Newtown Hospital Radio

A2 Computing Project 2001/2002

Contents

Analysis

Background to the problem

Newtown Hospital Radio is a voluntary organisation, and a registered charity. They broadcast music shows, as well as some talk shows, to Newtown and Formby District General Hospital twenty-four hours a day.

There are three groups of volunteers who work at the hospital radio: The support staff, the presenters, and the executives.

The support staff completes the more menial tasks that are, nonetheless, essential to maintaining the hospital radio. These tasks include visiting the wards to get requests from patients, keeping the studio clean and tidy, and keeping the record library in order. Also, technical support staff must keep all of the studio equipment in full working order. There are more specific support staff, too, such as the secretary and the treasurer, who complete all the more demanding jobs at the station.

The presenters are responsible for recording a new edition of their show every week onto MiniDisc. Each show is played several times each week, so that the presenters do not have to record too much material.

The executives are in charge of the radio station. The executives are made up of elected representatives from each of the groups of volunteers, as well as a chairperson. They are responsible for all of the major decisions about the running of the radio station.

I have chosen to base my project on the Newtown Hospital Radio as I have worked with them in the past on many different projects, and because they are desperately in need of a new system to replace the current one.

Identification of the problem

Currently, an index of the record library is kept using a system of index cards. One set of cards is arranged in alphabetical order of song title, and one set is arranged in alphabetical order of artist. However, the number of items in the record library is beginning to grow to such a level that an index card system is rather impractical. Also, if a request for a specific song is made, but neither the full title or artist's name is known, then the song cannot be found.

Member's details are also currently kept in an index card system. There are also several problems with this system. For example, if a member of Support Staff needs to be contacted, then the secretary must look though the cards (which are arranged in alphabetical order of name) until a member of Support Staff is found. Also, if all executives are to be contacted, then the entire index card system must be browsed to find those members who are on the executive committee. As the number of members grows, this is getting to be rather time-consuming. Also, it is inevitable that index cards go missing, as people remove them to copy out an address or a phone number, and then forget to replace them. This makes the whole system totally inaccurate.

Twenty-four hour broadcasting is a new venture for Newtown Hospital Radio, and has required an increase in the number of presenters. Previously, presenters were allowed to record their shows in the studio whenever they had a spare hour. But, as more presenters are recruited, several have arrived to record their show at the same time. This has led to a number of disagreements within the organisation, and, ultimately, several presenters have moved to other hospital radio stations. Currently, no system is in place to deal with this problem.

The data that I will use in my systems once they have been constructed can be easily researched from the current index card system.

Identification of the prospective users

There will be many different users of the new system, since all support staff and presenters will need to be able to find records using the system, and all presenters will need to be able to book time in the studio to record their shows. Also, executives may need to access the member database to communicate with volunteers. Therefore, there will be many different users with varying degrees of computer experience. Therefore, the system will have to be very user friendly, since some users have never have tried to operate a computer before, and few have the time to be trained how to use it. Operation of the system must therefore be clear, so that most people will be able to operate it without training.

Identification of User Needs

The system must be able to do the following:

- Store and allow update and deletion of information about each song in the record library
  - Track ID (So that it can be found in record library). Track ID is made up of a four digit number (MiniDisc number), followed by a dash, and then a two digit number (Track Number)
  - Track Title
  - Artist
  - Style of music (Jazz, pop etc.)
  - Length of track
- Find a track or tracks by searching on any of the above criteria. The results of this search will be displayed on the screen for immediate reference. Hard copies would not be beneficial since the majority of presenters plan their shows from the studio.
- Allow the user to browse through the songs available and view details of any chosen song.
- Store and allow update and deletion of information about each member
  - Name
  - Address

    - Telephone Number
    - Status (Support / Presenter / Executive Support / Executive Presenter)
- Find a member or members by searching on any of the above criteria. Again, the results of this search will be displayed onscreen.
- Allow the user to browse through a database of members view the details of any chosen member.
- Allow a presenter (and **only** a presenter) to book an hour when they can use the studio. Although not originally suggested, after discussion with current members, a system whereby a data sheet of bookings could be printed has now been suggested. Therefore, my system will also allow the printing of a list of bookings made by members. Also, a list of tracks that each presenter wants to use during their booked time will be stored, and printed on the booking data sheet.
- Print members' details ordered by status should be available, as this could serve as a useful telephone directory.
- Print a list of all of the tracks in the music database should also be made available, so that the few presenters who choose to plan some of their shows at home are catered for.

I will complete this task by using a Microsoft Access Database, fronted by a Visual Basic program. This way, I will be able to customise the look and feel of the way in which the database is accessed, and I will be able to do much of the complicated searching and querying "Behind the scenes", thus making the system very easy to operate.

There are relatively constraints placed upon the system, as Newtown Hospital Radio is equipped with a computer and Microsoft Office. Since the database is never, at least in the foreseeable future, going to reach millions of data items stored, then processing power is not going to cause too much of a problem, as the system is fairly up-to-date and has a powerful processor. Also, the 20Gb hard disk should be ample for this system. All of the hardware needed to run my system is clearly available.

A backup is already made of the system on a weekly basis, as the computer is currently used by the secretary for typing letters. Backups are made on Zip Discs, which are then stored at another location (the secretary's house). Using specialist software, the technical manager is able to remotely take control of the computer from home, so that any problems with the system can easily be fixed (the technical manager is computer literate).

Realistic Appraisal of the Feasibility of Potential Solutions

There would be many ways of approaching this project, using software and programming languages as described below:

- Pascal: A program could be written in Pascal to handle the above data.
- Access: Microsoft Access could have been used on its own, without any fronting being completed by a Visual Basic program.
- Excel: Microsoft Excel could be used to store the data.
- Basic: A program could be written in Basic to handle the data

- Manual: A manual system, with a few modifications and additions to the current manual system, could be used to store the data.

Justification of chosen solution

I chose not to complete the project using the above solutions for these reasons:

- I did not want to complete the system in Pascal or Delphi, since a user interface with which the majority of people would be familiar (i.e. a Windows-style interface) would be very difficult to create using this programming language.
- I did not want to complete the project using Microsoft Access alone, as this would have made it very difficult to present the user with an interface that is easy to understand. It would, realistically, have meant that the user would have to be familiar with writing Microsoft Access queries. Although, theoretically, a user can be protected from this by using Microsoft Access's own programming language, this is extremely difficult, and would probably have resulted in a very high maintenance system.
- The project could not practically be completed in Microsoft Excel. Again, attempting to build a system of this scale in Microsoft Excel would make it very difficult to present an easy-to-use, intuitive user interface, since many advanced options, which the user does not need to use, would be ever present.
- Basic has the same inherent problems as Pascal: It is very difficult to build a windows-style environment using this language.
- A manual system would not be able to fulfil all of the criteria set out for the new system, since it would be impossible to search a manual system. Furthermore, a computerised system can have regular backups made with greater ease, and security of information can be improved. Also, presentation of printed material is far better than presentation of hand-written material, and can reduce error (e.g. when information about a record is copied from an index card, it is very easy to incorrectly copy the record number, which ultimately results in the user being unable to find the record.)

I have chosen to complete my project in Visual Basic and Microsoft Access as I familiar with these, and a suitable system can be built using them. I will also be able to decide exactly what I want to show to the user, and so will be able to hide any unnecessary options from the user, whilst still maintaining a windows style.

There are, however, some disadvantages of my system. For example, those with absolutely no computer experience will need training in its operation. However, since the computer software is already in place, I do not think that the new system will be any more expensive than the current one.

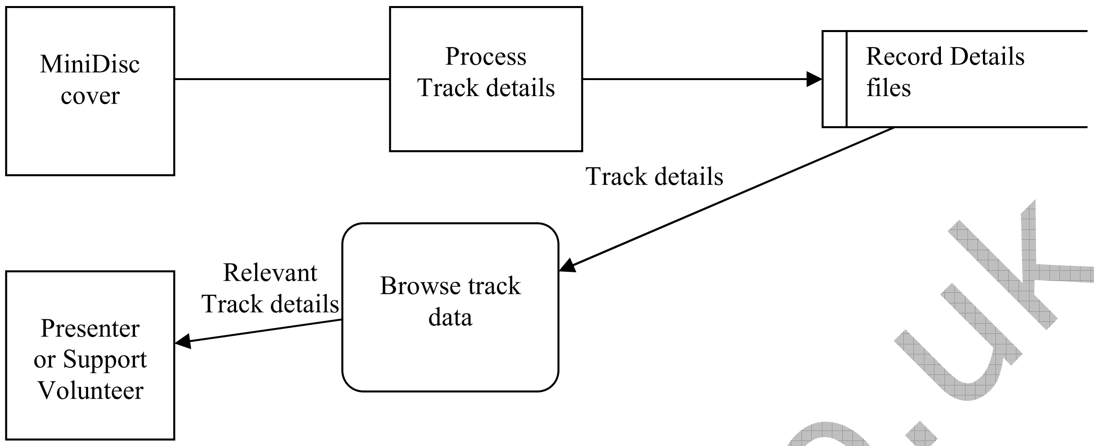
Data source(s) and destination(s), logical DfDs, and an E-R model

For this section, it seems logical to split the project into three separate sections:

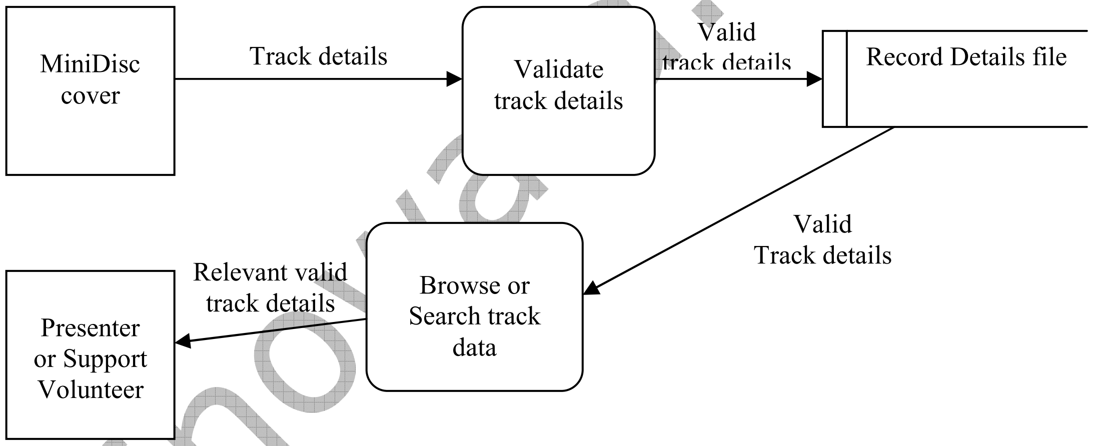
- Record database
- Member database

- Booking system

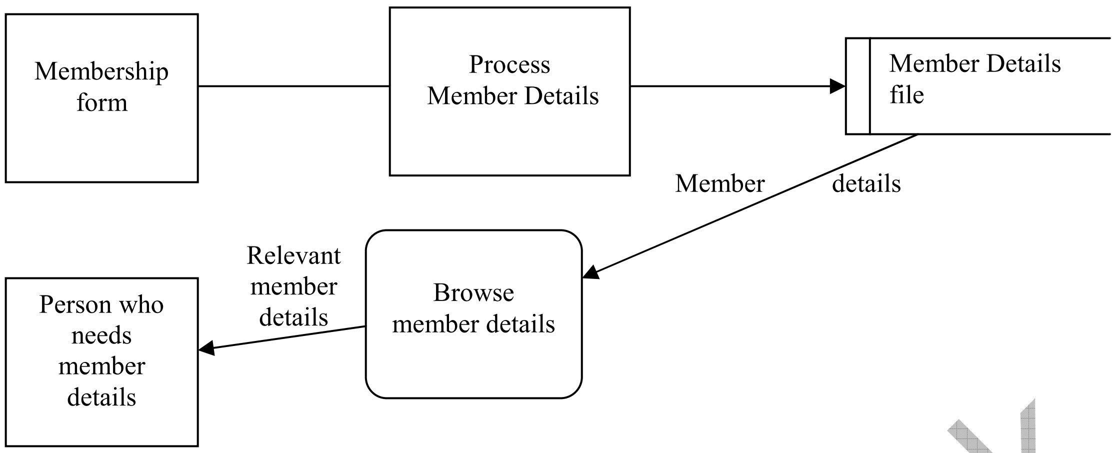
In the record database, the data that comes from the cover of the MiniDisc is entered into the manual database, for retrieval at a later date by a presenter or member of support staff:



The new system will be much the same, but only one Record Details file will be kept. Also, searches and validation will take place:

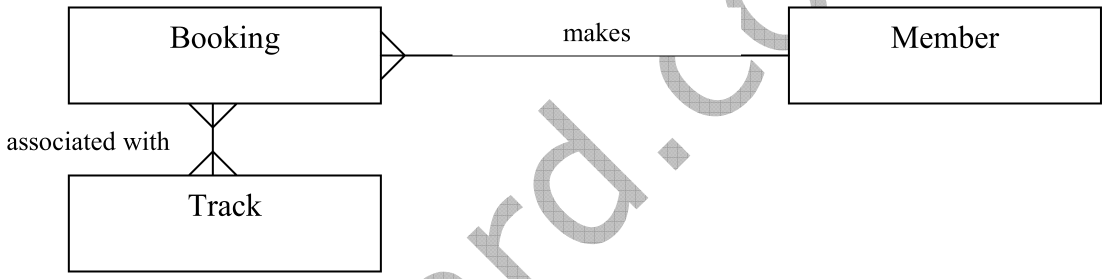


Currently, the member database is very similar to the record database in terms of data flow:



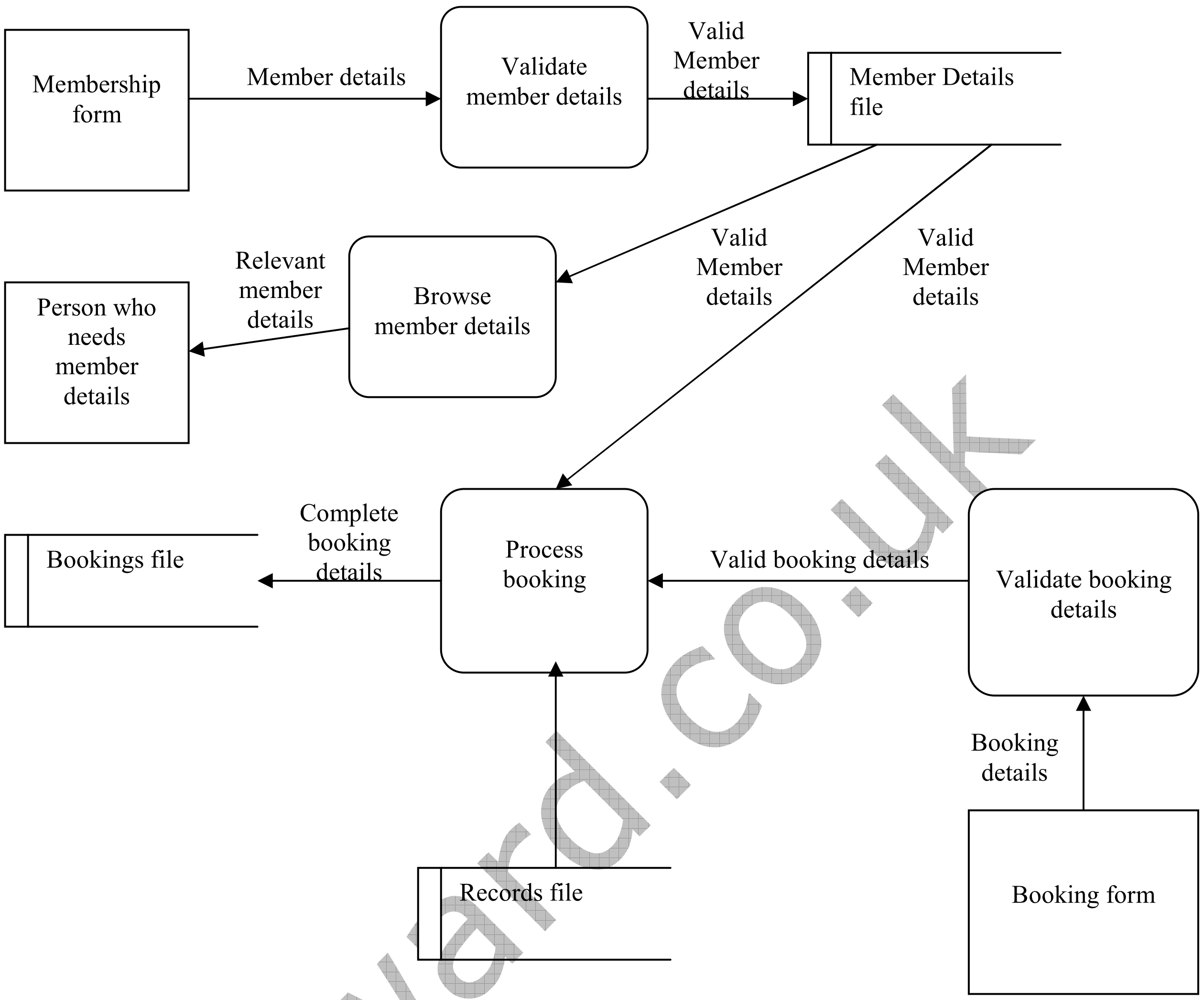
However, the new system will have a slightly more complicated method of data flow, as a booking system is to be added.

The entity relationships for this system will be as follows:



Thus, one presenter could, in theory, have more than one booking per week.

The data flow in the proposed system will be as follows:



Objectives of the Project

The objectives of this project are to fulfil the users' needs by producing a system that can:

- Store and allow update and deletion of information about each song in the record library
    - Track ID (So that it can be found in record library). Track ID is made up of a four digit number (MiniDisc number), followed by a dash, and then a two digit number (Track Number)
    - Track Title
    - Artist
    - Style of music (Jazz, pop etc.)
    - Length of track
- Find a track or tracks by searching on any of the above criteria. The results of this search will be displayed on the screen for immediate reference. Hard copies would not be beneficial since the majority of presenters plan their shows from the studio.
- Allow the user to browse through the songs available and view details of any chosen song.
- Store and allow update and deletion of information about each member

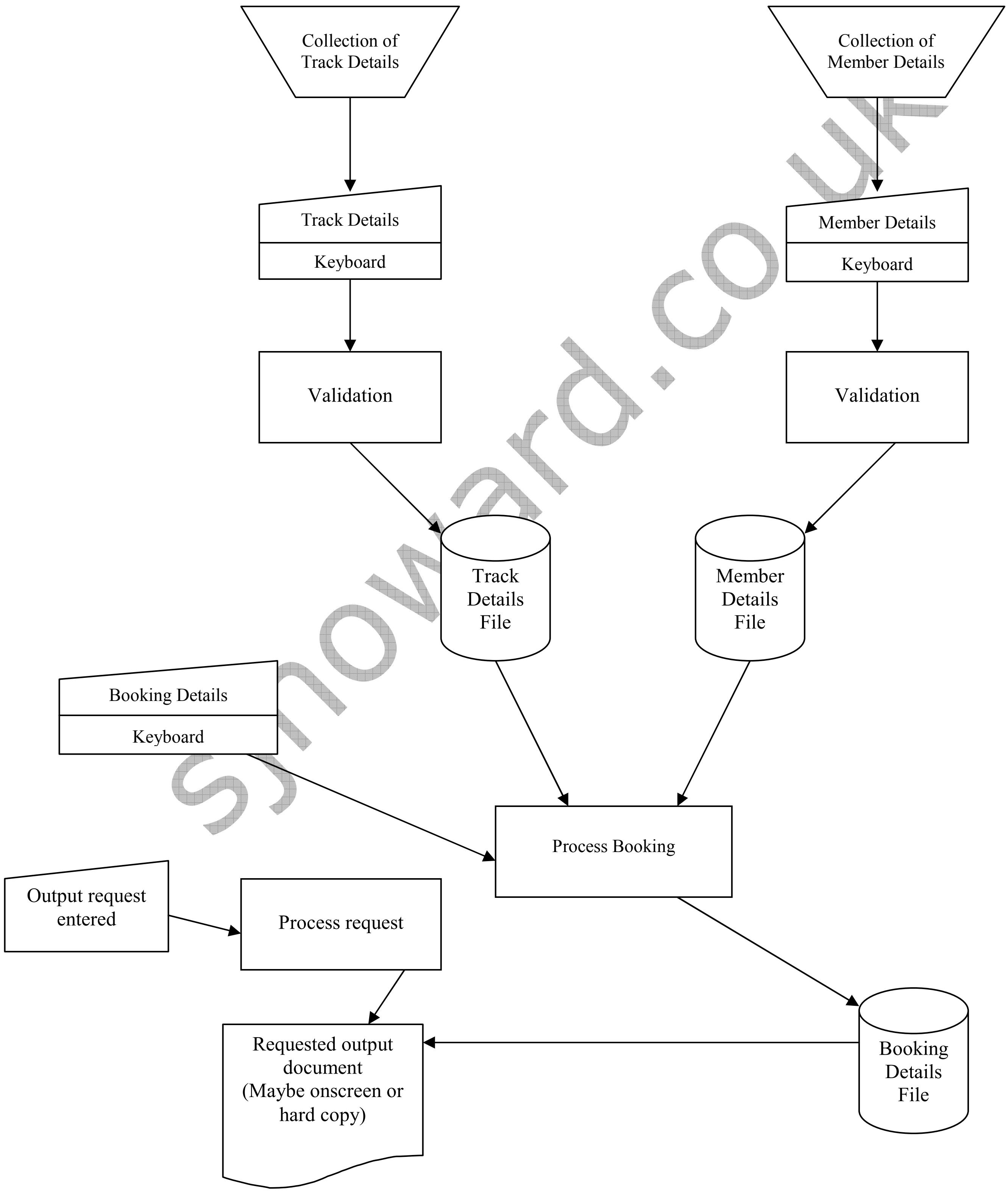
- Name
- Address
- Telephone Number
- Status (Support / Presenter / Executive Support / Executive Presenter)

- Find a member or members by searching on any of the above criteria. Again, the results of this search will be displayed onscreen.
- Allow the user to browse through a database of members view the details of any chosen member.
- Allow a presenter (and **only** a presenter) to book an hour when they can use the studio. Although not originally suggested, after discussion with current members, a system whereby a data sheet of bookings could be printed has now been suggested. Therefore, my system will also allow the printing of a list of bookings made by members. Also, a list of tracks that each presenter wants to use during their booked time will be stored, and printed on the booking data sheet.
- Print members' details ordered by status should be available, as this could serve as a useful telephone directory.
- Print a list of all of the tracks in the music database should also be made available, so that the few presenters who choose to plan some of their shows at home are catered for.

Design

Overall system design

My system will not be entirely automatic – some manual processed will obviously be required. However, the manual processes should work together with the computerised processes should work together efficiently to provide a full working solution. I will construct the system to allow dataflow in this pattern:



Modular Structure of System

The system as shown in the system flow chart can be thought of as being made up of several different modules, each with its own section of coding. These are:

- Validation Modules: There are two independents validation modules which are the ones which validate the entry of data about members and tracks. However, there a further validation module will be "built in" to the Process Booking module, to ensure that only valid bookings are made. For more information on the validation process, see "Validation" on page 14.
- Process Bookings Module: This will take data from both the Member and Music files in order to build up and store a valid booking.
- Search Modules: These will have their own algorithms to search the databases for the relevant information. For more information on the algorithms, see "Algorithms" on page 14.

Other parts of the system, which cannot be truly referred to as "Modules", are also necessary. These manual processes are:

- Collection of Track Details: This involves the typing in of details about each track. More information can be found in "Data Capture and Entry" on page 17.
- Collection of Member Details: This involves the typing in of details about each member, and keeping these details up to date. More information can be found in "Data Capture and Entry" on page 17.

Data Requirements

The best way to express the data requirements of the system is in a data dictionary.

Table		**Field**					
Name	**Description**	**Name**	**Description**	**Data Type**	**Length**	**Default Value**	**Validation**
Music	Holds data about the music collection	ID	Track ID (Already implemented). Primary Key.	Text	7		Must be of the format 0000-00
		Title	Title of piece of music	Text	30		
		Genre	The type of music stored	Text	10		
		Artist	The artist or composer of a piece of music.	Text	30		
		Length	The length of the given track in seconds.	Integer			Must be between 1 and 3600.
Members	Holds data about the members	ID	A unique member ID. Primary Key.	Text			
		Name	Member Name	Text	30		
		Address Line 1	Member's Address Line 1	Text	50		
		Address Line 2	Member's Address Line 2	Text	50		
		Address Line 3	Member's Address Line 3	Text	50		

		Postcode	Member’s Postcode	Text	8		
		Phone	Member’s Phone Number	Text	12		
		Status	Member’s Status				
Bookings	Hold details about studio bookings	BookerID	Member ID of person who booked the studio.	Integer			The member must be a presenter
		Tracks	Series of Track IDs associated with this booking.	Text	140		
		Time	Time of booking (Start of 1hr slot). Combined primary key with date.	Time			
		Date	Date of booking. Combined primary key with time.	Date			

Because of the many-to-many relationship involved in this data, a separate “link” table containing the fields Date, Time and Track ID will be necessary to normalise the database. The E-R model can then be constructed as shown in “Database Design: E-R Model” on page 19.

Storage Media and Format

The data will be stored as Access Database files, and will be stored on the computer’s hard drive, and backed up regularly onto Zip Disc. Storing the files as Access Database files is the most practical solution, since I will be accessing the files from Visual Basic via Microsoft Access. Since these files are of a recognised and widely used file format, they will also be easy to use with other programs if the system is changed in the future.

In terms of a volumetric calculation:

> The “Music” table could have up to about 70 characters => 70 bytes per record and will hold approximately 2000 records => 140kB
>
> The “Members” table could have up to about 150 characters => 180 bytes per record and will hold approximately 30 records => 4.5kB
>
> The “Bookings” table could have up to about 150 characters => 150 bytes per record and about 20 records will be added per week => 3.6kB added per week => 187.2kB added per year

Therefore, the database cannot be expected to grow over about 350kB over a year, or 2.5MB over ten years. Therefore, the 20Gb hard drive is ample.

Algorithms

The main algorithms used in the project will be those for the searches. The search algorithms will be so similar that it only seems worth including one of them at this stage, since they will be fully documented in the program listings. Therefore, I have only included the algorithm for the search of the member database:

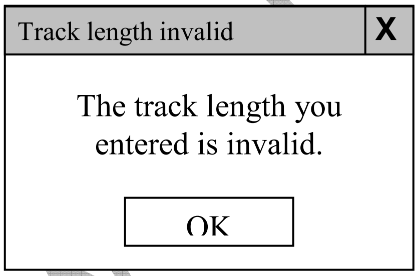
```
    Dim sql As String
```

```
sql = "SELECT * From Client WHERE
(((UCASE(Client.ClientId)) Like '%?%'));"

sql = Replace (sql, "?". Ucase(txtSearchFor.Text))

adoClients.RecordSource = sql

adoClients.Refresh
```

Validation

The validation procedures will validate the data as detailed in the data dictionary in "Data Requirements" on page 12. I will leave the error handling to Access, as I do not feel that there is any way in which I can improve upon the error handling provided. However, to reassure the user, I will intercept the default error message with my own, as new users may not understand those provided by Access. I will also put the incorrect field in focus so that the user can correct it.

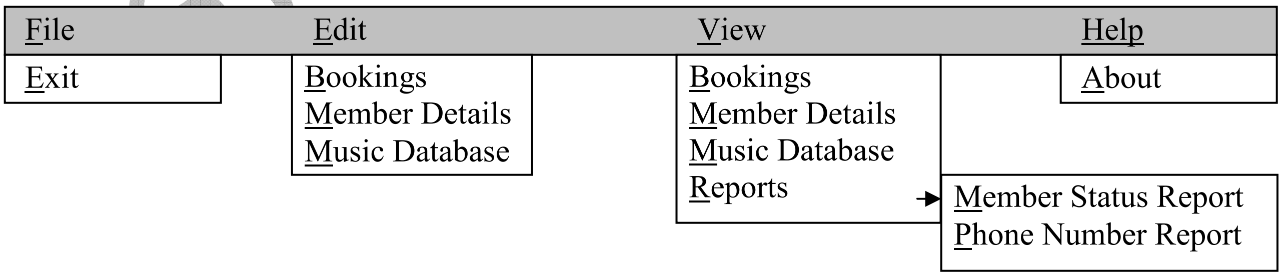
The error message will be a VB message box, and so will look something like this:



The track length you entered is invalid.

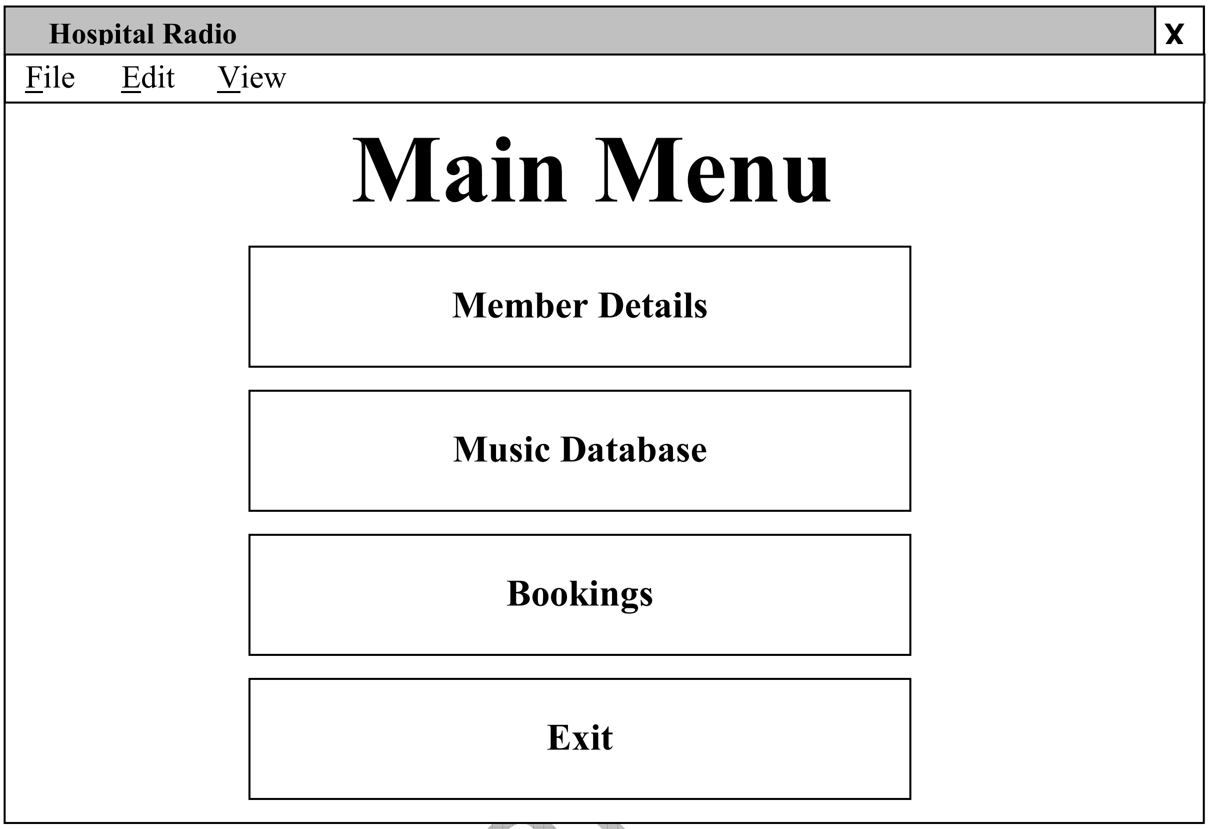
User Interface Design

In order to navigate the system, menus will be used. Before looking in detail at the form design of the menus, it is logical to look at the menu structure as a whole. This is shown below. I have designed it to be as logical and intuitive as possible, so that each part of the program is easy to find. The menu structure is shown here:

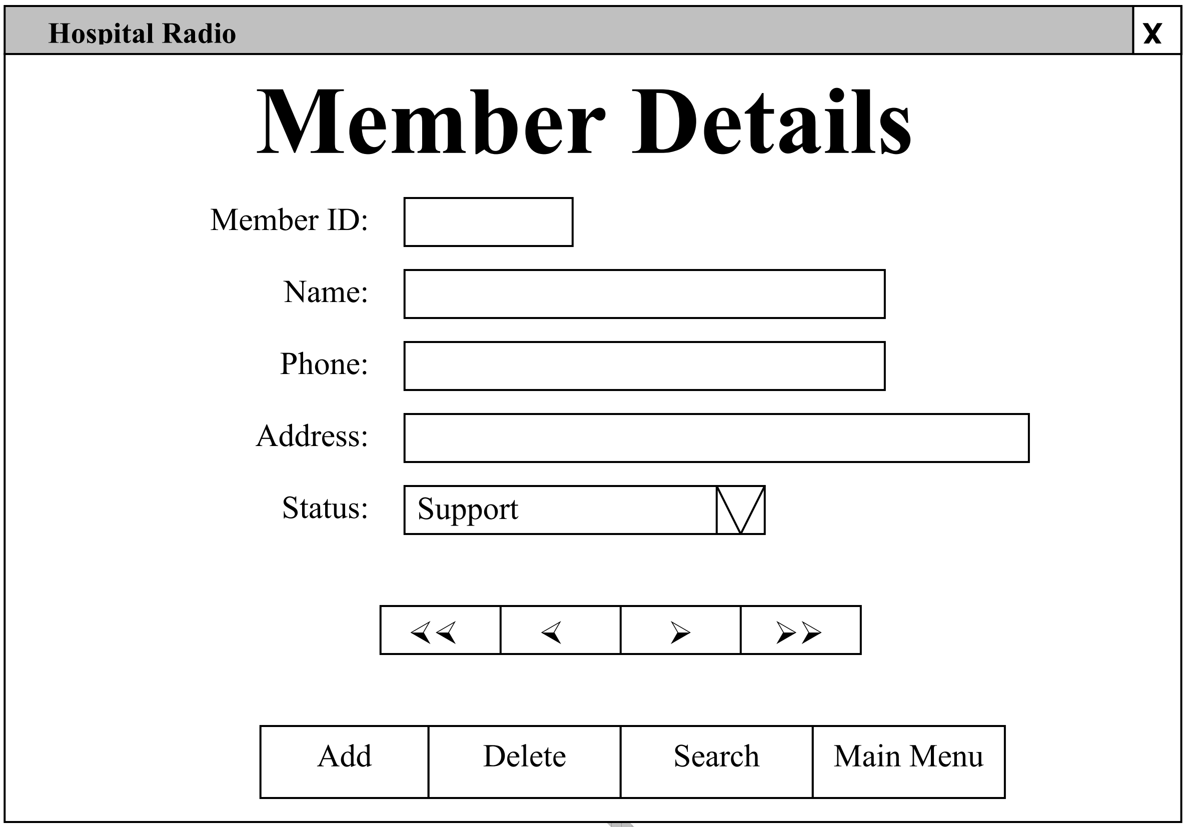


Clicking on the large command buttons to navigate the system will take the user directly to the editing dialog as relevant. This will mean that the less computer-literate users will have access to all the data without becoming confused over more complex features such as reports.

The menu forms will look like this:



The Member Details form will be designed like this:


Hospital Radio

Member Details

Member ID:

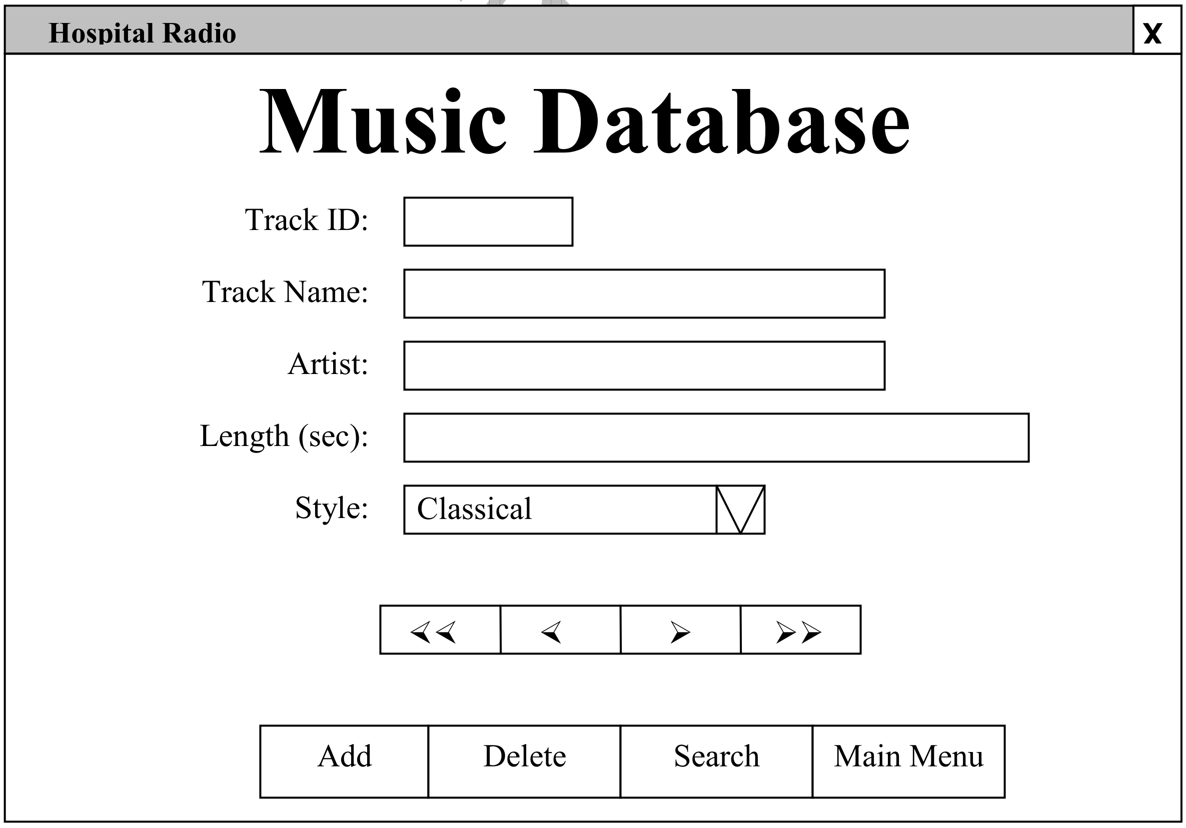
Name:

Phone:

Address:

Status: Support

When the form loads, the first member's details will be displayed in the text boxes. Changing the details as displayed on the screen will edit the database. This system will also be used in the music database:


Hospital Radio

Music Database

Track ID:

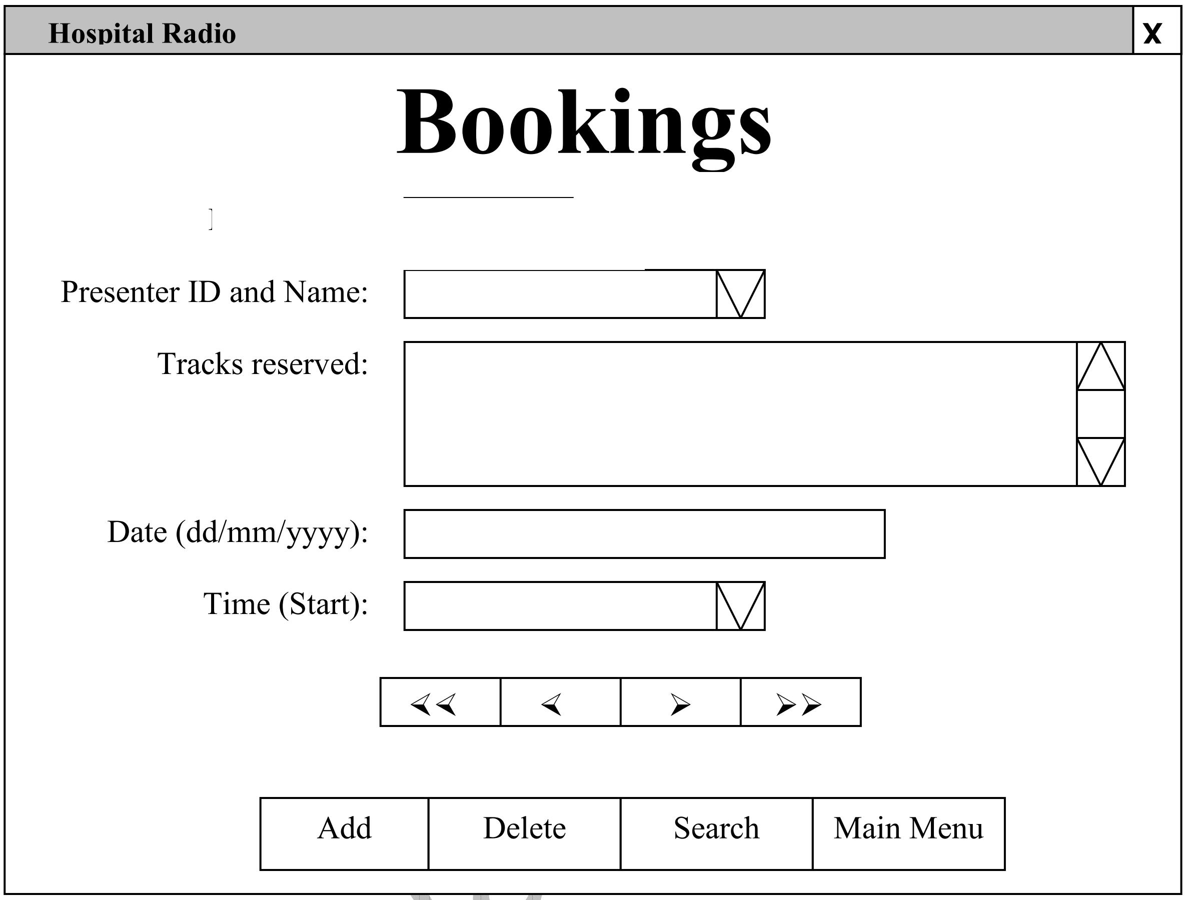
Track Name:

Artist:

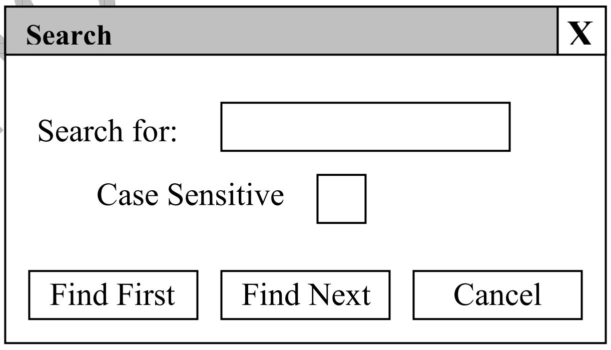
Length (sec):

Style: Classical

The bookings display will need to be slightly different, but the same principle will apply in that the first booking will be displayed when the form is loaded. The form will be designed to look like this:



In order to maintain consistency throughout the program, all of the search dialogs will be designed to appear similar. They will all look like this:



The reports will appear in Visual Basic's default report window. The reports themselves are discussed in "Output" on page 18.

Data Capture and Entry

For the data capture of the member details, a new member data capture form, which has been tried and tested over several years is already in place, and I see no good reason to change this. This form is given to all new members, so that their personal information can be recorded. They are required to inform the secretary of any changes to their information, and so the secretary can then update the new system in the same way that he currently updates the system in place at the moment. The data will be entered through the system which I have designed by navigating to the Member details screen and clicking "Add". The relevant information can then be entered via the keyboard.

The data for the music database can be found on the cover of the Minidisks, and is currently entered into the old system by support staff. There seems no valid reason to change this system, so the support staff can continue to enter the information by typing it into the new system. To do this, they will simply need to navigate to the "Music Database" form and click "Add". They will then be able to enter the data for the new track to add it to the database.

For the bookings system, the presenter who wishes to book the studio can simply enter the relevant information into the system when they are at the studio. Alternatively, they could phone the studio and someone could enter the booking information for them.

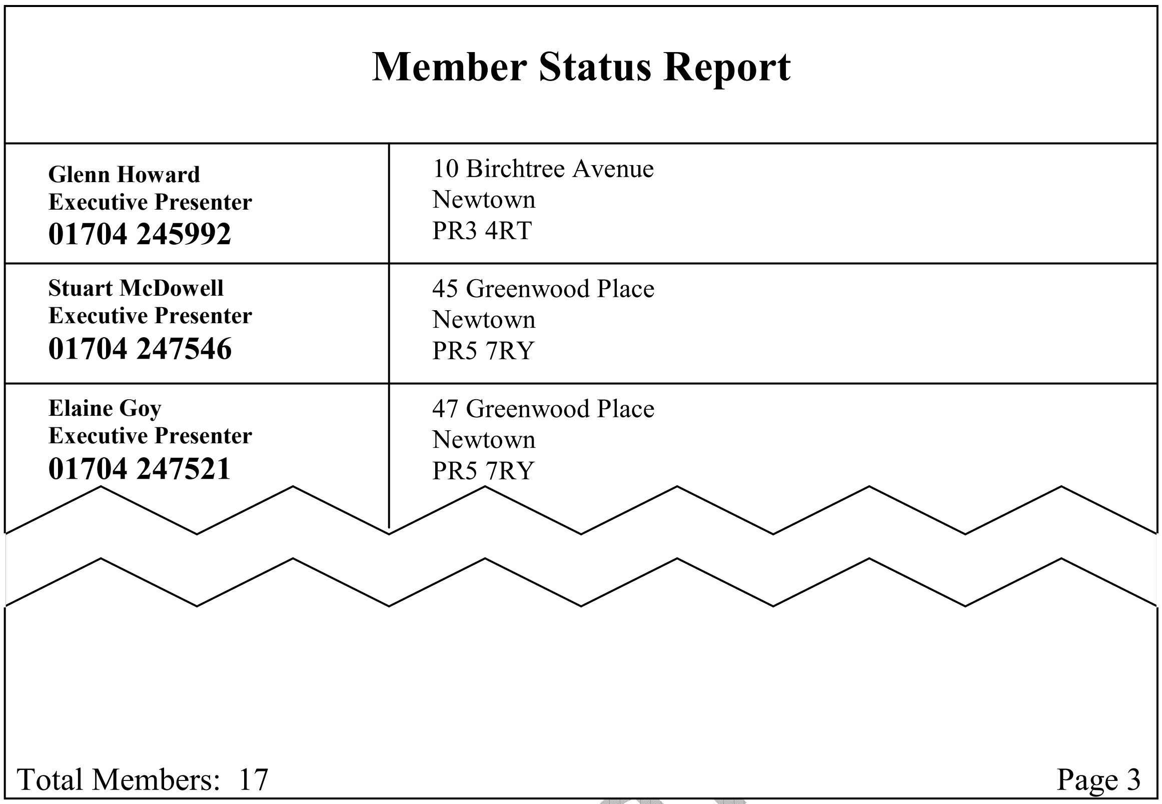
Record / Database Structure

The database will be made up of three tables: Music(ID, Title, Artist, Length), Members(ID, Name, Address, Status) and Booking(ID, BookerID, Tracks, Time, Date).

For more information on these, see the data dictionary under "Data Requirements" on page 12.

Output

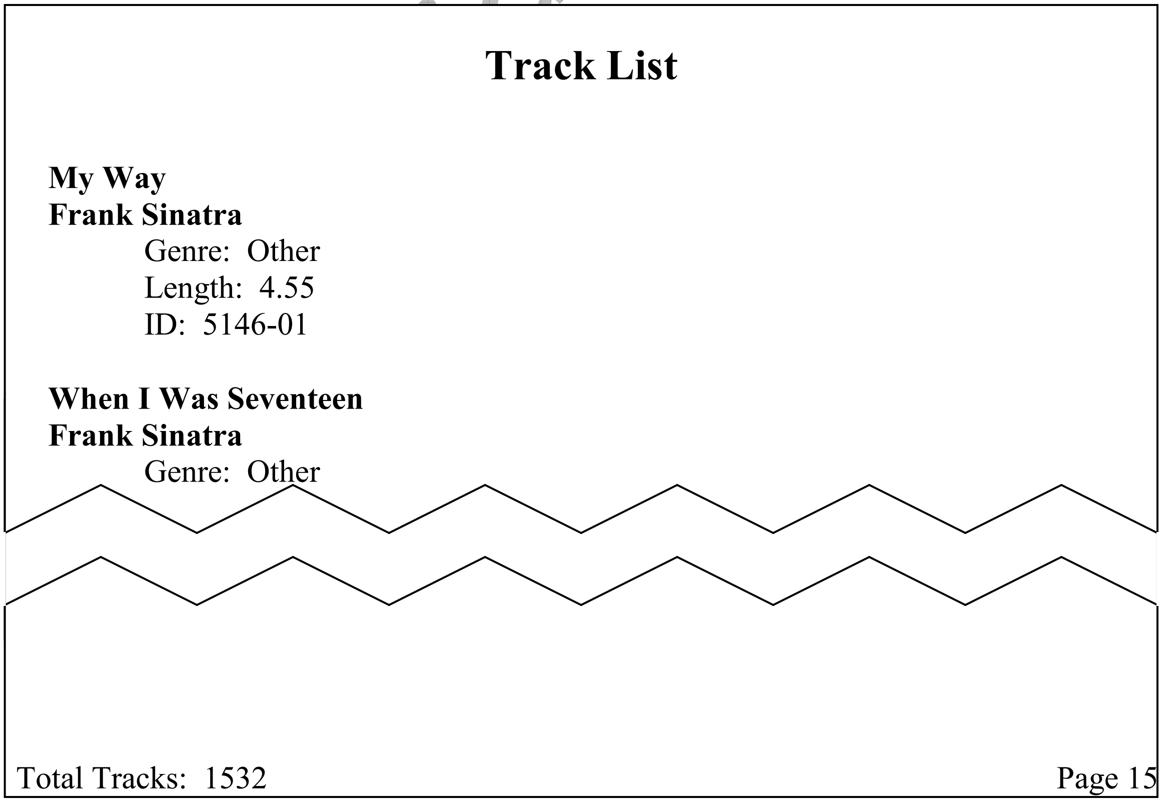
The hard output from the system will consist of several reports which can be printed if the user so wishes, or alternatively just viewed on the screen. The printed output will look like this:



Member Status Report

Glenn Howard **Executive Presenter** **01704 245992**	10 Birchtree Avenue Newtown PR3 4RT
Stuart McDowell **Executive Presenter** **01704 247546**	45 Greenwood Place Newtown PR5 7RY
Elaine Goy **Executive Presenter** **01704 247521**	47 Greenwood Place Newtown PR5 7RY

Total Members: 17 Page 3

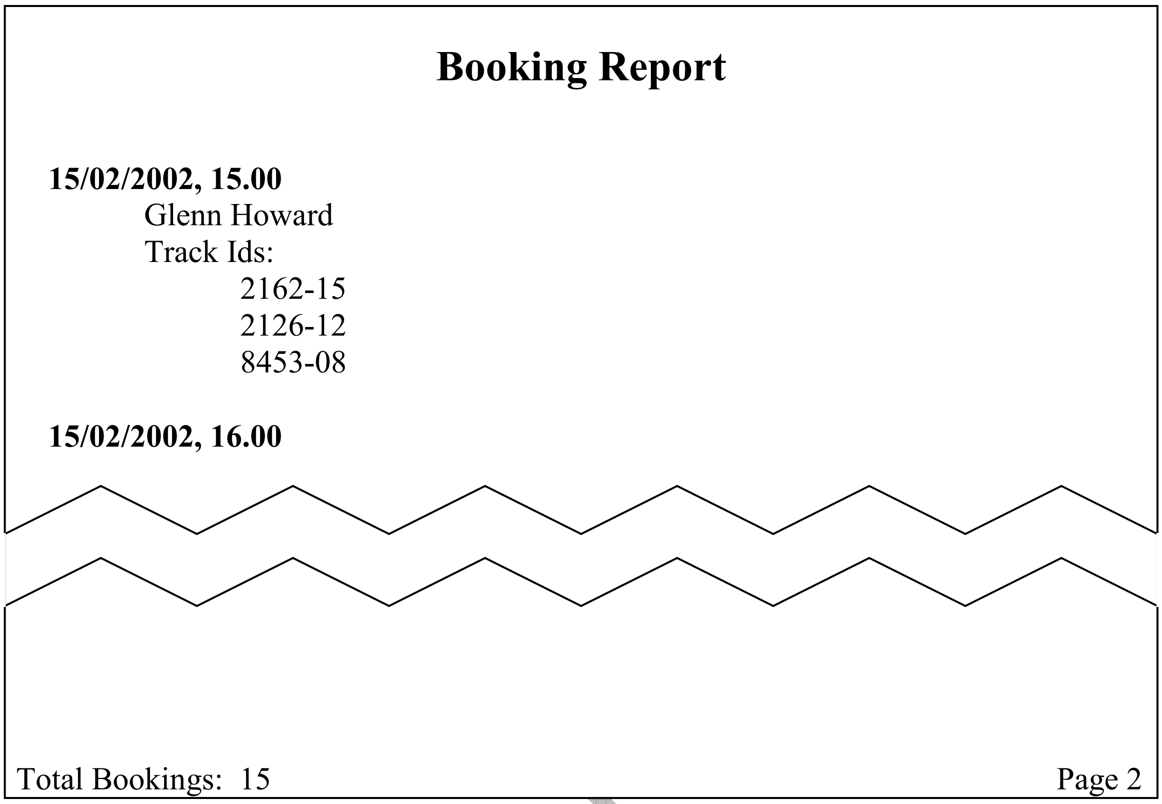


Track List

My Way
Frank Sinatra
Genre: Other
Length: 4.55
ID: 5146-01

When I Was Seventeen
Frank Sinatra
Genre: Other

Total Tracks: 1532 Page 15



Booking Report

15/02/2002, 15.00

Glenn Howard

Track Ids:

2162-15

2126-12

8453-08

15/02/2002, 16.00

Total Bookings: 15 Page 2

Security and Integrity of Data

The only data which needs to be secured is that of the members, since this is particularly sensitive. Any problems with the integrity of the other data will be detected and easily corrected during system usage. The Member information, however, only needs to be accessed by executives, and so will be password protected. The password protection will be built into VB. This will obviously not make the system impossible to get in to, but it should make it a little more difficult for opportunists. The password will be stored as a separate database table.

System Security

System security will be rather important in this system, since there will be a lot of data stored in it, and nobody would be pleased if the data became corrupt and had to be re-entered. Therefore, regular backups should be made. Since data is not often added to the system, weekly backups should be sufficient. The database files are the ones which should be backed up. The database can be backed up with the rest of the system as is the present system, and the backup is stored at an external location (the secretary's house). This should make the data secure enough. Password protection, as discussed in Security and Integrity of Data (p20), should help to prevent accidental deletion or corruption of the more sensitive data in the system.

Test Strategy

My Testing strategy will make sure that all parts of the program are able to handle typical and extreme data, whilst rejecting erroneous data. Particular parts of the program which will be subject to testing will be those with validation processes, such as the Music IDs and track lengths in the Music Database. Of course, other parts of the program will also need to be subject to testing in order to ensure that the system created works fully.

For further details on the testing actually carried out during the project, see the section "System Testing", which begins on page 26.

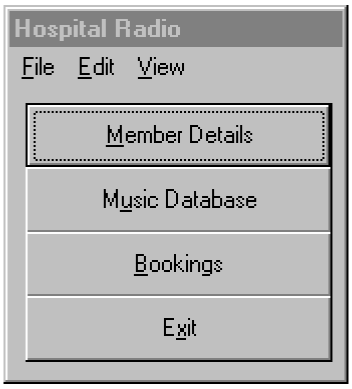
Technical Solution

Annotated Program Listings

These can be found in Appendix A.

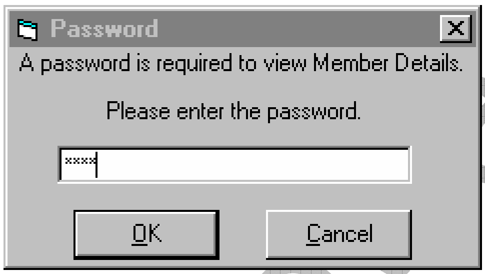
Screen Designs

During the implementation, I changed some of the screen designs slightly, as I felt that the changes made navigation and use of the program. The screen designs actually used are shown here:



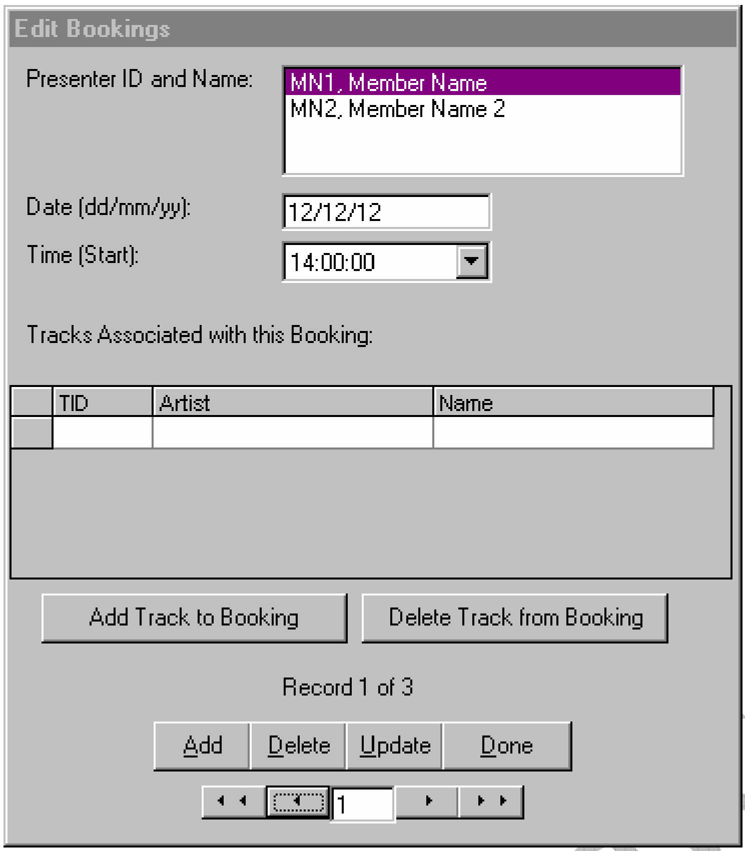
The Main Menu:

This design is very similar to the planned design. The main change is in the size of the window: All window sizes have been reduced so that users can easily refer to other programs if necessary whilst running this program on the screen. For example, if a request was received by email, the track could easily be searched for because this window could stay on the screen along with the email application window.


Password
A password is required to view Member Details.
Please enter the password.
****
OK
Cancel

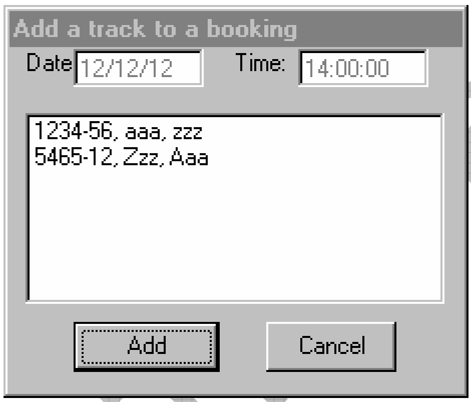
The Password Entry Window:

The password entry window design is consistent throughout the program, so that the user can very quickly become familiar with all password entry screen designs. The password is not displayed during entry to provide added security.

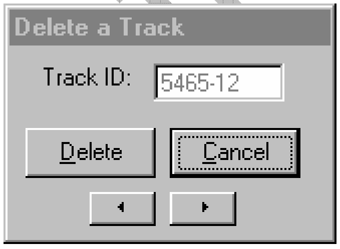


The Bookings Window:

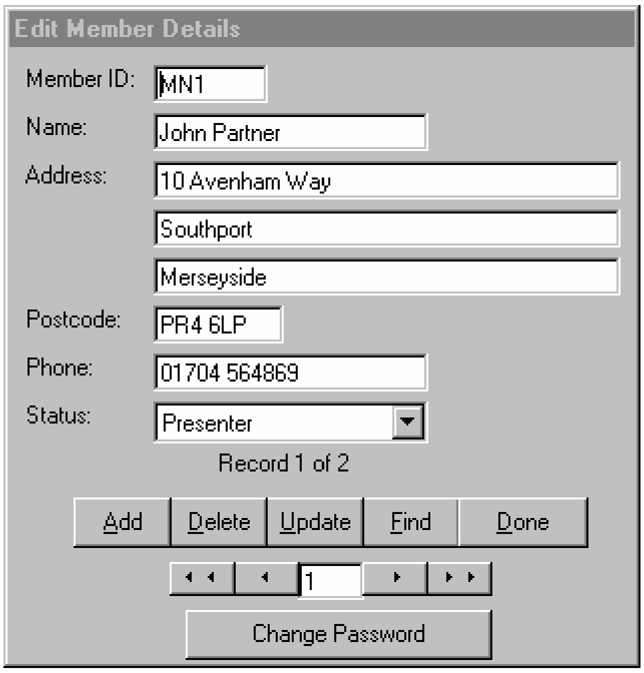
This has been modified for simpler navigation (with the facility to enter any record number and jump straight to it). Also, the Booking ID, which is meaningless to the user, has been hidden for increased simplicity. Another improvement, implemented throughout the program, is the changing of the non-standard "Main Menu" button to a windows standard "Done" button.



This window is for adding tracks to bookings, and is fairly self-explanatory.



The Delete a Track window is for removing tracks from the selected booking. Ideally, it would have been designed more like the Add Track window, but the constraints of time made this impossible. In its current state, the window serves its purpose.

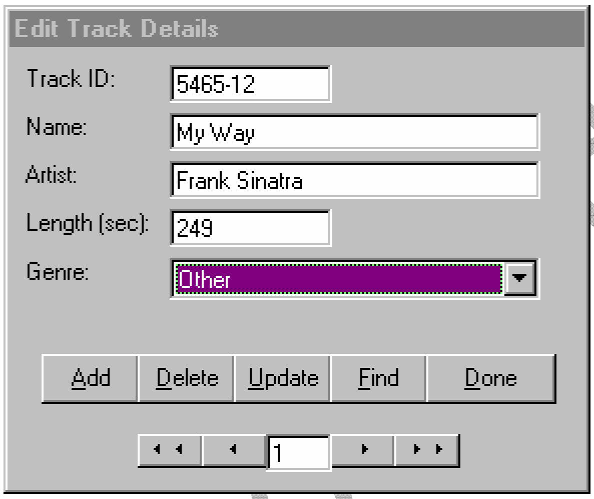


Member ID:	MN1
Name:	John Partner
Address:	10 Avenham Way
	Southport
	Merseyside
Postcode:	PR4 6LP
Phone:	01704 564869
Status:	Presenter

Record 1 of 2

The Edit Member Details screen:

This has been modified only slightly, with three lines now present for entry of the full address, and a Change Password button added.



Track ID:	5465-12
Name:	My Way
Artist:	Frank Sinatra
Length (sec):	249
Genre:	Other

The Edit Track Details Screen:

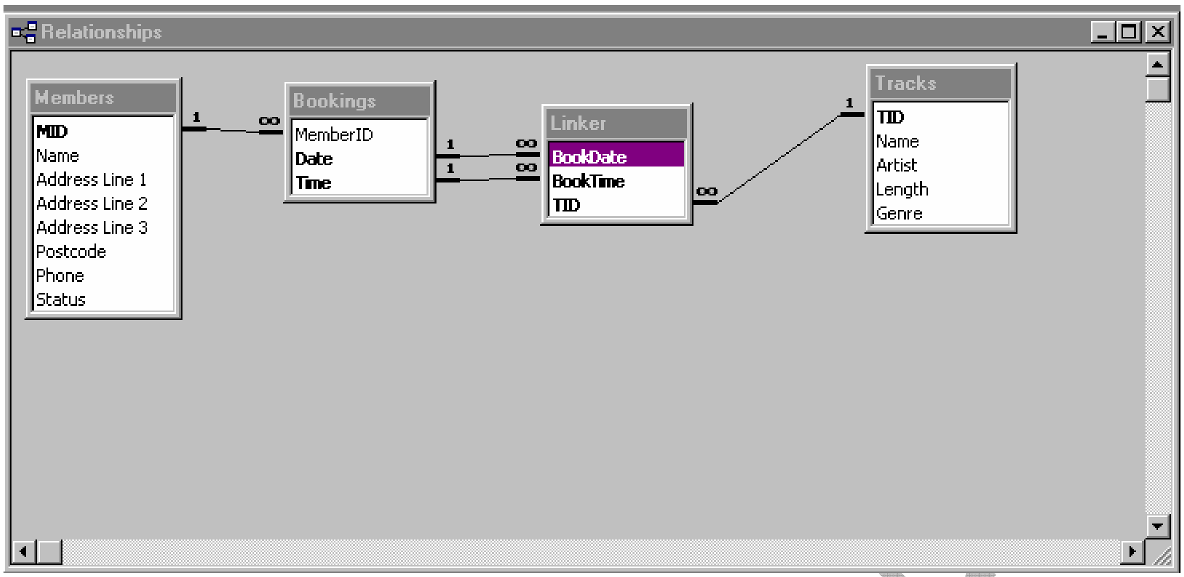
Again, the design of this screen has changed minimally. The new navigation system, implemented throughout the project, has been added on this screen.

Report designs have also changed minimally. Hard copies of test reports can be found in Appendix B.

Database Structure

The Access tables have remained the same as in the design stage, but with one addition: The Password Table. This has been set up as a single field table simply storing the password.

Finally, the database relationships have remained the same as in the initial design, as a screen dump from Microsoft Access shows:



How the technical solution was achieved

I constructed this technical solution over a number of months using Microsoft Visual Basic 6. I used a top-down design approach so that I could work steadily through the project, and make sure that no part was left out. I tested the program during the coding stage, and have documented all testing in “System Testing”, which begins on page 26.

System Testing

Throughout this section, please refer to the Screen Dumps starting on page 34.

Typical Testing

Test Number	Test & Screen Title	Reason For Test	Test Data	Expected Result	Actual Result
1	Main Menu	Check correct screen is shown	Start the Program	Main Menu should be displayed	As expected: See Screen Dump 1, page 34.
2		Check correct screen is shown	Click Member Details	Password Entry Screen should be displayed with password entry field blank	As expected: See Screen Dump 2, page 34.
3		Check correct screen is shown	Click Music Database	Music Database Screen should be displayed with first record showing	As expected: See Screen Dump 3, page 34.
4		Check correct screen is shown	Click Bookings	Bookings Screen should be displayed, with first record showing	As expected: See Screen Dump 4, page 35.
5		Check that program exits	Click Exit	Program should close	As expected.
6	Main Menu:	Check that	Click File, Exit	Program should close	As expected.

	Drop-down Menu	program exits			
7		Check that correct screen is shown	Click Edit, Bookings	Bookings Screen should be displayed, with first record showing	As expected: See Screen Dump 4, page 35.
8		Check that correct screen is shown	Click Edit, Member Details	Password Entry Screen should be displayed with password entry field blank	As expected: See Screen Dump 2, page 34.
9		Check correct screen is shown	Click Edit, Music Database	Music Database Screen should be displayed with first record showing	As expected: See Screen Dump 3, page 34.
10		Check correct report is shown	Click View, Bookings	Booking Report should be displayed	Failed: Nothing displayed. See Corrections Resulting from Failed Tests, page 33.
11		Check correct screen is shown	Click View, Member Details	Password Entry Screen should be displayed with password entry field blank	As expected: See Screen Dump 2, page 34.
12		Check correct report is shown	Click View, Music Database	Complete Track Listing report should be displayed	As expected: See Screen Dump 5, page 35.
13	Edit Bookings Screen	Check that record is added correctly	Member ID: MN1 Date: 04/05/02 Time: 14:00	Access Table should be successfully updated	As expected: See Screen Dump 6, page 36
14		Check that track is	Member ID: MN1	Track should be added to the	As expected: See Screen Dump 7,

		correctly added to booking	Date: 04/05/02 Time: 14:00 Track ID: 5465-12	"Linker" table in Access	on page 36.
15		Check that track is correctly deleted from booking	Member ID: MN1 Date: 04/05/02 Time: 14:00 Track ID: 5465-12	Track should be deleted from the "Linker" table in Access	As expected: See Screen Dump 8, on page 37.
16		Check that navigation buttons work	All navigation buttons clicked	The navigation buttons should move the user forwards and backwards through the database	As expected.
17		Check that navigation text box works	Record number 3	Record number 3 should be displayed	As expected.
18		Check that Done button works	Clicked Done button	The user should be returned to the main menu.	As expected: See Screen Dump 1, page 34.
19	Edit Member Details Screen	Check that record is added correctly	Member ID: MN3 Name: Nicola Pacey Address: 23 Hanover Street, Birkdale, Newtown Post code: PR9 9SP Phone: 01704 293854 Status: Support	The details should be added to the Access database	As expected: See Screen Dump 9, page 37.
20		Check that record is deleted correctly	Member ID: MN3 Name: Nicola Pacey Address: 23 Hanover Street, Birkdale, Newtown	The details should be deleted from the Access database	As expected: See Screen Dump 10, page 37.

			Post code: PR9 9SP Phone: 01704 293854 Status: Support		
21		Check that record is edited correctly	Record added in Test 19 will have Member ID changed to MN9 and Name changed to Nicki Pacey	The details should be updated in the Access database table	As expected: See Screen Dump 11, page 38.
22		Check that search facility works	Search for Glenn, case sensitive.	"Glenn" should be found and highlighted in the Member Details window	As expected: See Screen Dump 12, page 38.
23		Check that search facility works	Search for GLENN, case insensitive	"Glenn" should be found and highlighted in the Member Details window	As expected: See Screen Dump 12, page 38.
24		Check password update	Change password from "12greenbottles" to "3littlepigs"	Update should be reflected in Access database	Failed: Password not updated. See Corrections Resulting from Failed Tests, page 33.
25		Check that navigation buttons work	All navigation buttons clicked	The navigation buttons should move the user forwards and backwards through the database	As expected.
26		Check that navigation text box works	Record number 3	Record number 3 should be displayed	As expected.
27	Check Music Database Screen	Check that a record can be added correctly	TrackID: 8765-01 Name: Movies Artist: Alien Ant Farm	Addition should be reflected in Access database	As expected: See Screen Dump 14, page 39.

			Length: 224 seconds Genre: Other		
28		Check that a record can be deleted correctly	TrackID: 8765-01 Name: Movies Artist: Alien Ant Farm Length: 224 seconds Genre: Other	Deletion should be reflected in Access Database	As expected: See Screen Dump 15, page 39.
29		Check that a record can be edited correctly	The track added in test 25 will be edited as follows: Genre: Pop Length: 212 seconds	Update should be reflected in Access Database	As expected: See Screen Dump 16, page 40.
30		Check that search facility works	Search for "Frank", case sensitive	"Frank" should be highlighted in the Music Database window	As expected: See Screen Dump 17, page 40.
31		Check that search facility works	Search for "FRANK", case sensitive	"Frank" should be highlighted in the Music Database window	As expected: See Screen Dump 17, page 40.
32		Check that Done button works	Click done button	User should be returned to the main menu	Failed: Nothing happened. See Corrections Resulting from Failed Tests, page 33.
33		Check that navigation buttons work	All navigation buttons clicked	The navigation buttons should move the user forwards and backwards through the database	As expected.
34		Check that navigation text box works	Record number 3	Record number 3 should be displayed	As expected.
35		Check that reports print correctly.	Reports opened via main menu	Reports should print as designed.	As expected: See report printouts in Appendix B.

Extreme and Erroneous Testing

Test Number	Test & Screen Title	Reason For Test	Test Data	Expected Result	Actual Result
36	Password Window	Check that incorrect password is rejected	"Bananas" (Erroneous Data)	Error message should be displayed, user should be returned to Main Menu	As expected: See Screen Dump 18, page 41.
37		Check that correct password with wrong case is rejected	"3LITTLEPIGS" (Correct password, incorrect case)	Error message should be displayed, user should be returned to Main Menu	As expected: See Screen Dump 18, page 41.
38	Member Details Find Facility	Check that data not found is handled suitably	"Stephanie" (A name which is not in the database)	An error message should be displayed	As expected: See Screen Dump 19, page 41.
39	Music Database Find Facility	Check that data not found is handled suitably	"Sunshine" (A track which is not in the database)	An error message should be displayed	As expected: See Screen Dump 19, page 41.
40	Edit Member Details Window	Check that system reacts appropriately when	"34" (The Member Details database does not	An error message should be displayed	As expected: See Screen Dump 20, page 41.

		invalid record number entered	currently have 34 tracks)		
41	Music Database Window	Check that system reacts appropriately when invalid record number entered	"54" (The Member Details database does not currently have 34 tracks)	An error message should be displayed	As expected: See Screen Dump 20, page 41.
42		Check that system reacts appropriately when an invalid track length is entered	"0" (A track length cannot be less than 1 second long)	An error message should be displayed	As expected: See Screen Dump 21, page 41.
43		Check that system reacts appropriately when an invalid track length is entered	"4000" (A track length cannot be more than 1 hour – 3600 seconds - long)	An error message should be displayed	As expected: See Screen Dump 21, page 41.
44	Edit Bookings Window	Check that system reacts appropriately when invalid record number entered	"54" (The Member Details database does not currently have 34 tracks)	An error message should be displayed	As expected: See Screen Dump 20, page 41.
45		Check that the system responds appropriately when invalid date entered	"31/02/02" (invalid date)	The system should attempt to rearrange the date into a correct format (e.g. 04/31/99 becomes 31/04/99) and display and error message if this is not possible	As expected: Date rearranged to 02/02/31
46		Check that the	"31/31/31"	The system should attempt to	As expected: Error message

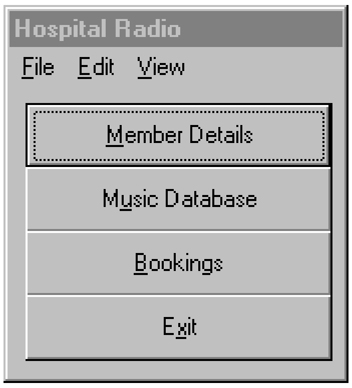
		system responds appropriately when invalid date entered	(invalid date)	rearrange the date into a correct format (e.g. 04/31/99 becomes 31/04/99) and display and error message if this is not possible	displayed.

Corrections Resulting from Failed Tests

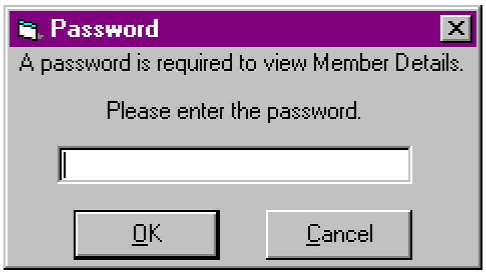
Test Number	Remedial Action Taken	Retest
10	Correct coding added (drBookings.Show)	Result as Expected
24	Coding corrected	Result as Expected: See Screen Dump 13, page 39.
32	Correct coding added (frmMain.Show, Me.Hide)	Result as Expected: See Screen Dump 1, page 34.

Testing: Screen Dumps

Screen Dump 1: The Main Menu



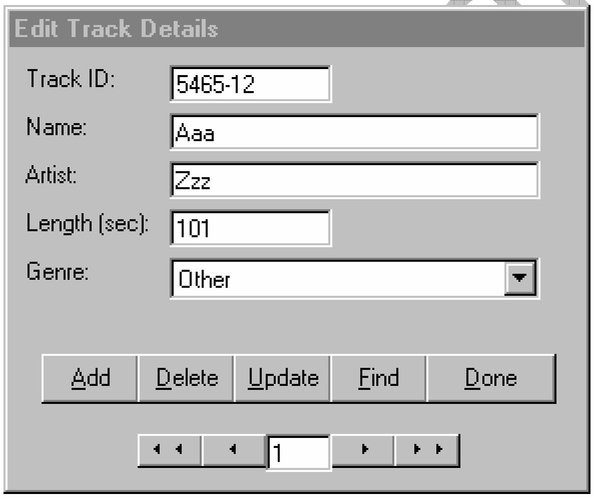
Screen Dump 2: The Password Entry Window



A password is required to view Member Details.

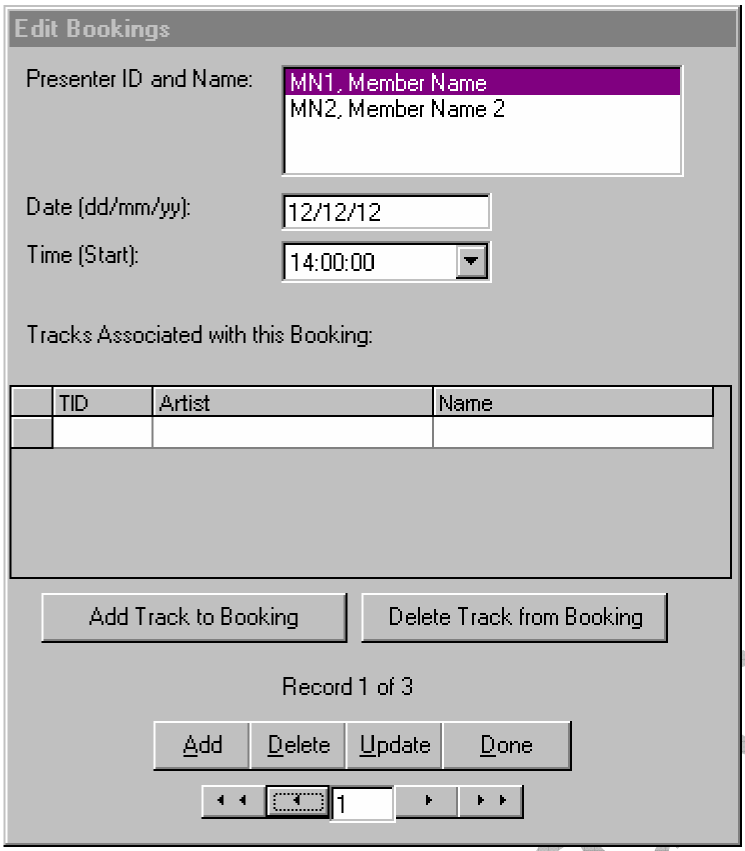
Please enter the password.

Screen Dump 3: The Edit Track Details Screen



Track ID:	5465-12
Name:	Aaa
Artist:	Zzz
Length (sec):	101
Genre:	Other

Screen Dump 4: The Edit Bookings Screen



Presenter ID and Name:
MN1, Member Name
MN2, Member Name 2

Date (dd/mm/yy): 12/12/12

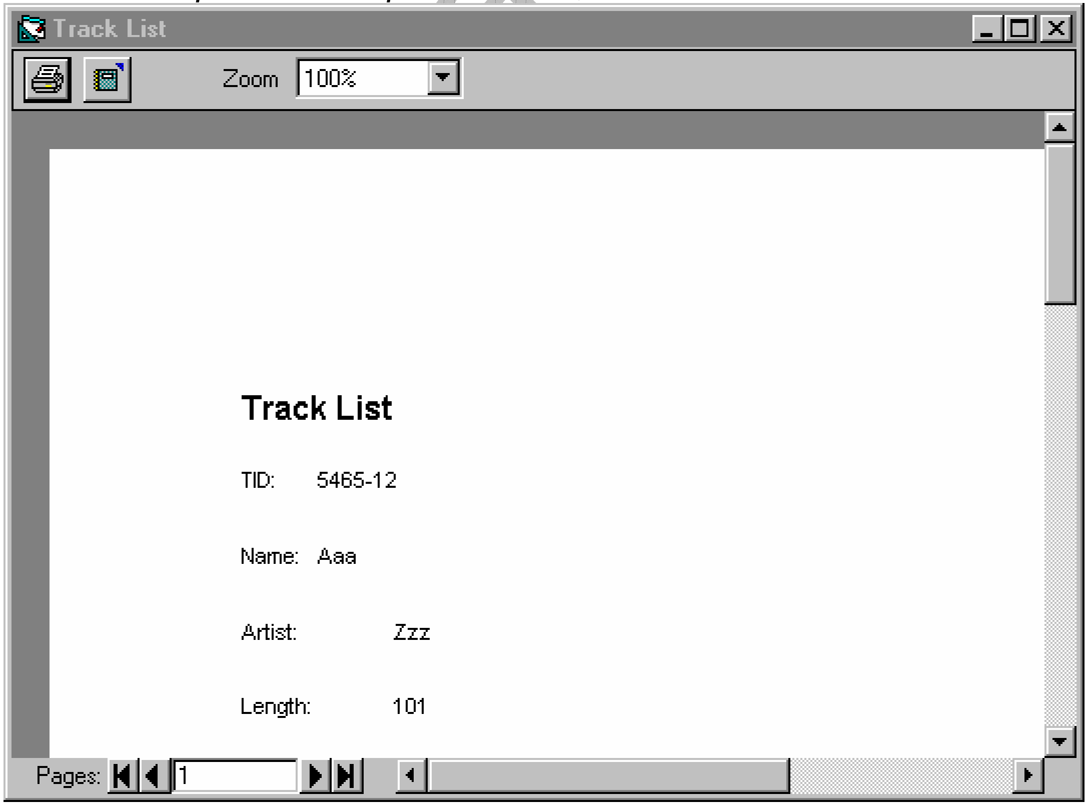
Time (Start): 14:00:00

Tracks Associated with this Booking:

TID	Artist	Name

Record 1 of 3

Screen Dump 5: The Reports Window



Track List

TID: 5465-12

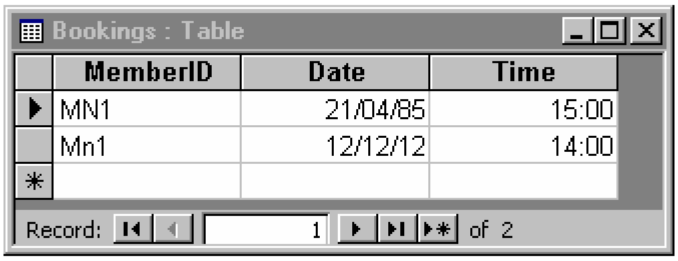
Name: Aaa

Artist: Zzz

Length: 101

Screen Dump 6: Adding a Member (Access Tables)

Before addition of record:

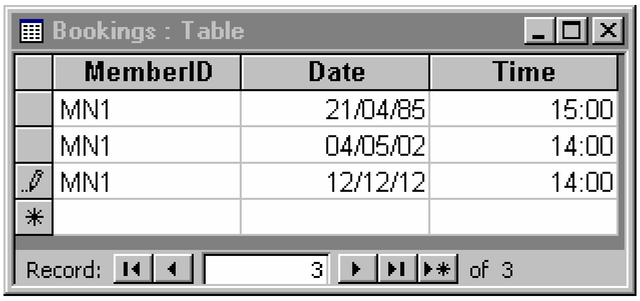


Bookings : Table

	MemberID	Date	Time
▶	MN1	21/04/85	15:00
	Mn1	12/12/12	14:00
*			

Record: 1 of 2

After addition of record:



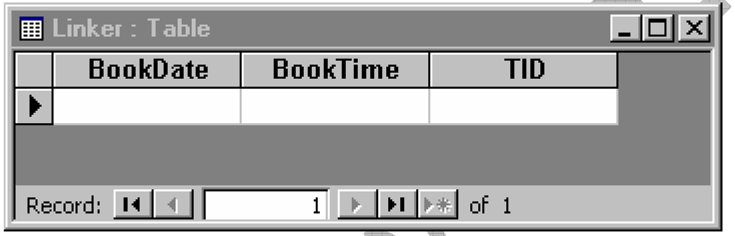
Bookings : Table

	MemberID	Date	Time
	MN1	21/04/85	15:00
	MN1	04/05/02	14:00
✎	MN1	12/12/12	14:00
*			

Record: 3 of 3

Screen Dump 7: Adding a Track to a Record (Access Tables)

Before addition of track:

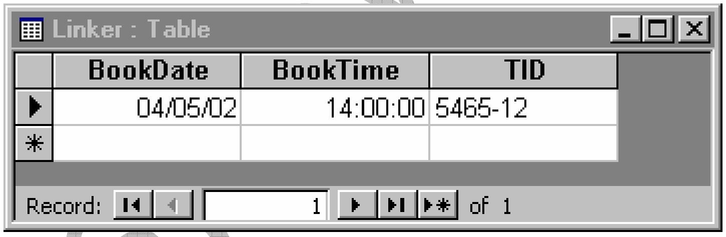


Linker : Table

	BookDate	BookTime	TID
▶			

Record: 1 of 1

After addition of track:

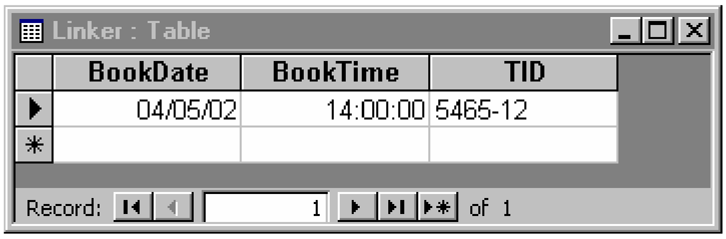


Linker : Table

	BookDate	BookTime	TID
▶	04/05/02	14:00:00	5465-12
*			

Record: 1 of 1

Screen Dump 8: Deleting a Track (Access Tables)
Before track is deleted:

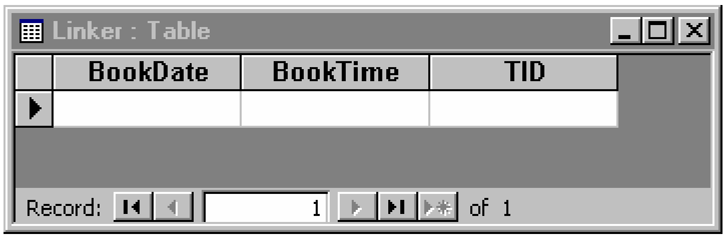


Linker : Table

	BookDate	BookTime	TID
▶	04/05/02	14:00:00	5465-12
*			

Record: 1 of 1

After track is deleted:

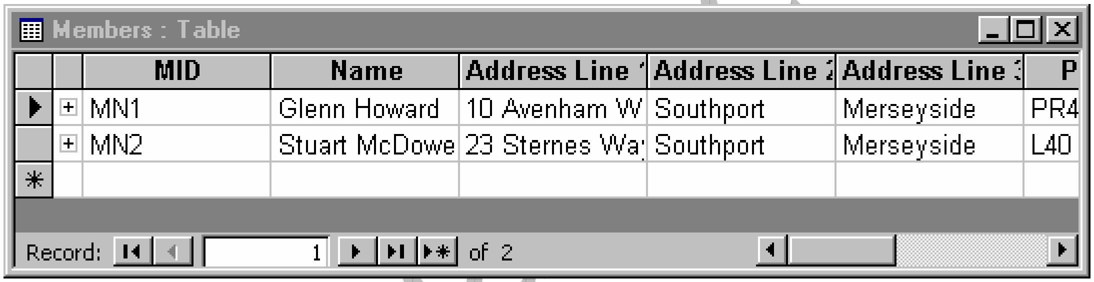


Linker : Table

	BookDate	BookTime	TID
▶			

Record: 1 of 1

Screen Dump 9: Adding a Member (Access Tables)
Before member is added:

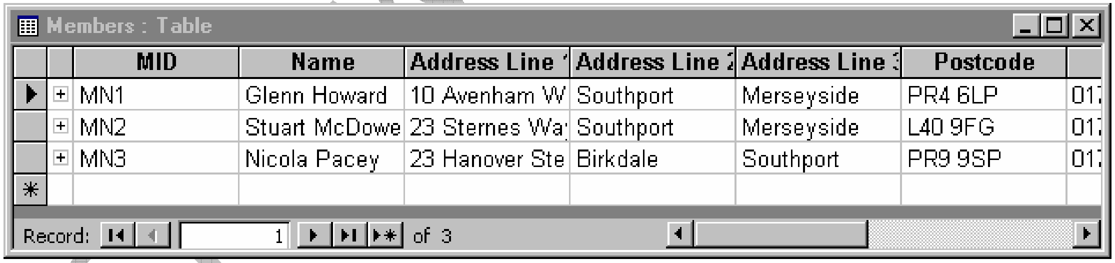


Members : Table

		MID	Name	Address Line	Address Line	Address Line	P
▶	+	MN1	Glenn Howard	10 Avenham W	Southport	Merseyside	PR4
	+	MN2	Stuart McDowe	23 Sternes Wa	Southport	Merseyside	L40
*							

Record: 1 of 2

After member is added:

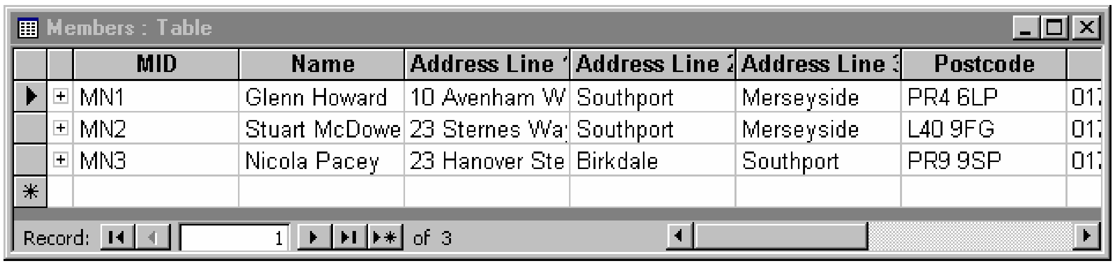


Members : Table

		MID	Name	Address Line	Address Line	Address Line	Postcode	
▶	+	MN1	Glenn Howard	10 Avenham W	Southport	Merseyside	PR4 6LP	01
	+	MN2	Stuart McDowe	23 Sternes Wa	Southport	Merseyside	L40 9FG	01
	+	MN3	Nicola Pacey	23 Hanover Ste	Birkdale	Southport	PR9 9SP	01
*								

Record: 1 of 3

Screen Dump 10: Deleting a Member (Access Tables)
Before member is deleted:

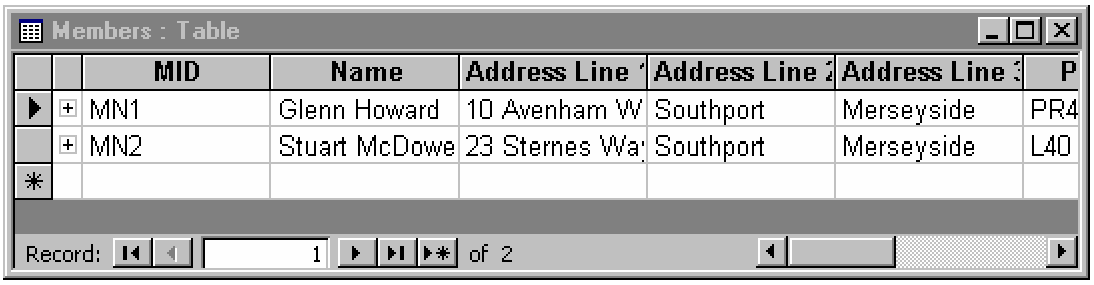


Members : Table

		MID	Name	Address Line	Address Line	Address Line	Postcode	
▶	+	MN1	Glenn Howard	10 Avenham W	Southport	Merseyside	PR4 6LP	01
	+	MN2	Stuart McDowe	23 Sternes Wa	Southport	Merseyside	L40 9FG	01
	+	MN3	Nicola Pacey	23 Hanover Ste	Birkdale	Southport	PR9 9SP	01
*								

Record: 1 of 3

After member is deleted:



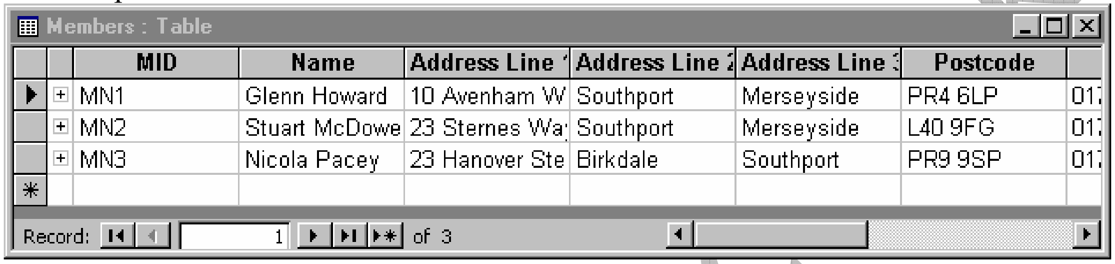
Members : Table

	MID	Name	Address Line 1	Address Line 2	Address Line 3	P
▶	MN1	Glenn Howard	10 Avenham W	Southport	Merseyside	PR4
	MN2	Stuart McDowe	23 Sternes Wa	Southport	Merseyside	L40
*						

Record: 1 of 2

Screen Dump 11: Editing a Member (Access Tables)

Before update:

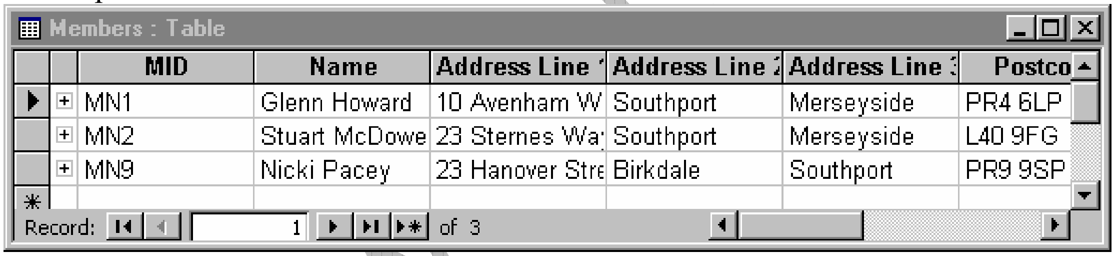


Members : Table

	MID	Name	Address Line 1	Address Line 2	Address Line 3	Postcode	
▶	MN1	Glenn Howard	10 Avenham W	Southport	Merseyside	PR4 6LP	01
	MN2	Stuart McDowe	23 Sternes Wa	Southport	Merseyside	L40 9FG	01
	MN3	Nicola Pacey	23 Hanover Ste	Birkdale	Southport	PR9 9SP	01
*							

Record: 1 of 3

After update:

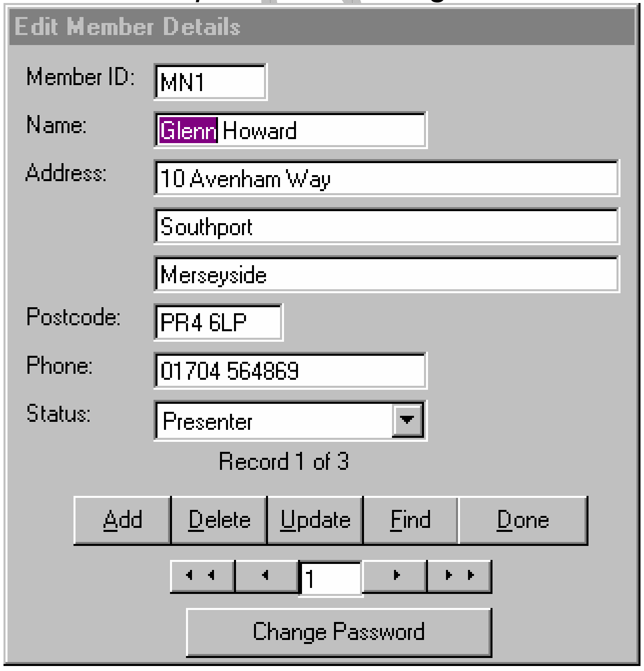


Members : Table

	MID	Name	Address Line 1	Address Line 2	Address Line 3	Postco
▶	MN1	Glenn Howard	10 Avenham W	Southport	Merseyside	PR4 6LP
	MN2	Stuart McDowe	23 Sternes Wa	Southport	Merseyside	L40 9FG
	MN9	Nicki Pacey	23 Hanover Stre	Birkdale	Southport	PR9 9SP
*						

Record: 1 of 3

Screen Dump 12: Searching for Member Details



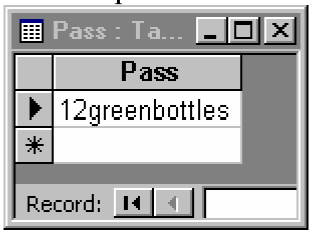
Edit Member Details

Member ID:	MN1
Name:	Glenn Howard
Address:	10 Avenham Way
	Southport
	Merseyside
Postcode:	PR4 6LP
Phone:	01704 564869
Status:	Presenter

Record 1 of 3

Screen Dump 13: Changing the Password (Access Tables)

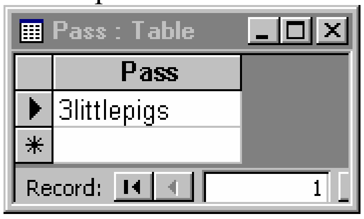
Before update:


Pass : Ta...

Pass
12greenbottles

Record:

After update:

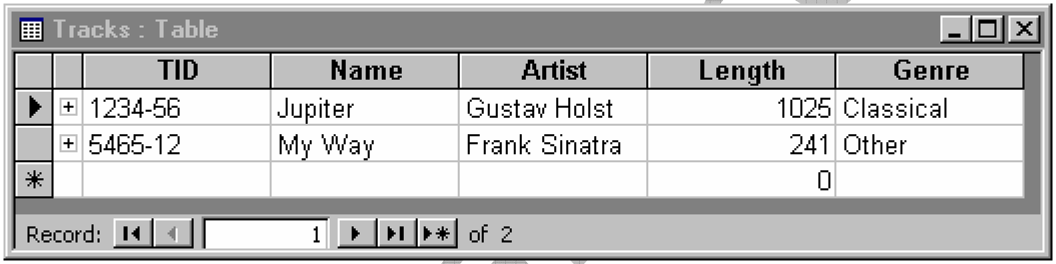

Pass : Table

Pass
3littlepigs

Record: 1

Screen Dump 14: Adding a Track (Access Tables)

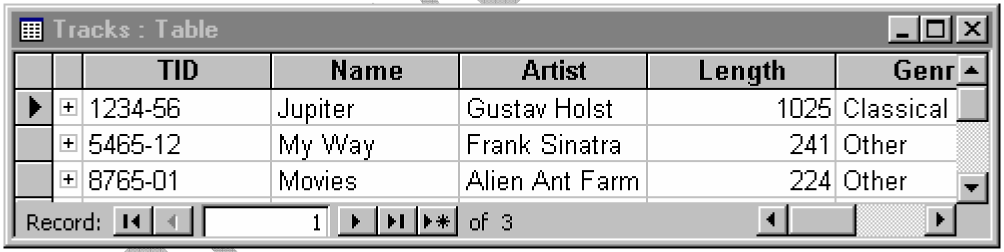
Before addition:


Tracks : Table

TID	Name	Artist	Length	Genre
1234-56	Jupiter	Gustav Holst	1025	Classical
5465-12	My Way	Frank Sinatra	241	Other
			0	

Record: 1 of 2

After addition:

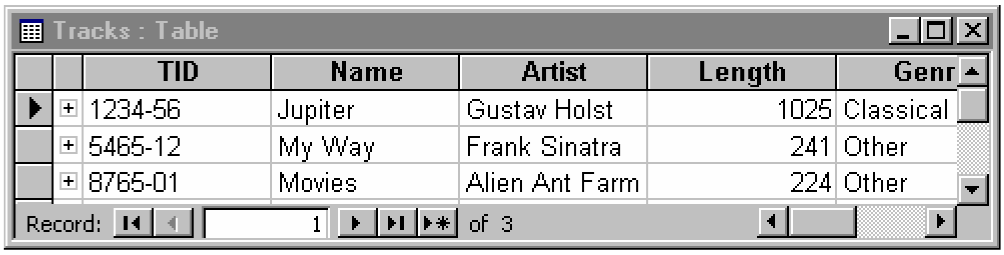

Tracks : Table

TID	Name	Artist	Length	Genr
1234-56	Jupiter	Gustav Holst	1025	Classical
5465-12	My Way	Frank Sinatra	241	Other
8765-01	Movies	Alien Ant Farm	224	Other

Record: 1 of 3

Screen Dump 15: Deleting a Track (Access Tables)

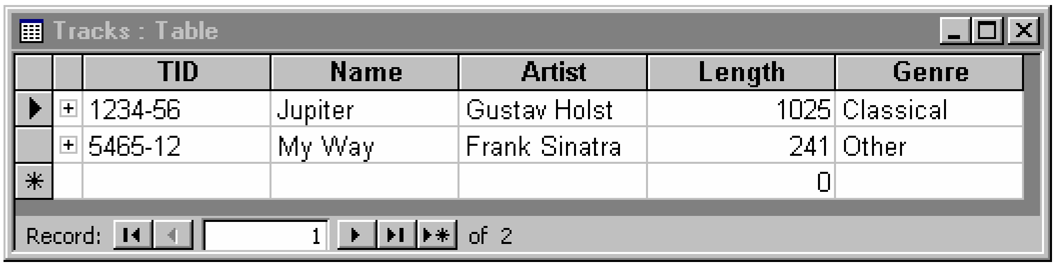
Before deletion:


Tracks : Table

TID	Name	Artist	Length	Genr
1234-56	Jupiter	Gustav Holst	1025	Classical
5465-12	My Way	Frank Sinatra	241	Other
8765-01	Movies	Alien Ant Farm	224	Other

Record: 1 of 3

After deletion:



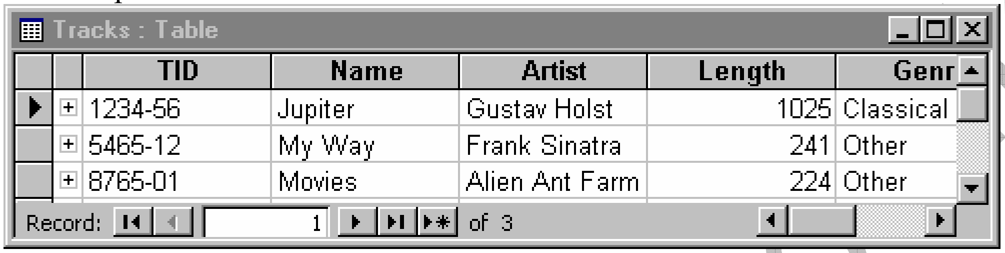
Tracks : Table

TID	Name	Artist	Length	Genre
1234-56	Jupiter	Gustav Holst	1025	Classical
5465-12	My Way	Frank Sinatra	241	Other
			0	

Record: 1 of 2

Screen Dump 16: Editing a Track (Access Tables)

Before update:

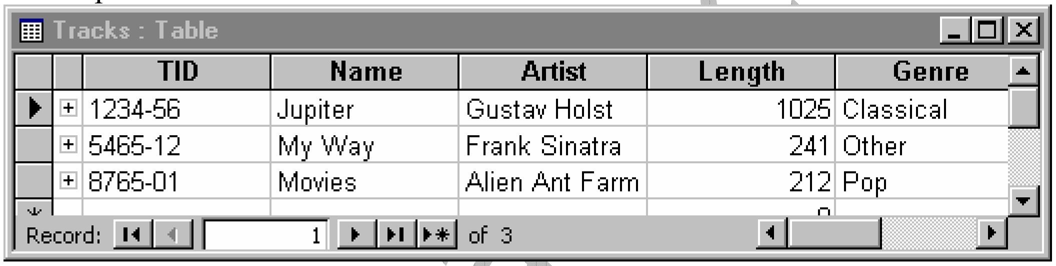


Tracks : Table

TID	Name	Artist	Length	Genr
1234-56	Jupiter	Gustav Holst	1025	Classical
5465-12	My Way	Frank Sinatra	241	Other
8765-01	Movies	Alien Ant Farm	224	Other

Record: 1 of 3

After update:

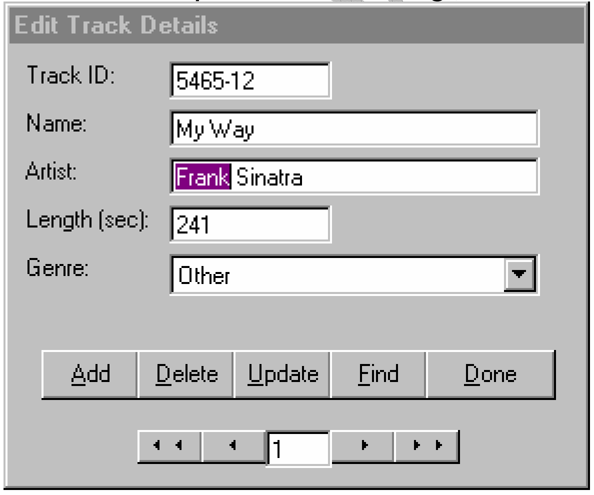


Tracks : Table

TID	Name	Artist	Length	Genre
1234-56	Jupiter	Gustav Holst	1025	Classical
5465-12	My Way	Frank Sinatra	241	Other
8765-01	Movies	Alien Ant Farm	212	Pop

Record: 1 of 3

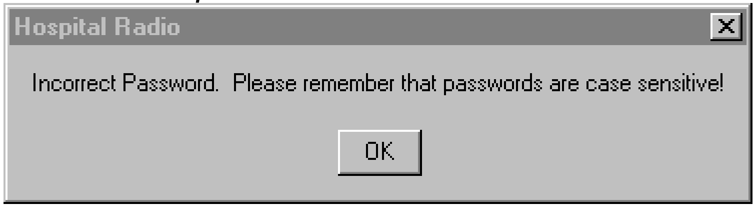
Screen Dump 17: Searching for a Track



Edit Track Details

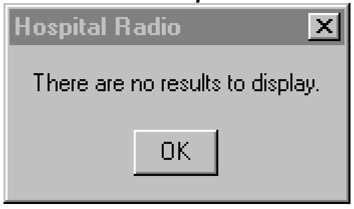
Track ID:	5465-12
Name:	My Way
Artist:	Frank Sinatra
Length (sec):	241
Genre:	Other

Screen Dump 18: Incorrect Password Error Message



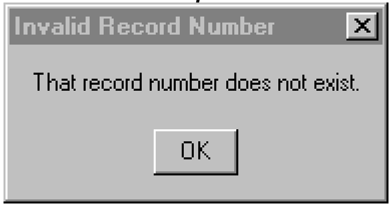
Incorrect Password. Please remember that passwords are case sensitive!

Screen Dump 19: No More Results Error Message



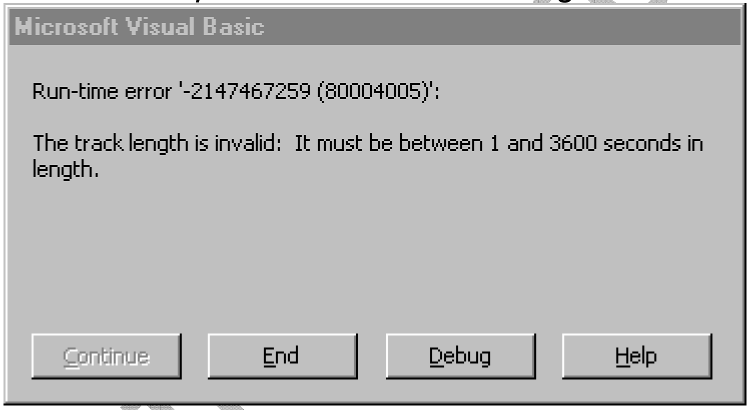
There are no results to display.

Screen Dump 20: Invalid Record Number Error Message



That record number does not exist.

Screen Dump 21: Invalid Track Length Error Message



Run-time error '-2147467259 (80004005)':

The track length is invalid: It must be between 1 and 3600 seconds in length.

System Maintenance

Access Features Used

In Microsoft Access, I used only tables, relationships, and validation procedures to ensure that invalid data cannot be added to the database.

Visual Basic Features Used

In Visual Basic, I used several features:

- Form design
- The Reporting Environment
- ADODCs to link the program to the Access Database
- Coding

Detailed Algorithm Design

The algorithms here are listed in alphabetical order of title. See the full contents (beginning on page 1) for page references. Please note that the algorithms in this section are line numbered for easy reference: such line numbering is not included in the program code. The complete program code is in Appendix A, and a list of the variables in these algorithms and elsewhere in the program is contained in the section "Procedures and Variables", beginning on page 45.

I recognise that my algorithms may not be as economical with coding or as quick to run as they may be, but speed is relatively unimportant in this program, and as long as the algorithms actually work then I am happy to use them.

Disabling and hiding irrelevant command buttons and labels

This algorithm uses a For / Next loop to make all irrelevant command buttons and labels invisible. It is used several times in the main program, and uses the variable "num":

```
1    For num = 0 to 3
        cmdNav(num).Visible = False
        If num <> 3 Then cmdAction(num).Visible = False
     Next
5    cmdAddTrack.Enabled = False
     cmdDeleteTrack.Enabled = False
     txtRec.Visible = False
```

This algorithm is placed behind all Add buttons to prevent the user from performing an operation which would interrupt the file update. The `If num <> 3` on line 3 is necessary because there is no cmdAction(3).

Find a Member: Case Insensitive

This algorithm is used to find a member when the Case Sensitive option is disabled.

```
1    If Trim(txtName.Text) <> "" Then
        namePart = UCase(Trim(txtName.Text))
        With frmMemberEdit.adoMembers.Recordset
           Do While Not .EOF
```

```
5             For num = 0 to 3
                 Position = InStr(Ucase(frmMemberEdit.Field(num).
                                                         Text), namePart)
                 If Position <> 0 Then
                    frmMemberEdit.Field(num).SelStart = Position - 1
                    frmMemberEdit.Field(num).SelLength = Len(namePart)
10                  frmMemberEdit.Field(num).SetFocus
                  Exit Sub
                 End If
              Next
              .MoveNext
15         Loop
           MsgBox "There are no results to display.", vbOKOnly,
                                                         "Hospital Radio"
           .MoveFirst
        End With
     End If
```

Line 1 ensures that the search field actually contains some text: If not, then the `If` condition is not fulfilled and the algorithm is not implemented. The `UCase` function, used on lines two and six, ensures that the text found and the text in the search field is all converted to upper case, which makes this algorithm case insensitive.

Find a member: Case Sensitive
This coding is very similar to that for the case insensitive search:

```
1    If Trim(txtName.Text) <> "" Then
        namePart = Trim(txtName.Text)
        With frmMemberEdit.adoMembers.Recordset
           Do While Not .EOF
5             For num = 0 to 3
                 Position = InStr(frmMemberEdit.Field(num). Text,
                                                                namePart)
                 If Position <> 0 Then
                    frmMemberEdit.Field(num).SelStart = Position - 1
                    frmMemberEdit.Field(num).SelLength = Len(namePart)
10                  frmMemberEdit.Field(num).SetFocus
                  Exit Sub
                 End If
              Next
              .MoveNext
15         Loop
           MsgBox "There are no results to display.", vbOKOnly,
                                                         "Hosiptal Radio"
           .MoveFirst
        End With
     End If
```

The notes for the case sensitive version still apply here, with the exception of those concerning the `UCase` function. Here, this is omitted, so that the text in the Search Field and the text with which it is compared retains its original case, and so the algorithm is case sensitive.

Re-enabling relevant command buttons and labels
This algorithm is very similar to the algorithm to disable and hide irrelevant command buttons and labels.

```
1    For num = 0 to 3
               cmdNav(num).Visible = True
               If num <> 3 Then cmdAction(num).Visible = True
            Next
5    cmdAddTrack.Enabled = True
            cmdDeleteTrack.Enabled = True
            txtRec.Visible = False
```

Conversely to the algorithm to disable and hide irrelevant command buttons and labels, this algorithm is placed behind all Add confirmation and Cancel Addition buttons in order to restore functionality once the update has been completed.

Searching for Tracks

The search method here is not dissimilar to that used to find members:

```
1    If Trim(txtName.Text) <> "" Then
        If frmMusicFind.chkExact = 0 Then
           namePart = UCase(Trim(txtName.Text))
        Else
5          NamePart = Trim(txtName.Text)
        End If

        With frmTrackEdit.adoTracks.Recordset
           Do While Not .EOF
              For num = 0 to 5
10               If frmMusicFind.chkExact = 0 Then
                    Position = InStr(UCase(frmTrackEdit.Field(num).
                                                  Text), namePart)
                 Else
                    Position = InStr(frmTrackEdit.Field(num).Text,
                                                         namePart)
                 End If

15               If Position <> 0 Then
                    frmTrackEdit.Field(num).SelStart = Position - 1
                    frmTrackEdit.Field(num).SelLength = Len(namePart)
                    frmTrackEdit.Field(num).SetFocus
                   Exit Sub
20               End If
              Next
              .MoveNext
           Loop
           MsgBox "There are no results to display.", vbOKOnly,
                                                  "Hospital Radio"
25         .MoveFirst
        End With
     End If
```

Whilst, at first glance, this coding seems longer than that for the Member searches, it is actually far more economical since this search will work for both case sensitive and case insensitive searches. I realised that this was a possibility after having implemented the member searches, but chose to leave those as two separate algorithms and combine this one, so that development in my code writing skills can be seen. The way in which this dual-purpose code is written is that `If` statements are included on lines 2-6 and 10-14 which make the fields upper case if case insensitive is selected, but leave them as entered if case sensitive is selected. Many of the notes given for the other search algorithms also apply here: For example, the coding on line 1 prevents the user from searching with a blank search field.

Program Code

For a full copy of the program code, please refer to Appendix A.

Procedures and Variables

There are three public subroutines in this program, and each is dealt with in detail in Detailed Algorithm Design, and are seen in full in Appendix A. Their page references are:

Function of Subroutine	Subroutine Name	Page References
Searching for Members (Case Insensitive)	FindClient	Detailed Design: Page 42 Appendix A: Page xi
Searching for Members (Case Sensitive)	FindClientExact	Detailed Design: Page 43 Appendix A: Page xii
Searching for Tracks	FindTrack	Detailed Design: Page 44 Appendix A: Page xviii

A list of all the variables in the program follows, along with their functions and any relevant references:

Variable Name	Type	Function	Reference
BookDate	String	Used to store the date of a booking when updating dynamic SQL	Appendix A, page vi
BookTime	String	Used to store the time of a booking when updating dynamic SQL	Appendix A, page vi
found	Boolean	Used to signify whether any tracks associated with the current booking have been found (used in the coding to delete a track from a booking)	Appendix A, page ix
namePart	String	Used in searches to store the text being searched for	Searching for Tracks, page 44
num	Integer	Used throughout the program when hiding irrelevant buttons and labels, and also when showing relevant ones	Appendix A, page iii (one example of many)
origin	Integer	Used when on the form for deleting a track from a booking to store the record number of the first track found which is associated with a particular booking.	Appendix A, page ix
position	String	Used in searches to store the current string of letters to which the search text is being compared.	Searching for Tracks, page 44
recnum	Integer	Used in implementation of navigation text boxes	Appendix A, page vii (one example of many)
recordno	Integer	Used in the coding for deleting a track from a booking to store the current position in the file	Appendix A, page v

User Manual

Introduction

This Hospital Radio database system is designed so that it can:

- Store and allow update and deletion of information about each song in the record library
- Find a track or tracks by searching on any of the above criteria.
- Allow the user to browse through the songs available and print details of any chosen song.
- Store and allow update and deletion of information about each member
- Find a member or members by searching on any of the above criteria.
- Allow the user to browse through a database of members view the details of any chosen member.
- Allow a presenter (and **only** a presenter) to book an hour when they can use the studio.
- Allow the printing of a list of bookings
- Allow the printing of members' details ordered by status should be available, as this could serve as a useful telephone directory.
- Allow the printing of a list of all of the tracks in the music database should also be made available, so that the few presenters who choose to plan some of their shows at home are catered for.

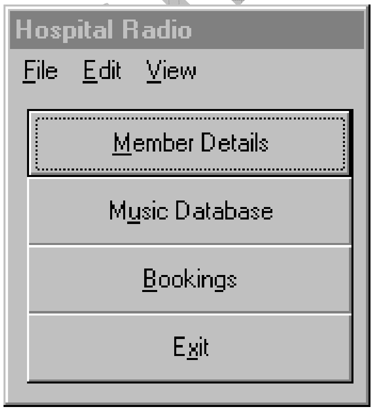
This User Manual is designed to guide you through this program, and also to provide information on any error messages which you may experience.

Installation

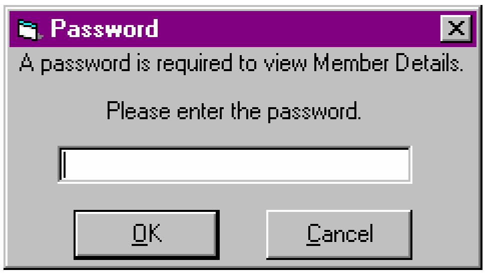
Installation is automated: Simply run the set-up executable and follow onscreen instructions. Any modern system should be able to run this program, and it has been specifically designed to run on the Hospital Radio computer.

Screen Displays

To open the program, click the Hospital Radio icon.



This is the **main menu**. All of the programs features can be accessed from here, and this will load when the program is first started. There are two ways to navigate the program: For access only to simple features, simply click the large buttons. For access to more complex features, use the standard windows menu at the top of the window. Printable reports are accessible via the "View" menu, whilst find, edit, add and delete features are accessed via the large buttons or the "Edit" menu.

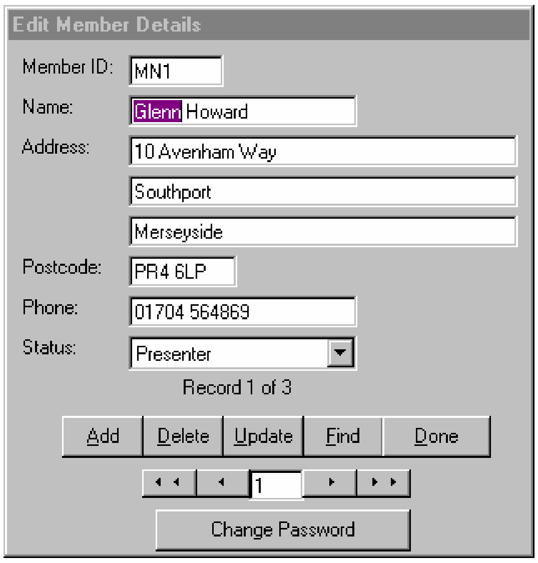


Password

A password is required to view Member Details.

Please enter the password.

This is the **password dialog box**. This will appear if you attempt to edit or view Member Details. This is to ensure that only people with the relevant access rights can access these sensitive details. The password is always available from the secretary, and he will change the password regularly for added security.



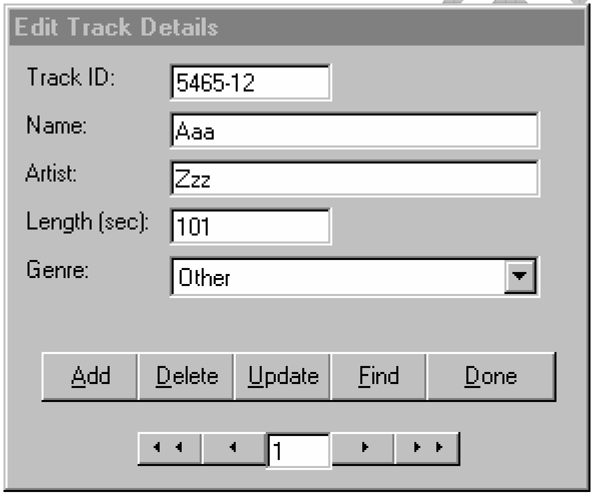
Edit Member Details

Member ID:	MN1
Name:	Glenn Howard
Address:	10 Avenham Way
	Southport
	Merseyside
Postcode:	PR4 6LP
Phone:	01704 564869
Status:	Presenter

Record 1 of 3

This is the **Edit Member Details** screen, which can only be accessed after a password has been entered. To edit someone's details, simple change the displayed information and click Update. To navigate to a particular person's details, use the navigation buttons or enter their record number in the navigation text box. Alternatively, use the **Find Facility**. To Delete someone from the database, simply click Delete. To Add someone, simply click "Add", enter their details, and click "Add" again.

The password can be changed by clicking "Change Password" and following the onscreen instructions.

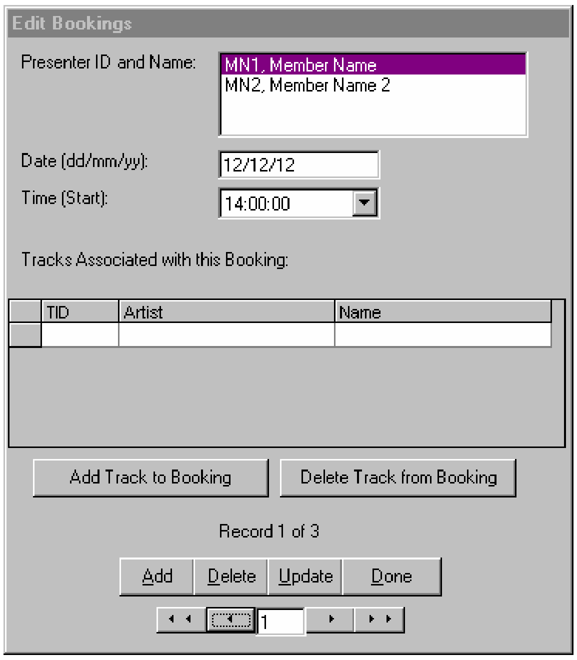


Edit Track Details

Track ID:	5465-12
Name:	Aaa
Artist:	Zzz
Length (sec):	101
Genre:	Other

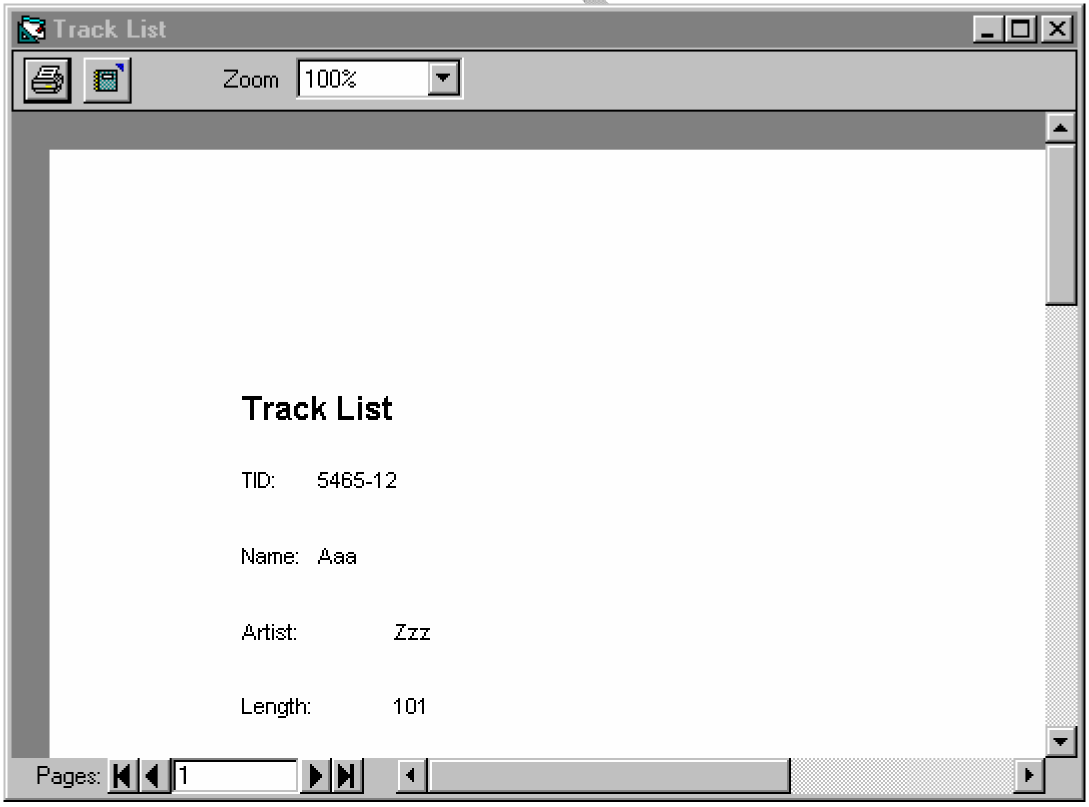
This is the **Edit Track Details** screen, and works in the same way as the **Edit Member Details** screen. However, no password is required to access these details.

Again, a **Find Facility** is available to help you to search for a particular track.



This is the **Edit Bookings** screen. From here, you may browse bookings which have been made, as well as delete or add new bookings. To book the studio, click “Add”, select the presenter you would like to add, enter the details and click “Add” again. If you wish to book tracks to use during your booking, then find your booking and click “Add Track to Booking”. Select your track, and click Add. If you wish to remove a track from your booking, click “Delete Track from Booking”, then select the track and click Delete. To delete a booking altogether, delete all tracks from the booking and then click Delete on the main **Edit Bookings** screen.

Note: If you do not book the studio, you should not expect it to be available for your use!



Track List

TID: 5465-12

Name: Aaa

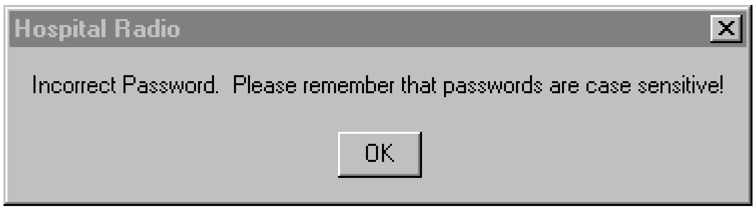
Artist: Zzz

Length: 101

This is the window in which reports will be displayed. You may browse the reports on the computer, or click the Printer button in order to print the report.

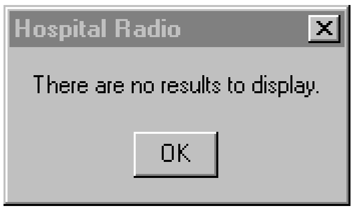
Error Messages

Remember that any error messages are there to guide you – there is not need to be afraid of them! If an error not indexed here is displayed, then please get in touch with me and I'll try and sort it out for you.



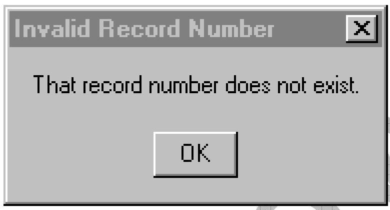
Incorrect Password. Please remember that passwords are case sensitive!

This message is displayed if you enter an incorrect password. Check the password, and re-enter it. If you still cannot access the Member Details, then I suggest you contact the secretary to check the password again. If, for any reason, the password is forgotten, then please contact me and I will reset it.



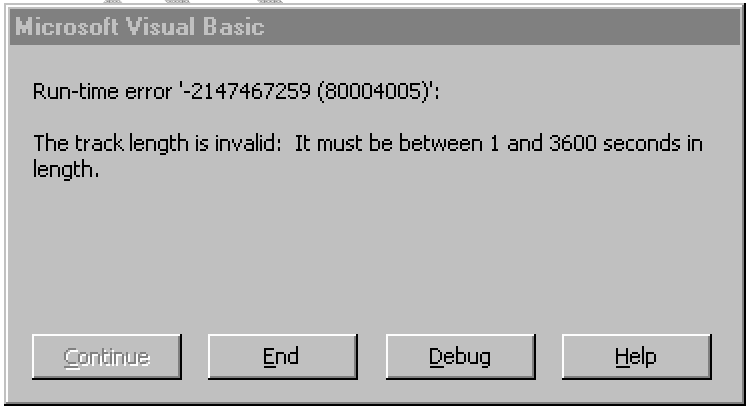
There are no results to display.

This will be displayed if your search finds no results, or no further results if you have asked the program to "Find Next". I suggest that you check the case sensitivity setting, as "FRANK", for example, will not find "Frank" if case sensitivity is turned on.



That record number does not exist.

This message will be displayed if you enter a record number into the navigation text box, and that record number does not exist.



Run-time error '-2147467259 (80004005)':

The track length is invalid: It must be between 1 and 3600 seconds in length.

This error will be displayed if you enter an invalid track length. Currently, due to a limitation of the system, this may cause the program to close.

Known Problems and Limitations

- Entering an invalid track length may cause the program to close.
- A booking cannot be deleted until all tracks associations have been deleted.

Appraisal

Project Performance against Analysis Objectives and Possible Extensions

The Objectives of the Project were discussed on page 9, during the Analysis stage.

Taking each objective in turn:

The system must store and allow update and deletion of information about each song in the record library

This objective has been fully met, with all the required data being stored in the Music Database. In future development, however, it may be prudent to add more track genres than those originally suggested and implemented.

Find a track or tracks by searching on any of the above criteria. The results of this search will be displayed on the screen for immediate reference.

Again, this criterion has been fulfilled by the Find Facility of the program. Printing of search results is not available in the program, but as this was not one of the original objectives, I do not see this as a problem. However, it may be an area which could be extended in future by using a combination of dynamic SQL and the Data Reporting Environment.

Allow the user to browse through the songs available and view details of any chosen song.

This feature of the program has been implemented. For future development, the opportunity to print the details of a single track could be implemented, or even a request system whereby a track could be added to a list of track to be played, along with the details of a particular patient.

Store and allow update and deletion of information about each member.

This criterion has been fulfilled, with all the required data being stored. In future, further details about each member may become necessary or useful, and so this may be an area for future development.

Find a member or members by searching on any of the above criteria.

Again, this criterion has been fulfilled by the Find Facility of the program. Printing of search results is not available in the program, but as this was not one of the original objectives, I do not see this as a problem. However, it may be an area which could be extended in future by using a combination of dynamic SQL and the Data Reporting Environment.

Allow the user to browse through a database of members view the details of any chosen member.

This feature of the program has been implemented. For future development, the opportunity to print the details of a single member could be implemented.

*Allow a presenter (and **only** a presenter) to book an hour when they can use the studio. Although not originally suggested, after discussion with current members, a system whereby a data sheet of bookings could be printed has now been suggested.*

Both the original booking objective, and the suggested printing objective have been met. In future, a request system could also be devised to complement the booking system.

Print members' details ordered by status should be available, as this could serve as a useful telephone directory and print a list of all of the tracks in the music database should also be made available

Both of these criteria have been met using the Visual Basic Data Reporting Environment, and examples of the printed reports can be seen in Appendix B. I cannot see much potential for future development in this particular area of the program.

User Feedback

A letter from the Technical Manager of Newtown Hospital Radio providing feedback on the program can be found in Appendix C. Since all of the original criteria specified were met, the Newtown Hospital Radio team were pleased with the new system.