |
| **2** | Add a new room | 209, Yprinter , 30 | Black box | The room will be added to the end of the rooms file and it will be displayed in the listbox. | See screenshot 3 and screenshot 3.1 below |
| **3** | Save a booking | 1,1,J.cole,209,19/03/2018 | Black box | The table will display a grey box with the username in. it will also save the booking to the booking file and display it in the listbox. | See screenshot 4 and 5 below |
| **4** | Remove a room |  | Black box | The room will be removed from the lsitbox and it will not appear in the combobox for room seletion. Furthermore, any timetables with this room will be removed. | See screenshot 6 and 7 below |
| **5** | Insert a rooms timetable | FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX | Black box | The timetable will be saved to a text file with “F” s representing a free period and “X”s representing unavailability. | See screenshot 8 and 9 below |
| **6** | Cancel a booking | 1,1,J.cole,209,19/03/2018 | Black box | The booking is removed from the file and the block appears free again. | See screenshot 10, 11 and 12 |
| **7** | Booking more than one Room | 1,1,J.cole,209,19/03/2018  1,2,J.cole,123,12/03/18 | Black box | Both bookings appear on the graphic on the bottom of the page and the list at the top. | See screenshot 13 and 14 |
| **8** | Storing multiple Rooms | 209, Yprinter , 30  123,Nprinter,60 | Black box | Both rooms will appear on the program and in the rooms file. | See screenshot 15 and 16 |



Screenshot 1

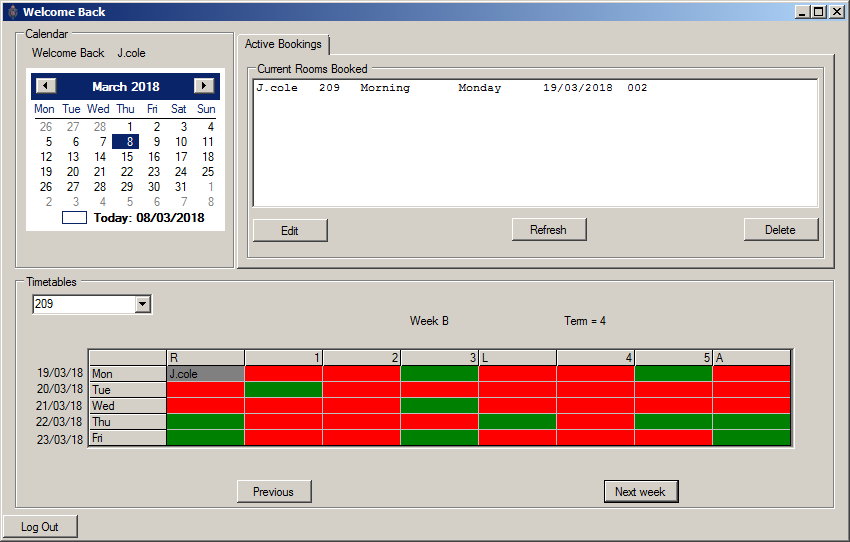
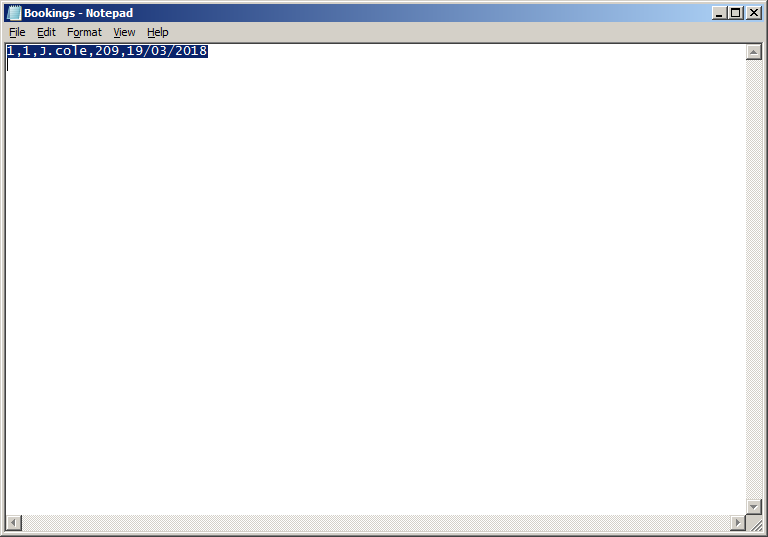
Screenshot 1

Screenshot 2



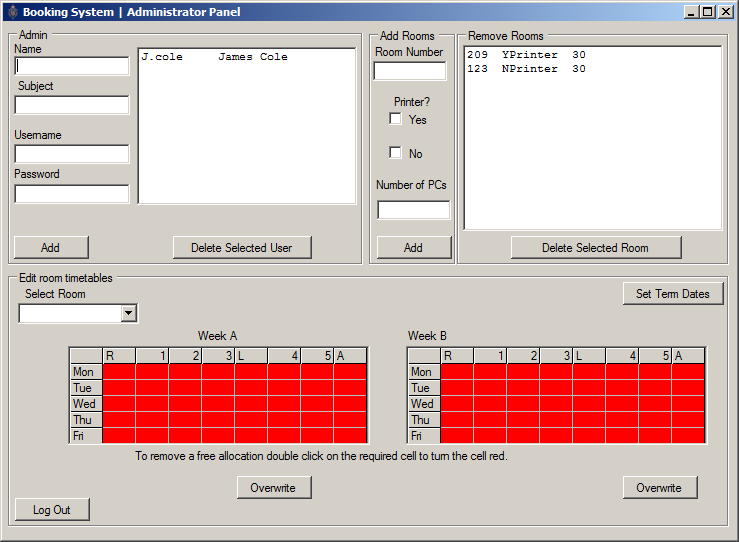
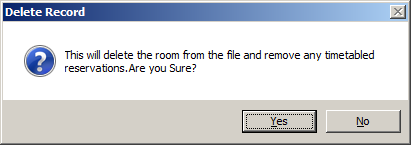
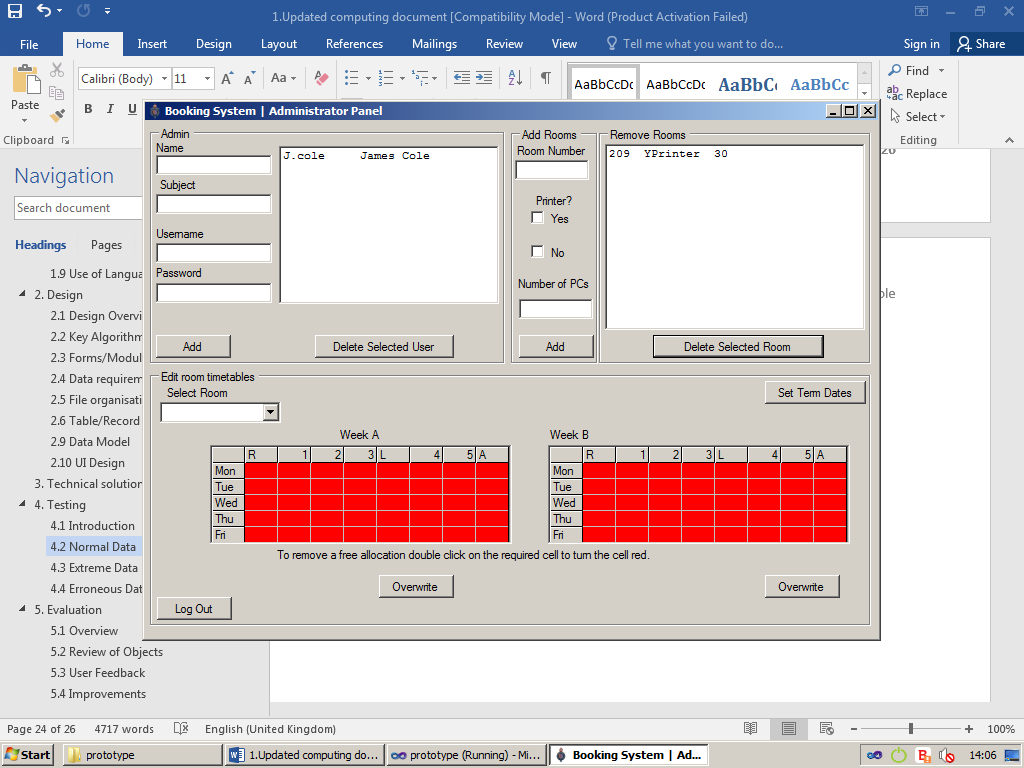
Screenshot 3.1

Screenshot 3

****

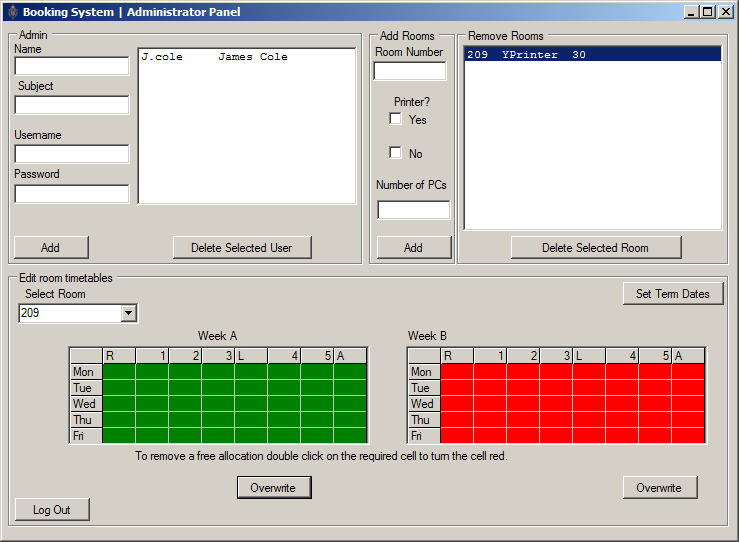
Screenshot 5

Screenshot 4

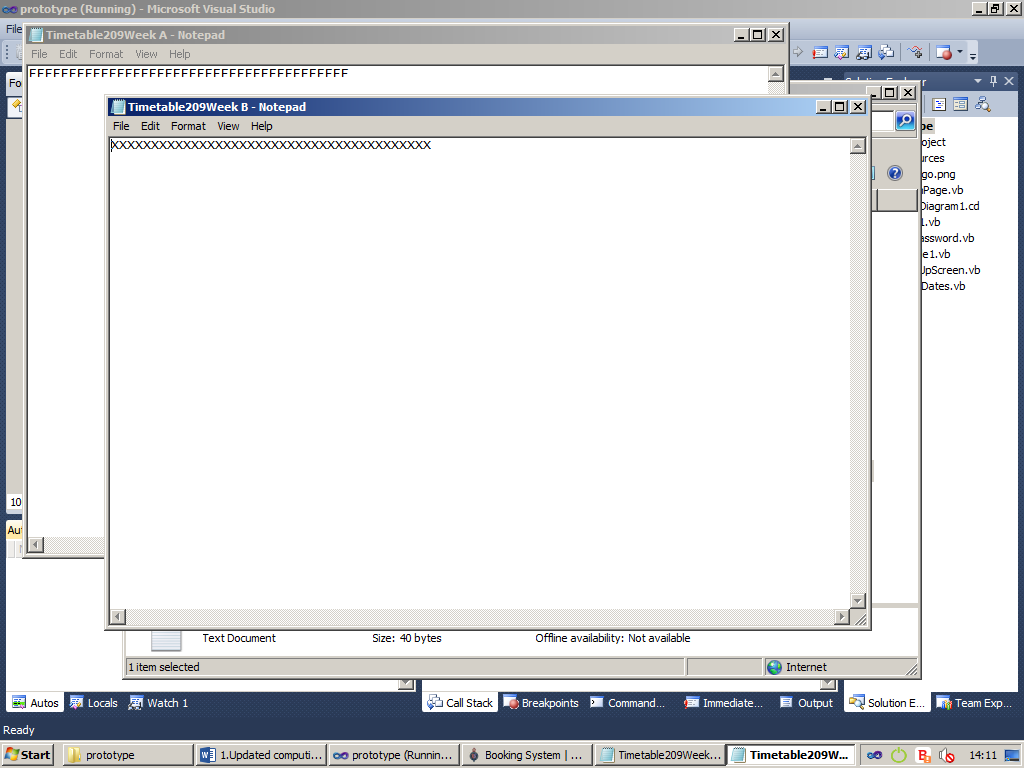


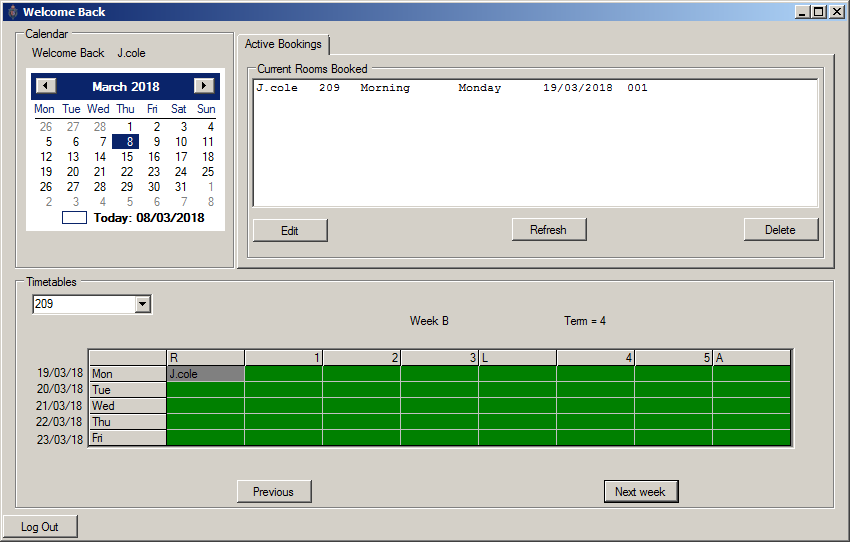
Screenshot 6

Screenshot 7



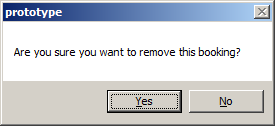
Screenshot 8

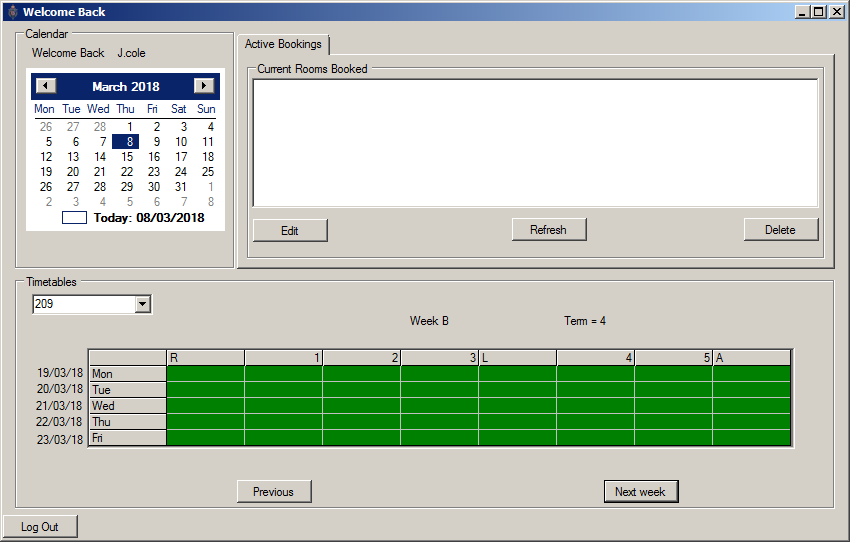




Screenshot 9

Screenshot 10

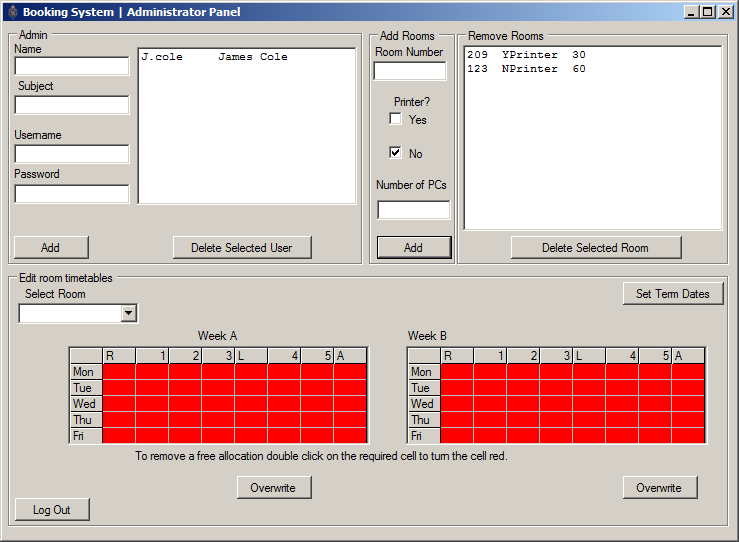
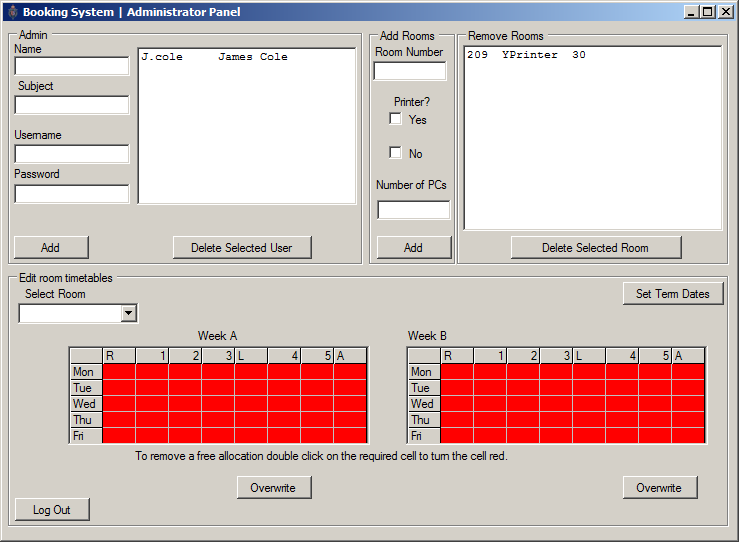




Screenshot 11

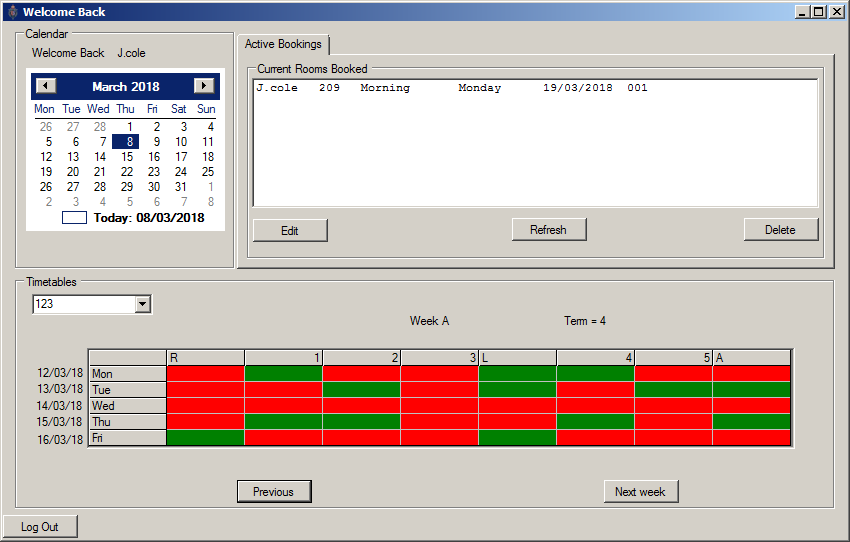
Screenshot 12

Screenshot 12

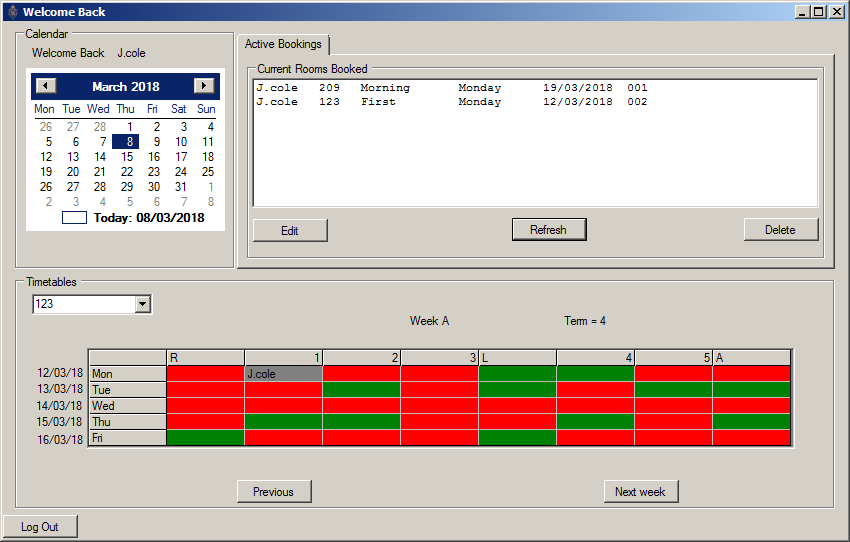


Screenshot 14

Screenshot 13



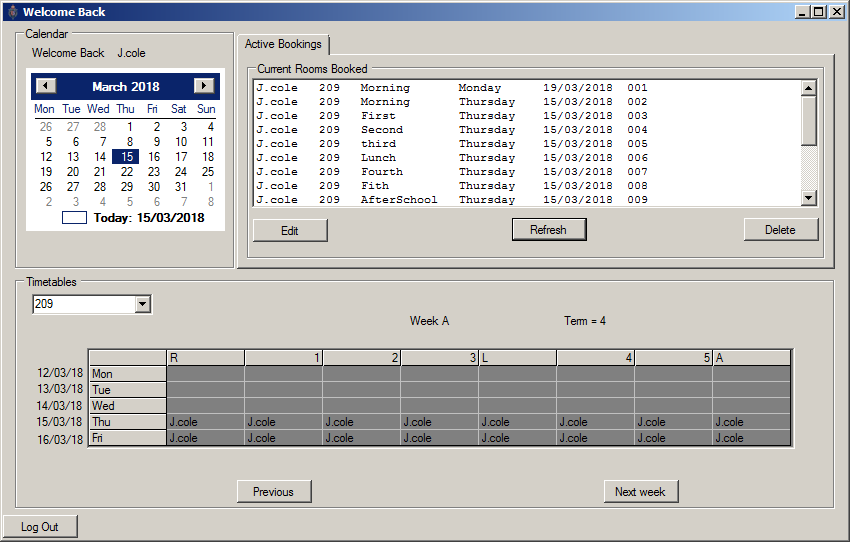
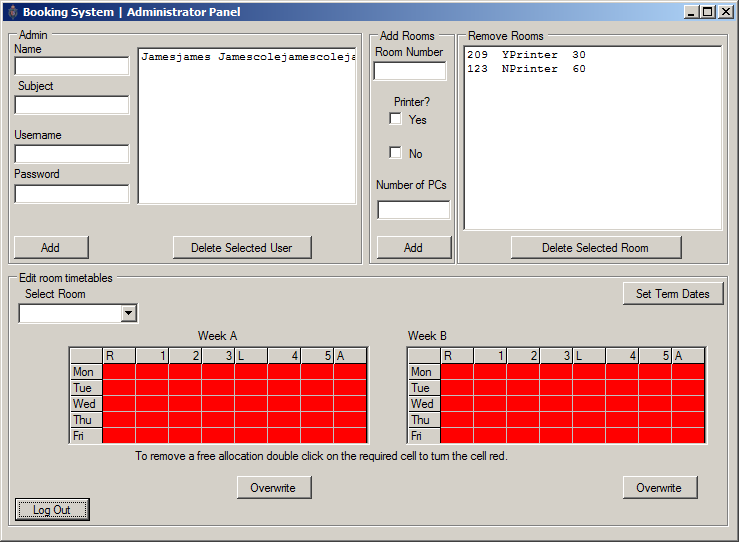
Screenshot 15



Screenshot 16

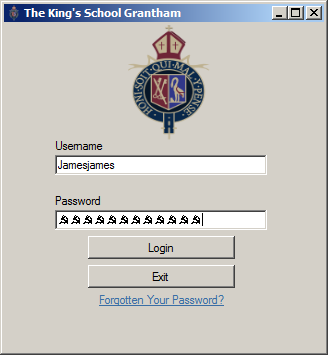
## 4.3 Extreme Data

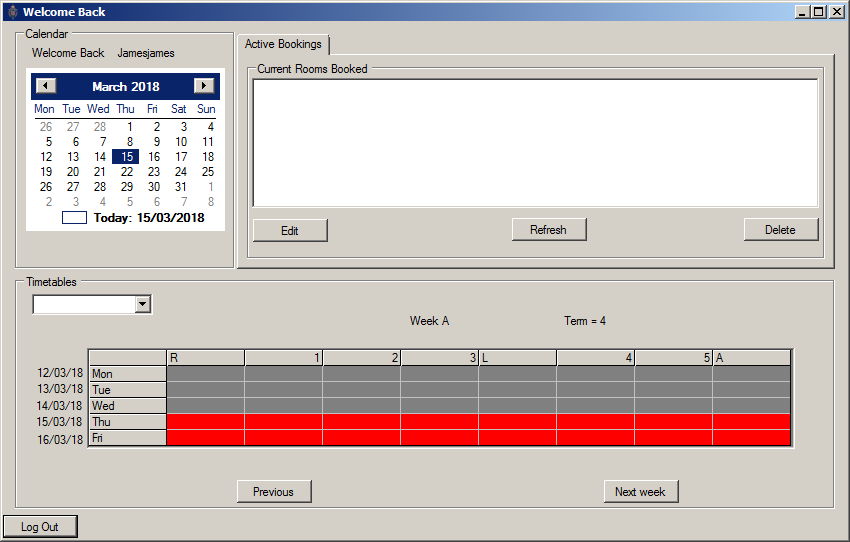
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test No.** | **Description / Purpose of test** | **Input Data** | **Test Method** | **Expected Results** | **Actual Results** |
| **1** | Book all periods in TWO days | Book all periods on Thursday and Friday. | Black box | The days will be full with greyed out booked squares. | See screenshot 17 below |
| **2** | Longest teacher name | Jamesjames  123456789112  Jamescolejamescolejamescolejam  artartartartartart | Black box | The list box will likely be unable to display all the information but the user will be able to login with their username and password | See screenshot 18 ,19 and 20 below |
| **3** | Save a booking as far in advance as you can | 20/07/18 last period | Black box | The room will be booked fine and the term it is booked in will be displayed | See screenshot 21 below |
| **4** | Add a room with the max size in all text boxes | 999 YPRINTER 99 | Black box | The room will be displayed and wont behave any differently on the system | See screenshot 22 and 23 below |
| **5** | Add a room with the smallest values in each text box | 001 NPRINTER 01 | Black box | The room will again behave normally in the system, being displayed as normal | See screenshot 24 and 25 below |



Screenshot 18

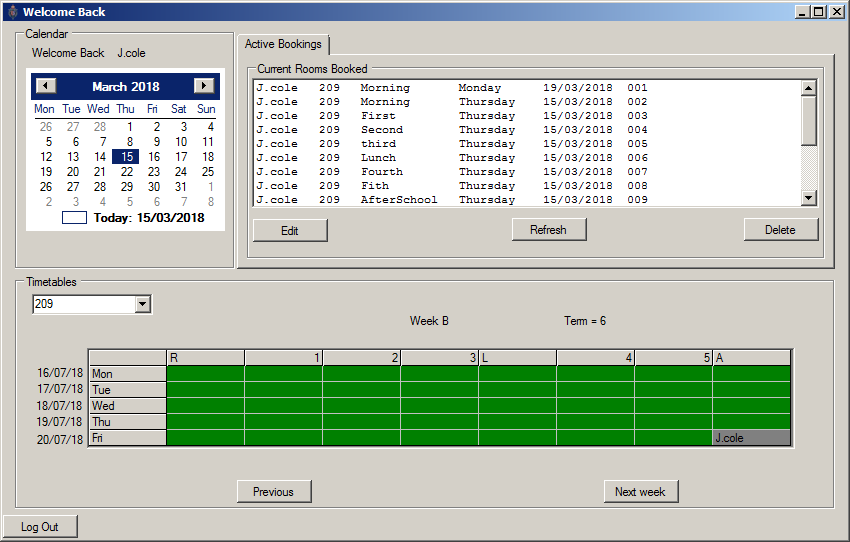
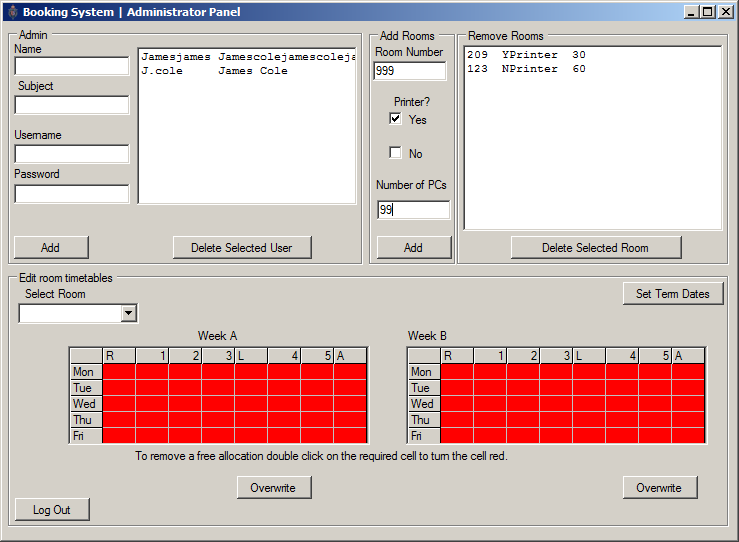
Screenshot 17





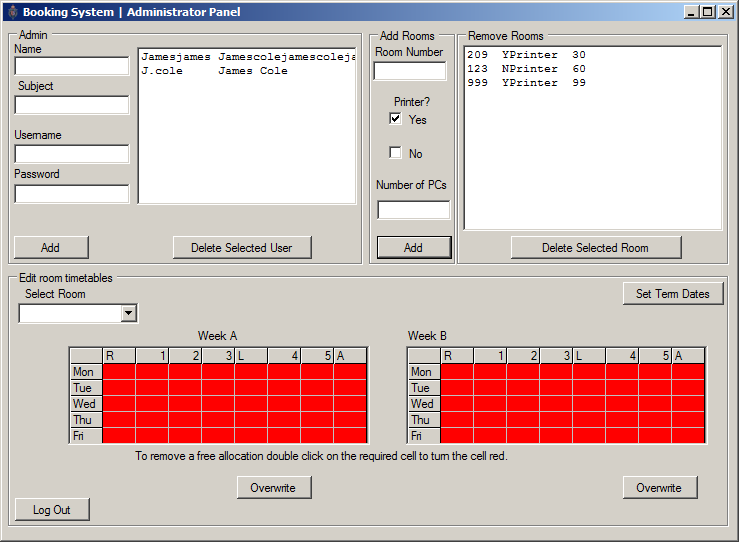
Screenshot 20

Screenshot 19

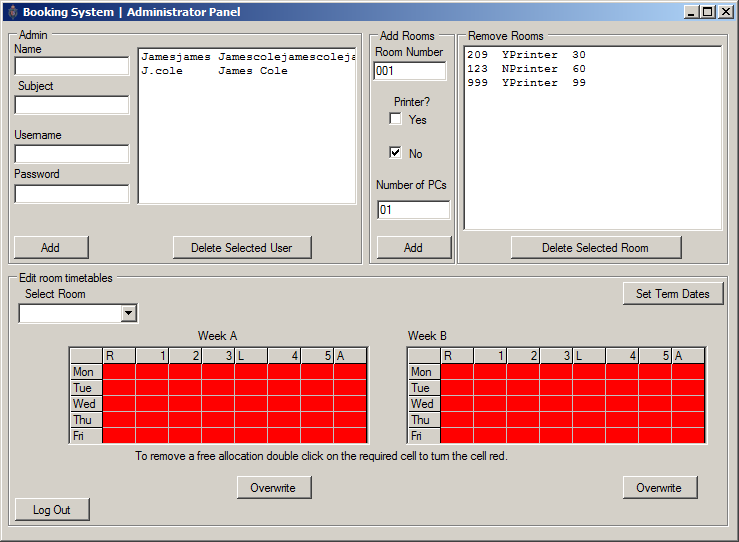


Screenshot 22

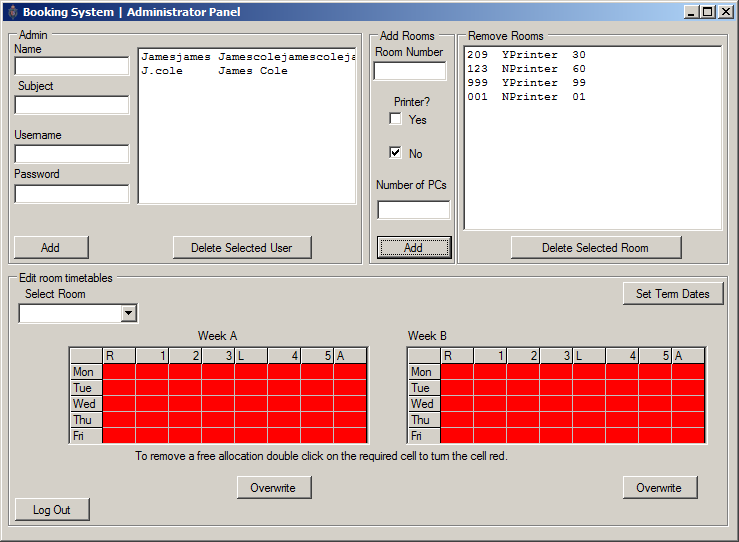
Screenshot 21



Screenshot 23



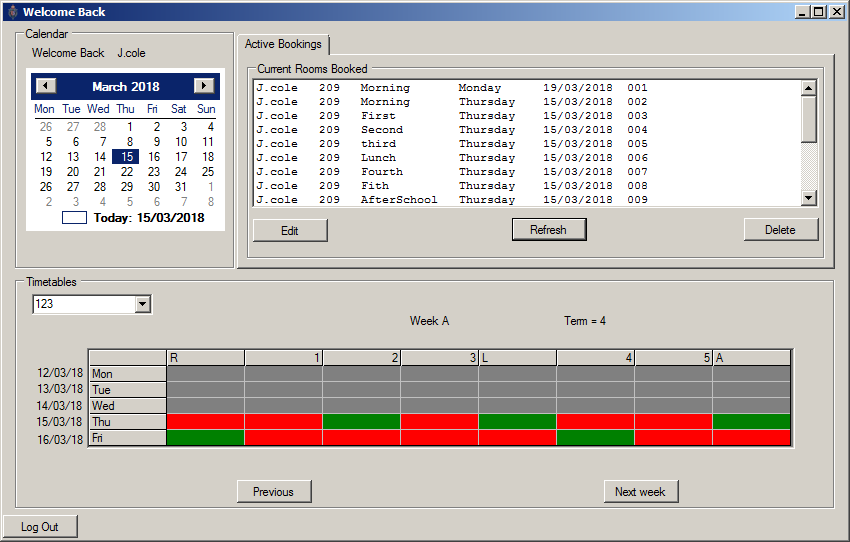
Screenshot 24



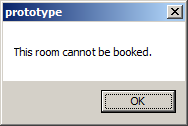
Screenshot 25

## 4.4 Erroneous Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test No.** | **Description / Purpose of test** | **Input Data** | **Test Method** | **Expected Results** | **Actual Results** |
| **1** | Try booking a room that isn’t available |  | Black box | Message box saying this room isn’t available will pop up and the room won’t be able to be booked. | Screenshot 26 and 27 below |
| **2** | Try booking a room that is already booked by another teacher |  | Black box | The new user will not be able to select the room and therefore cant book it. | Screenshot 28 below |
| **3** | Try adding a room without selecting if it has a printer | 123 …….. 45 | Black box | A message box will pop up and not add the room. | Screenshot 29 and 30 below |
| **4** | Try saving a teacher with no username or password | Sam art …………. | Black box | A message box will pop up and not allow the teacher to be saved | Screenshot 31 and 32 below |
| **5** | Try saving a room with no room number | … Yprinter 30 | Black box | Again, a message box should pop up and not let the room be created | Screenshot 33 and 34 below |
| **6** | Try entering invalid term dates | Start :01/01/2017  End: 01/01/1956 | Black box | A message box will pop up and the dates won’t be saved | Screenshot 35 and 36 below |

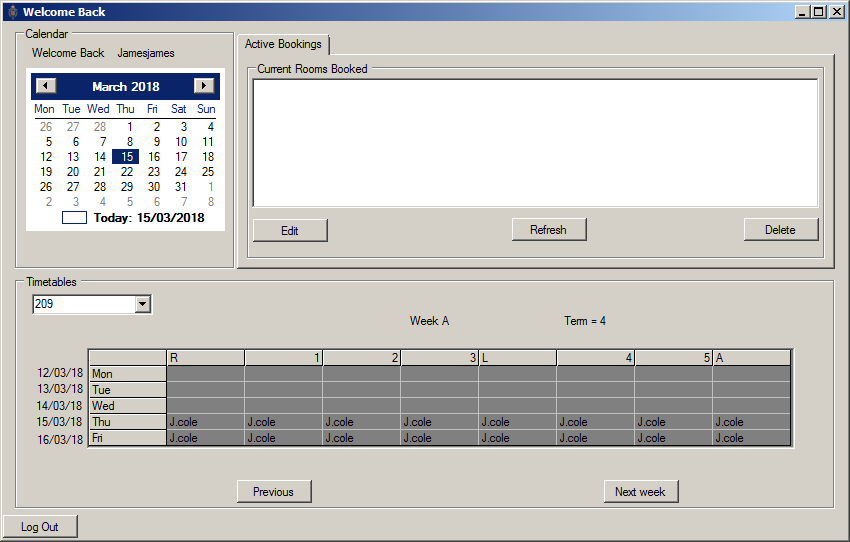


Screenshot 26

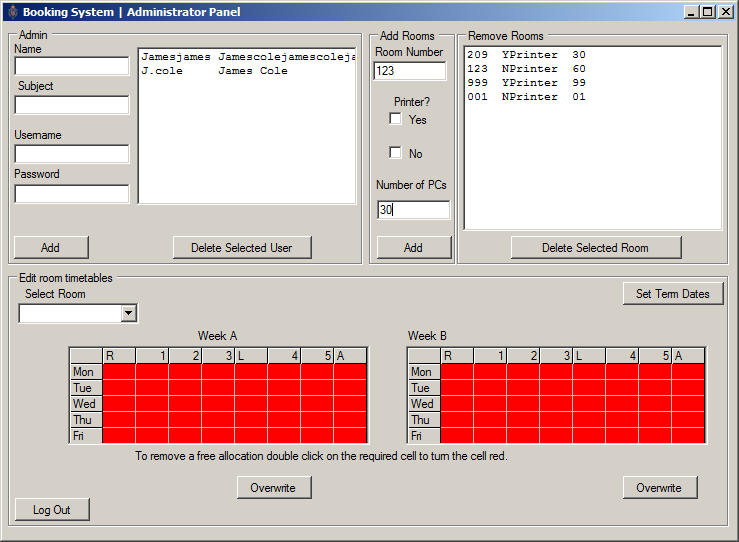


Screenshot 27

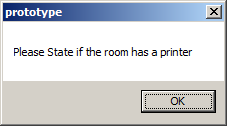
Screenshot 28



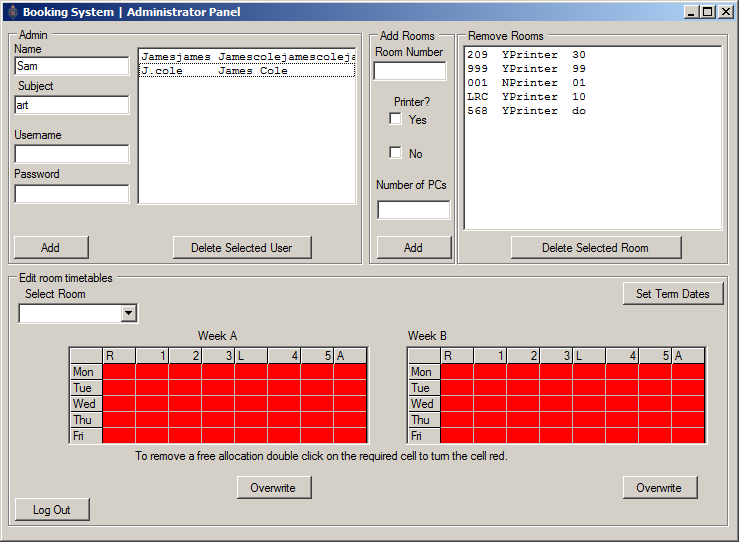
Screenshot 28



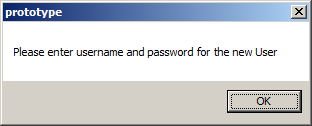
Screenshot 29



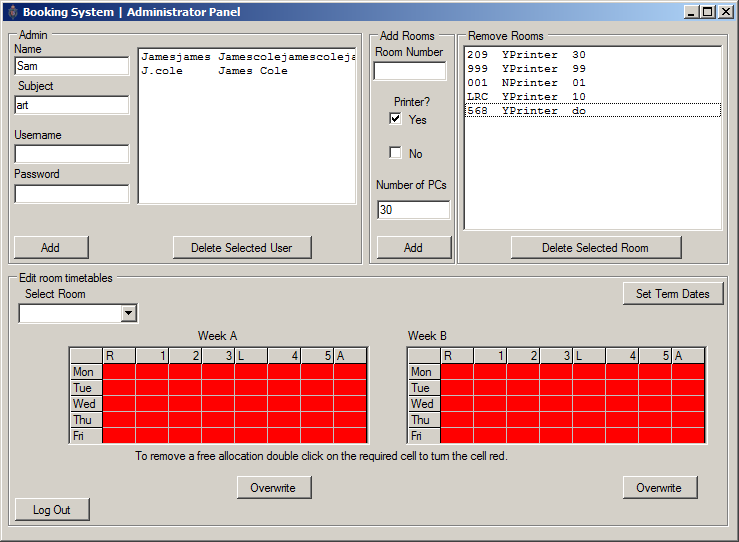
Screenshot 30



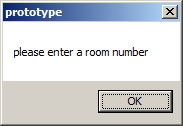
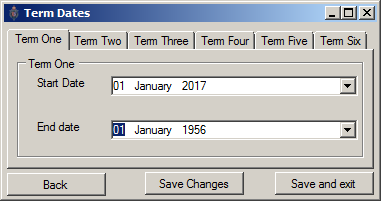
Screenshot 31



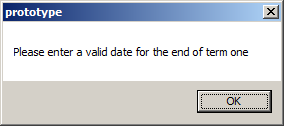
Screenshot 32



Screenshot 33



Screenshot 34



Screenshot 35

Screenshot 36

# 5. Evaluation

## 5.1 Overview

The new system works really well. It offers a graphical way of booking rooms which is much easier to understand than the previous solution. Furthermore, you don’t need any other programs in order to run my solution. The previous version required Microsoft office where as my program will run completely independently as the program creates all the files needed in order to operate. In addition, the layout of the program is such that you wouldn’t need extensive experience to use the program, it is nice and simple and the core functions are easily accessible. I think the fact I tried to use as few pages as possible to complete the program helps with the ease of use because the user doesn’t have to navigate through hundreds of pages to find the function they want. Furthermore, the files used for the program are constantly updated and so old data isn’t stored needlessly.

## 5.2 Review of Objects

The first key requirement was to make sure a user can book a room. This has been done by creating a dynamic random access file which constantly checks that the data in it is up to date and it therefore stays a manageable size. Furthermore, the way the rooms is booked is nice and easy. You simply search for the room you want and click on the relevant box which corresponds to the period you want to book.

The second requirement is to be able to cancel a booking. This is done by selecting the booking which was made in the list box on the user’s homepage. You highlight the booking and press delete. This removes the booking from the file and from the graphical page. This means other users can then book the room. This has been done successfully because it means the user doesn’t have to scroll through the year to find their specific booking, instead it can be done from a simple list of their active bookings.

The third key requirement is for the admin to be able to edit the rooms used in the program. This has been done by having a simple section on a page that only the admin can access. This means that normal users don’t have the ability to add or remove rooms. Only the admin has access to this. The way the rooms are added has been done in such a way which allows the information to be entered very quickly and only the necessary information is saved. This allows for rapid expansion of the school.

The fourth key requirement the program must do is to store the rooms timetable information. It does this through the use of two text files. The files contain “X”s or “F”s which tell the program when the room is free or not. This then corresponds with the graphical representation on the users screen. The timetable file then references a different file called bookings before it loads. This makes sure that in addition to any periods that are timetabled that the users bookings are displayed but not in such a way as to compromise the rest of the year’s timetable for the room.

The fifth key requirement is that the program has to have the ability to facilitate the large number of changing staff and as such needs the ability to change the users who can use the program. This is done in a very similar way to the rooms. With a small part of the admin page, the admin can enter the name and subject of the new teacher. It only saves the necessary information.

The sixth key requirement is that the program must be able to handle multiple bookings. And each user must be able to make more than one booking. The design of the booking method actually encourages multiple bookings because it doesn’t take a long time to make one. Also a person can see all their active bookings and so they can make lots and then they will be saved until the date passes.

The seventh key requirement is that the program must be easy to use. I have already touched on this. The program lends itself to effortless booking because, rather than filling in forms, the user can make all their bookings using a graphical interface which is as simple as clicking on the green squares to book a room. No need for the teacher to enter any information about themselves because the program handles all that for them.

The eighth thing it needs to do is to show a user when their next booking is. It does this by displaying all of a specific user’s active bookings in a table. This is personalised for each user as it only shows their bookings.

The ninth requirement is that the program has to have a reminder system. The program does this but it doesn’t do it for bookings. It does it for any appointments a user makes via the calendar on the left of the screen.

The tenth thing my program must do is have multiple users and each page must be personalised for that specific user. This has been done by saving logins in a file and then loading the bookings made by a certain person and only displaying their bookings rather than everyone’s.

The eleventh thing my program has to do is be updated in real time. This is so bookings aren’t processed in a batch and clashes can’t occur. This has been done successfully by processing requests as soon as they come in rather than a batch process at a non-peak time.

The program also has to be able to support term dates. This is done at the start of each year and then subsequently tells the user is it a week a or week b.

The final thing my program must do is only the admins and the original booker have the ability to cancel bookings. This is done by not allowing other users to have access to the file or see the bookings in their list. As a result, they can’t remove them. They can however see it on the graphical table. This allows them to get in touch with the teacher and request a change.

## 5.3 User Feedback

1. Is the new program an improvement on the current solution?

Yes, the new system is an improvement as it allows teachers with a limited knowledge of computers to use the program. Furthermore, it is clearer as it doesn’t have lots of tables on one page. It simplifies the problem and solves it elegantly.

2. Is there anything missing from the program?

The program does not display the information about the room for the user. This would make selecting a room for new users who are unfamiliar with the school easier.

3. What aspect of the program is the easiest to understand?

The way you actually book the room is the easiest thing the program does. This is how it should be because it’s such a large part of the program. It means a user can make lots of bookings in the middle of teaching. In addition, it makes lesson planning easier because you can see which room is free as you are planning and plan the lesson in advance to meet when you book the room.

4. In an ideal program what would you add?

The program is nearly perfect, the only small thing I would add would be the ability to see a departments phone number in the program.

5. Overall, are you happy with the new program?

Yes, I am happy with the new program is it full fills all the needs we set out at the start of the project.

## 5.4 Improvements

The user feedback has given me a few things I would add to the program if I were to do it again. The first of which is that the program should display the hardware in the room when the user books the room. They should also be able to refine their search based on software/hardware. This would make it easier for people new to the school who don’t yet know the rooms. This could be done by adding a few labels to the form and loading the information from the room file to fill it in. I could also store the programs that are specific to those rooms. For example, a computing set would require the visual basic program and so if the teacher needed to change room they would have to make sure the new room also had these programs. Furthermore, I would add the ability to send users a message to request a room change. This would be easy to do as I could create a standardised message that could be sent at the push of a button. The message could say. “(JCOLE) is requesting a room change with you on the (13/04/18, period 3, room209) would you like to accept the change now?” furthermore I could also store the head of departments phone numbers on the program with a simple table which links the department given to a teacher to the phone number associated with it.

Bibliography

<https://www.deskflex.com/Home/gclid/CM3n4OmButECFUueGwodM7wI4w>

<http://www.littlehotelier.com/front-desk-system/>

<https://www.skedda.com/home/features>