CG2.1 ANALSYS DESIGN

Starshine amateur dramatic society

Farnborough Sixth Form College
Candidate number: 28914
Centre Number: 58913

Hassan Minhas

CG2.1: Analysis and Design

CG2.1/1 – Problem Definition

Introduction

The Starshine Amateur Dramatic Society, also known as SADS, is a dramatic society who put on productions which consists of modern day thrillers, family pantomimes and comedies through costume dramas since 1996. They usually put on two productions each year but due to increasing popularity they have decided to put another show on, in for early autumn. SADS have decided that this show will be a musical show to showcase the talent of their new members.

The shows are normally performed in the church hall but they will not be able to use this facility so they have decided to use the local theatre which is larger in size, has better facilities such as disables access and a bigger auditorium. Due to this SADS have also decided to produce a computer based system or database to ensure that all the bookings are recorded and all double bookings are avoided.

Aims

The aims of this project are to:

- Allow customer details to be input and stored
- Display all available seats
- Enable seats to be booked for the correct day
- Ensure that seats are not double booked
- Store and retrieve details of all bookings made by a customer
- Output total income from ticket sales for each performance
- Create a security system
- Enable cancelations of bookings/refunds
- To have access levels in the security system to allow the control of what information is being displayed
- Produce the receipt
- To be able to Archive bookings older than 12 months
- To be able to do group bookings
- To be able to pay by card
- To have a feature to add in extra performances
- To add validations to the database

Limitations

- No e-mail confirmation: the system is not capable to send information via the internet
- There will be no app for customers to use on their phones to be able to book seats
- No automatic book up system

- Will not be to book online
- There will be only one computer running this database because of networking issues.
- There will be no screen available for customers to book on themselves. No self-service system
- There won't be any touch screens available for customers to book their tickets by themselves
- No automatic logout
- We won't be able to print any tickets as we don't know the performance names and the time of the performances.

Assumptions

- Login Feature – To restrict some users from not accessing the customer card details
- Group bookings – Because customers will like to sit next to their friends and families. Furthermore, it can get a bit annoying if there wasn't a group bookings feature as you would have to keep on booking for every single customer.
- Admin levels for the login feature - to restrict access to some lower members of staff. We only want administrators to be able to view the card details.
- Feature to add in extra performances – so if in the future you need to use this system again it would be easy to add performances.
- End seats on a row are disabled seats. The actual end seat is the disabled seat rather than the space next to the disabled seat.
- It will store e-mail addresses just in case the company want to add an e-mail confirmation system or for promotional uses.
- Equipment – we assume that there is a computer available for the system to run on and that there is a printer so that we will be able to print receipts.
- Location - I will assume that the performances will be in Hampshire so I will assume that people only from the neighbouring counties will attend the performances as there is no description of the location of where the performances will be being performed at.

CG2.1/2 – Objectives

Date to be held: Booking details (eg, Cost) in a bookings table, Customer details (eg, name, address) in a Customer table, Passwords and Usernames in a login table for security.

Output: Profit, Total income, Seats available, Event details

Processing: Total sales, Profit, Queries, Avoid Double bookings, how many seats remaining.

Technology: Microsoft Access 2007, Microsoft Word and a modern computer which operates windows 7.

Validations: To stop any human error for being made and to allow the data to be easily read because it would be in a set format because of the validation eg. unique checks for Double bookings, length checks for telephone number, Format Checks for postcodes and dates, Range checks for age, Presence checks for all fields.

Aesthetics: Colour scheme - Dark Blue, Light Blue, Grey and white. Also a self-designed logo which will be present throughout most of the database to ensure a house style.

Security: A password protected entry system that requires a unique username with a strong password.

Ease of use: the database will consist of a well-structured layout, easy to find documents with easy readable buttons and a 'back' button to return to the mainframe in case of confusions in all forms and reports.

Types of user interface: The user will be initially asked to login, once this has happened they will redirected to a start screen from which they will be able to access all forms and reports which will be clearly laid out by buttons.

Objectives

- Allow customer details to be input and stored – I will use forms to input data and will use record sets in conjunctions with the forms to link them to the tables which will be where all the data is stored.
- Display all available seats – For this task I will use a report. First I will calculate how many seats are left then display the results on to a report.
- Enable seats to be booked for the correct day – There will be a conformations page in which the user will tell the buyer all of the details of the bookings. Also if they still make a mistake then there will be a drop down box (on the form where the booking is made) in which there will the times and dates of the performances which will be set in the performance form.
- Ensure that seats are not double booked – there will be a validation rule in the table to not let any booking to be the same.
- Store and retrieve details of all bookings made by a customer – I will use a table named customer table to store data and use record sets to link the table to a report in which the data will be displayed.
- Output total income from ticket sales for each performance – I will first calculate the total income then I will use a report to display this data.

Tables

- **Customer details** – To allow customer details to be stored (Aims this will help me meet are 'Allow customer details to be input and stored')

- **Booking Details** – to allow customer details to be stored (Aims this will help me meet are 'Store and retrieve' details of all bookings made by a customer')
- **Seats** - to allow me to store the number of seats and what they will be referred to throughout the database (eg , A1) (Aims this will help me meet are 'Enable seats to be booked for the correct day' and 'Ensure that seats are not double booked')
- **Security details** - to enable security details such as username and password to be stored

Input (forms)

- **Login Form** – Username, Password
- **Customer details** - Name, surname, telephone number, e-mail address, address, age, disabled or not
- **Booking form** – Date and time of performance, cost, surname, number of people, disabled or not, seat number, Card payment option
- **Refund form** – Customer Name, booking ID

Outputs (reports)

- Number of seats available
- Total income from tickets sales for each performance
- Customer Booking details- Date and time of performance, cost, surname, number of people, disabled or not, seat number
- Card Details – Card Number, Customer Name, expiry date of card, Security Number.

Processing tasks

- Calculate number of seats available – I would do total number of seats take away number of seats booked to get the number of seats available
- Total income
- Password system
- Invoice – To calculate the total cost of a booking.
- No Double Bookings - To add validations to stop any seats being booked twice.

Data Validations

- **Range check** - Restrict user ability so they are not allowed to amend the physical number of seats in each row. The total will always be more than 0 but less that the number of seats times the price of the seats.
- **Unique check** - Double Bookings (seats will not be able to be booked on twice). You can't have the same e-mail address as somebody else.
- **Presence Checks** - Customer surname will always be present. Seat ID will also be always present.

- **Lookup checks** – Customer title must be Mr, Mrs or Miss. Is the customer disabled or not? There will be a list of the neighbouring Counties only.
- **Input masks/format check** - Customer's postcode will be in the format of a postcode (eg GU23 7YU). Customer forename must not contain any numbers. Telephone number must not contain letters.
- **Type check** – The performance date must be in the format of the English date (dd/mm/yy) and price per ticket must only contain numbers (eg. £1.99).

CG2.1/3 – Justification

- Allow customer details to be input and stored – Access would be good for this task because it can easily create forms to input data and allows the user to create tables to store data. Furthermore, you can easily create record stets in access to link the table to the form.
- Display all available seats – I will use access for this task because access will easily let me create a form in which it will say the number of seats available. I will also need to do a calculation which will be total number of seats take away number of bookings. This will also be done in access be it is simple and efficient.
- Enable seats to be booked for the correct day – Access would be good for this task because I create a form called performances in which there are details of the performances like date and time and I will link this information to a drop down box in the bookings form using the drop down box feature in access and the record set feature in access
- Ensure that seats are not double booked – Access have a validation feature which I will use in order for double bookings not to be made. I will ensure the data must not be the same between each booking
- Store and retrieve details of all bookings made by a customer – access has the ability to create tables to store data and to create reports in order to retrieve data. Also I could link the report to the table using record sets.
- Output total income from ticket sales for each performance – Access has the ability to create reports from tables so that I can Output total income from ticket sales for each performance, making access the best choice for this task. Moreover, the visual basic function in access will enable me to do calculations and processing tasks which will allow me to display data such as the total income from ticket sales.
- Furthermore access will also be a very useful feature because it will enable me to cancel any bookings made by allowing me to link my booking table to be linked to a refund form.

CG2.1/4 – Data structures and methods of Access

Tables and validations

1). Table design and validation

tblBookings				
Attribute	Notes	Validations to be used		
		Type	Range	Lookup
CustomerID	Foreign Key	Auto Number		
PerformanceNo	Composite Key and Foreign Key linking to the performance table	Number		
SeatNumber	Composite Key Foreign Key linking to the Seats table	Number	1-20	
Row	Composite Key Foreign Key linking to the Seats table	Text 1	A – H J-L	
Methodofpayment		Text 11		'Cheque' 'Cash' 'Credit Card,'

tblCustomer						
Attribute	Notes	Default Value	Validations to be used			
			Type	Format (Input Mask)	Presence required	Lookup
CustomerID	Primary Key		Auto Number	N/A	N/A	N/A
Title		Mr	Text 4			Mr, Mrs, Miss, Ms
FirstName			Text 15			
Surname			Text 25		Yes	
DateofBirth			Date/Time	00/00/0000;0;_		
TelephoneNumber			Text 15	\(99999") "00090009;;_		
Postcode			Text 8	>LL00\ 0LL;0;_		
City			Text 20			
County			Text 15			
Road			Text 25			
Residencenumber			Text 20			
Email			Text 50			
CardType			Text 11			Debit Card;Credit Card
creditcardnumber			Integer			
expirydate			Date/time	00/00/0000;0;_		
Securitycode			Text 10	Password		

tblSecurity			
Attribute	Validations to be used		
	Type	Presence required	Lookup
Username	Text 20	Yes	
Password	Text 20	Yes	
AccessLevel	Text 20		'Staff';'Admin'

tblPerformance				
Attribute	Notes	Validations to be used		
		Type	Format (Input Mask)	Presence required
PerformanceNumber	Primary Key	Auto Number		Yes
Performancename	Primary Key	Text 20		Yes
performancedate	Primary Key	Date	00/00/0000;0;_	

tblSeats				
Attribute	Notes	Validations to be used		
		Type	Range	Format (Input Mask)
RowNumber	Primary Key	Text 1	1-20	
SeatNumber		Number	A – H J-L	
Disabled		Boolean		Yes/no
Price		Number		Currency

2). Tables in standard form

tblBookings :(*Customer ID*, Method-of-payment, *Performance Number , Seat Number, Row number*)

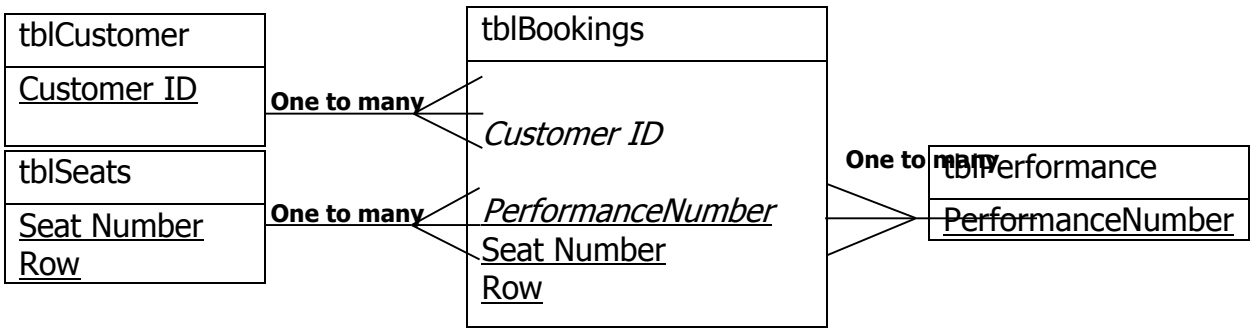
tblCustomer: (Customer ID, Title, First Name, Surname, Date of Birth, Telephone Number, Postcode, Town/City/Village, County, Road, Residence number/name, E-mail address, Card Type, Full credit card number, expiry date, Security code)

tblSecurity: (Username, Password, Access Level)

tblSeats: (Seat Number, Row Number, Disabled, Price)

tblPerformane (performance date, Performance name, Performance Number)

3). Entity relationships diagram

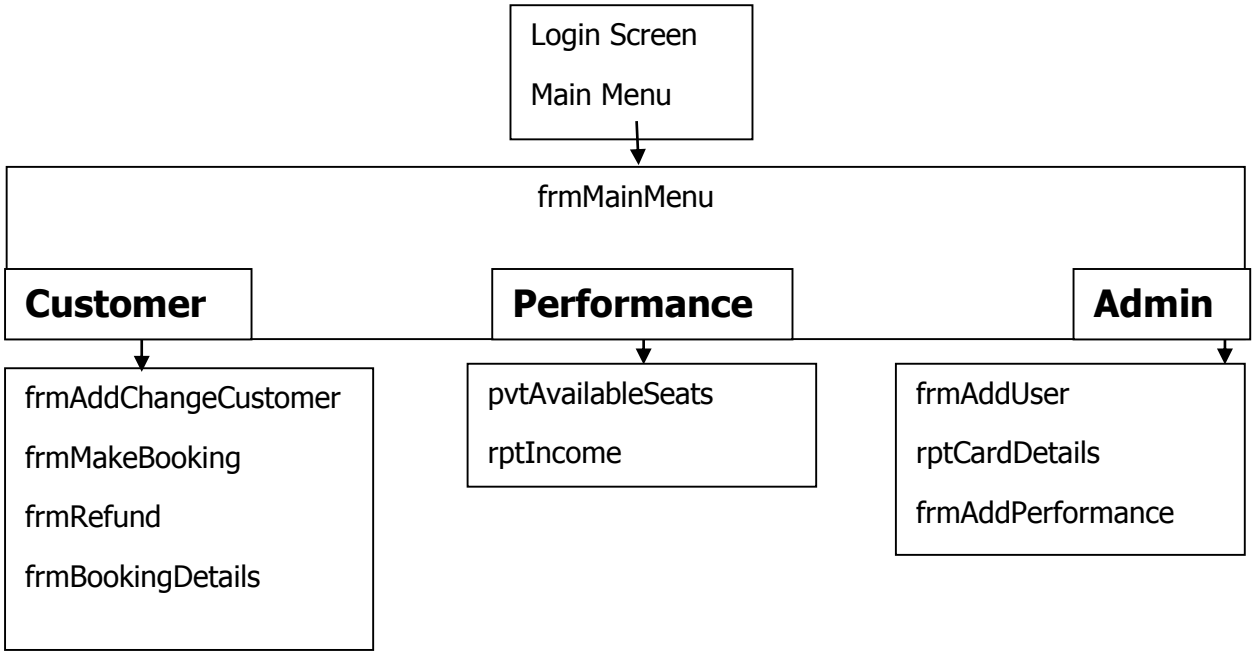


4). Data access forms required to use tables

Form Name	Purpose	Table(s)	Methods of data access
Login Form	To ensure important data (eg. Credit card details) can only be accessed by trusted users by giving access levels to certain users	tblSecurity	Unbound Form. Data entered using recordsets
Customer details frmAddChangeCustomer	To update, change or add customer records	tblCustomer	Unbound Form. Data accessed by traversing the records in the table using VBA. Data added using recordsets.
Create Booking form frmMakeBooking	To create a booking for a customer	tblBookings tblCustomer tblSeats tblCardDetails (if customer is paying by card)	Unbound Form. Data added using recordsets.
Main Menu	To allow user to browse through forms and reports		Unbound Form
frmBookingDetails	Allow user to see which customer made which bookings	tblBookings tblCustomer	Unbound Form. Data added using recordsets.
Refund form frmRefund	To allow customers to cancel a booking they have made	tblBookings tblSeats tblCardDetails (if customer is paying by card)	Unbound Form. Data deleted using recordsets.
Add extra performances	To allow users to add extra performances	tblPerformance	Unbound Form. Data added using recordsets.
frmAddUser	Allows admin staff to add users to the system	tblSecurity	Unbound Form. Data added using recordsets.

CG2.1/5 – User Interface

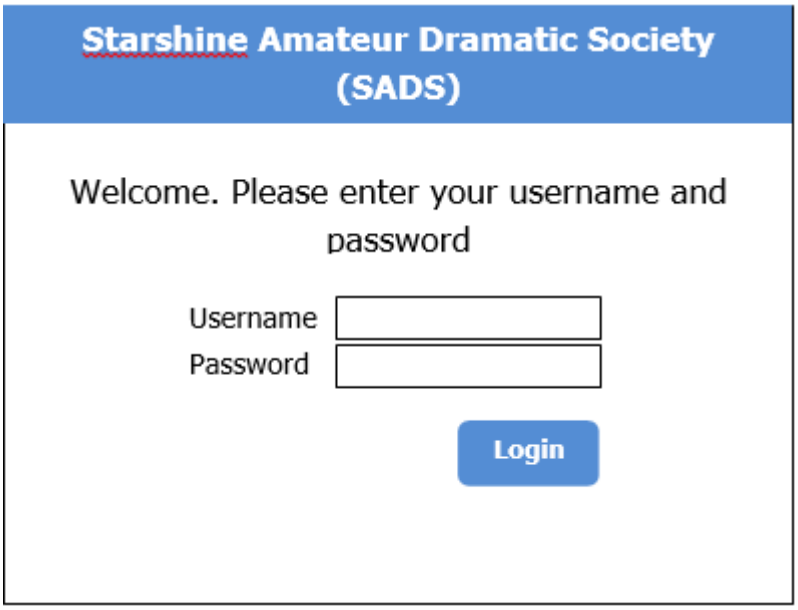
1) Forms and Outputs



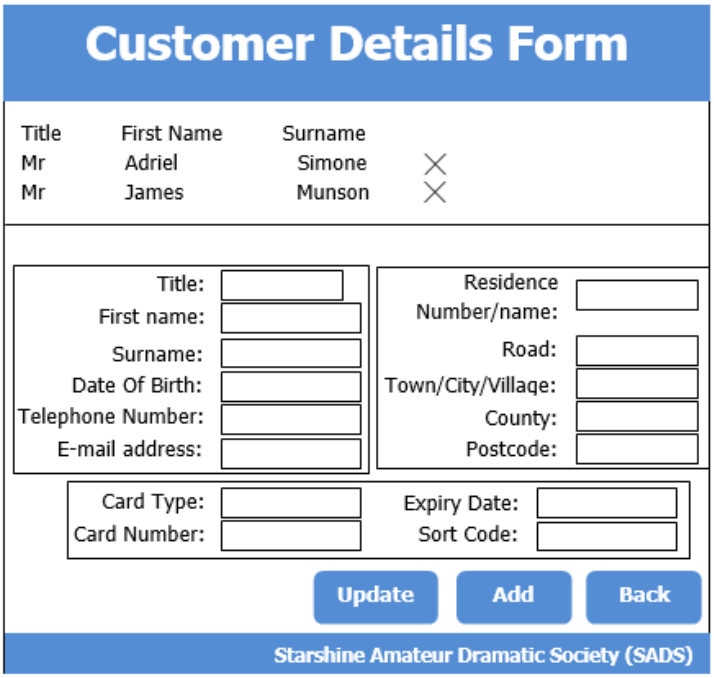
Target User

For this task I am assuming that the user is over 16 and has some computer experience on how to work with access and other Microsoft applications. Also I am assuming that the user knows how to start up a computer and how to navigate around programs. Furthermore, I am also assuming that the user know how to type. I will try my best in making a system that is easily to navigate through by clearly laying out information and buttons. The data will not be too small nor will it be too large because if it was too small, some elderly people may not be able to see what is written and if it was too large it would take up too much space. Moreover, I will keep the database very simple so it is easy to navigate through.

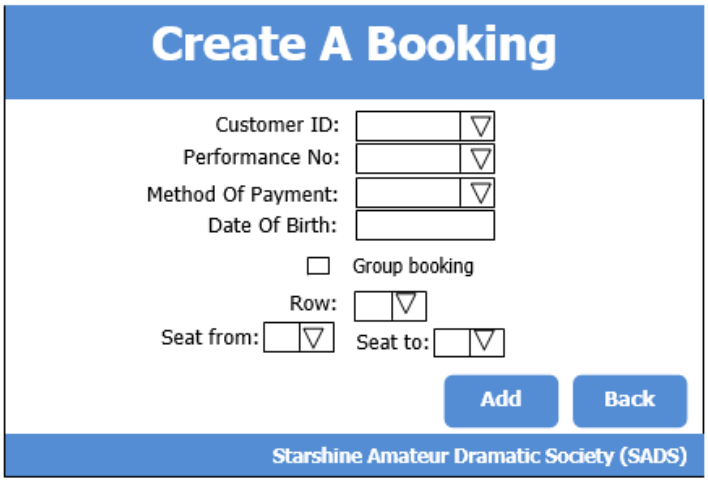
Forum Name	frmLoginForm
Created	Using Blank Form
Purpose	Prevents unauthorised access. Secures the system
Back colour	White
Text Font	Tahoma,11
Data source	N/A
Logo	N/A
Sub form	N/A



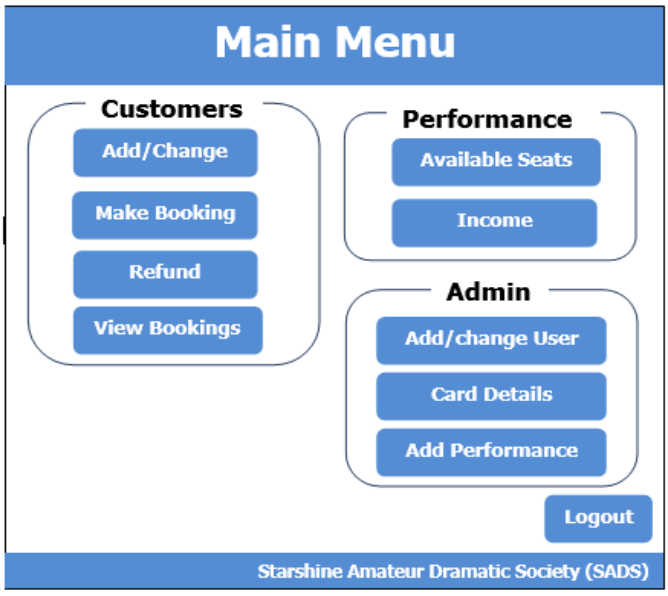
Forum Name	frmCustomerDetails
Created	Using Blank Form
Purpose	Allows user to enter customer details
Back colour	White
Text Font	Tahoma,9
Data source	tblCustomer
Logo	N/A
Sub form	N/A



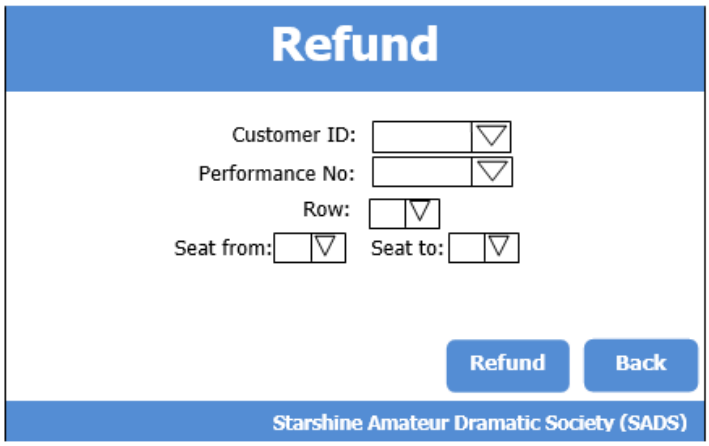
Forum Name	frmBooking
Created	Using Blank Form
Purpose	Allow bookings to be made
Back colour	White
Text Font	Tahoma,9
Data source	N/A
Logo	N/A
Sub form	N/A



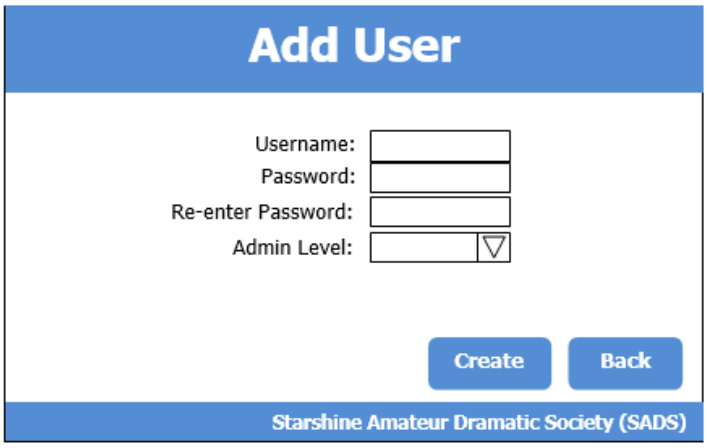
Forum Name	frmMainMenu
Created	Using blank form
Purpose	Allows users to navigate through the system easily
Back colour	White
Text Font	Tahoma, 9
Data source	N/A
Logo	Created using word
Sub form	N/A



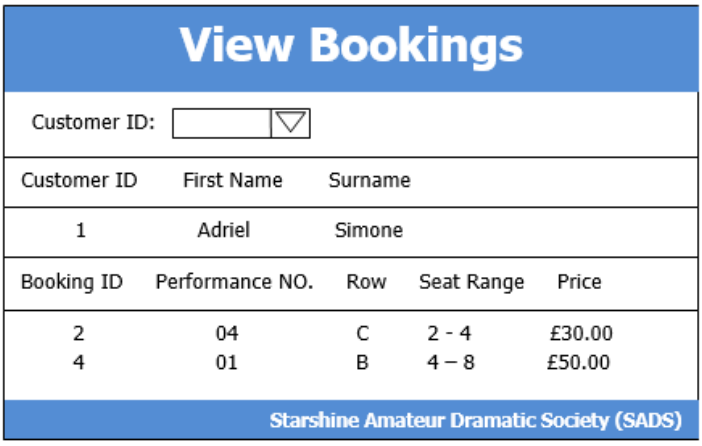
Forum Name	frmRefund
Created	Using Blank Form
Purpose	Allow Refunds to be made
Back colour	White
Text Font	Tahoma,9
Data source	tblBookings, tblCustomer
Logo	N/A
Sub form	N/A



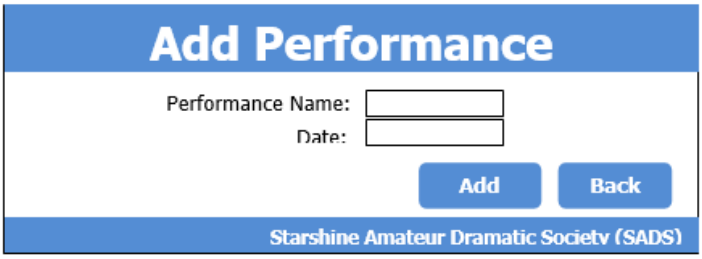
Forum Name	frmAddUser
Created	Using Blank Form
Purpose	Allows admin to add users to the database
Back colour	White
Text Font	Tahoma,9
Data source	N/A
Logo	N/A
Sub form	N/A



Forum Name	frmBookingDetails
Created	Using Blank Form
Purpose	Allows users to view the bookings made by a customer
Back colour	White
Text Font	Tahoma,10
Data source	tblBookings
Logo	N/A
Sub form	N/A

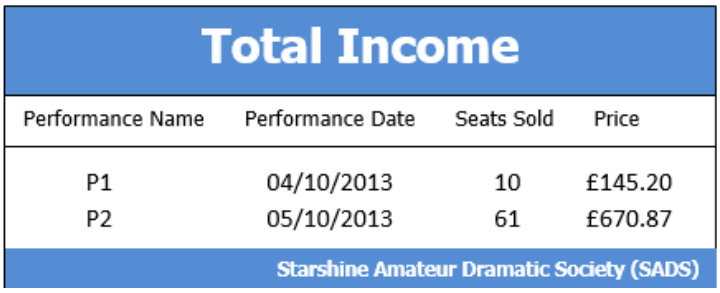


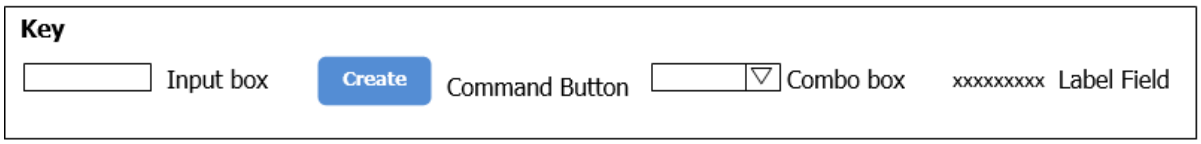
Forum Name	frmAddPerformance
Created	Using Blank Form
Purpose	Allows users to add extra performances
Back colour	White
Text Font	Tahoma,10
Data source	tblPerformance
Logo	N/A
Sub form	N/A



3) Data access report design

Report Name	Total Income
Data Source	qryincome
Purpose	To display the total income from ticket sales
Font	Tahoma,9
Logo	None
created	Using Wizard but modified as shown





Pivot Table	Available Seats
Data Source	tbleBookings, tblSeats
Purpose	To display the available seat
Font	Tahoma,12
Logo	None
created	Created using a blank pivot table

L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
K	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
J	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
H	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
G	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
F	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
E	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
D	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	
C	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17			
B	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15					
A	1	2	3	4	5	6	7	8	9	10	11	12	13	14						

Change Log	
Change	Reason
I will not produce a receipt	I will not enough time to complete this task. Furthermore this is not a main objective so it is not necessary for me to do this task
I will now include a performance table	I will need this in order to add extra performances
rptCustomerDetails will now be a form instead of a report	I will find it more easier and more practical to create a form to view a customer and a sub form in that form to view all the bookings made by that customer
I will now have a new form called frmAddperformance	To enable users to add extra performances to the system
I will now have a new form called frmAddUser	To enable administrators to add extra users to the system

Objectives

Objective	Description	Page Number
Allow customer details to be input and stored	I will use forms to input data and will use record sets in conjunctions with the forms to link them to the tables which will be where all the data is stored.	10
Display all available seats	For this task I will use a report. First I will calculate how many seats are left then display the results on to a report.	12
Enable seats to be booked for the correct day	There will be a conformations page in which the user will tell the buyer all of the details of the bookings. Also if they still make a mistake then there will be a drop down box (on the form where the booking is made) in which there will the times and dates of the performances which will be set in the performance form.	10
Ensure that seats are not double booked	There will be a validation rule in the table to not let any booking to be the same.	10
Store and retrieve details of all bookings made by a customer	I will use a table named customer table to store data and use record sets to link the table to a report in which the data will be displayed.	11
Output total income from ticket sales for each performance	I will first calculate the total income then I will use a report to display this data.	12
Add extra Performances	To have a feature to add in extra performances.	11
Login Form	To have access levels in the security system to allow the control of what information is being displayed.	10

G2.1/6 – Hardware and Software requirements

The minimum system requirements for the hardware are:
Monitor
Keyboard
Mouse
1 gigahertz (GHz) or faster 32-bit (x86) or **64-bit (x64)** processor
1 gigabyte (GB) RAM (32-bit) or 2 GB RAM (64-bit)
16 GB available hard disk space (32-bit) or 20 GB (64-bit)

The minimum system requirements for the Software are:
Microsoft Windows XP or above (Windows 7 recommended)
Microsoft Access 2007 V12.0.66.6.1000
Visual basics for applications (VBA)

CG2.1/7 – Processing stages

Query design

Query Name	Description	tables	Fields	Criteria
qrySales	A list that shows each of the bookings made on a performance	tblPerformance tblBookings	Performance Name Performance Date Ticket Price	Performance ID
qyrCustomerBookings	A query that lists all the bookings made by a customer	tblBookings tblCustomer	CustomerID, BookingsID, Seats, Price, Performance Date	CustomerID
qryTotalIncome	Calculates the total income from ticket sales	tblBookings tblSeats	Date, seat, row, price	Date
qryUpdateSeats	Updates the seat status	tblBookings tblSeats	Date Row Seat Booked	Booked
qryBetween	Copies the bookings to an archive table after 1 year	tblBookings		
qryAppend	Deletes the record which is in the bookings table which was copied to the archive table	tblBookings		

Macros

Macro name	pseudocode	description
AutoExec	Open frmLogin	Displays the login form when the application is invoked

Algorithms

Form Name: Login Form
Objective: To prevent unauthorised access

Login Button click

Create a recordset
Link recordset to security table

Loop until the end of the recordset has been reached
Check username is the same as the username in the security table
Check password is the same as the Password in in the security table
Close current form
Open main menu
Otherwise
Display a box saying password or username is wrong
End if
Exit Subroutine
End If
Move to the next record in the recordset
Loop
Display error message
End Subroutine

Name: Log out button
Objective: to let other users sign in

Log out button click

Declare the answer as a string
Display a message box asking whether the user is sure to log out

If the answer is yes then
Close the current form
And then open the login form
End If

End Subroutine

Form Name: Add performance form
Objective: To let admin users to add extra performances

Add performance Button click

Create a recordset
Link recordset to performance table

Create a new record
Performance date input box = performance date field in recordset
Performance name input box = performance name field in recordset

Update the record
Display a message box telling the user that the record has been created

Close the current form
Open the main menu
End Subroutine

Form Name: Add customer form
Objective: To let users to add customers
Add button click

Create a recordset
Link recordset to customers table

Add a new record to the customers table

Title input box = title field in recordset
First name input box = first name field in recordset
Surname input box = Surname field in recordset
Date of birth input box = date of birth field in recordset
Telephone Number input box = Telephone Number field in recordset
Email input box = Email field in recordset
Residence Number input box = Residence Number field in recordset
Road input box = Road field in recordset
City input box = City field in recordset
Postcode input box = Postcode field in recordset
Card Type input box = Card Type field in recordset
Credit Card Number input box = Credit Card Number field in recordset
Expiry Date input box = Expiry Date field in recordset
Security Code input box = Security Code field in recordset

Update the record
Close the current form

Display a message box telling the user the record has been created
Close the current form
Open the customers form

End Subroutine

Name: Delete customer button
Objective: To let users to delete customers
Delete button click

Display a message box asking whether you want to delete the record?

Declare a variable As String
Declare variable As Variant

For each selected variable item in the customers listbox
Delete the record from the table which is corresponding to the record selected
Next
Update the listbox

End If
End Subroutine

Form Name: Add users form
Objective: To let admin users to add users
Add button click

Create a recordset
Link recordset to security table

Add a new record

Username input box = Username field in recordset
Password input box = Password field in recordset
Level input box = Level field in recordset

Update the record

Close the current form
Open the Main Menu

End Subroutine

Form Name: Main Menu
Objective: To disable the add performances and add users button for normal users

As soon as the form has loaded

Create a recordset

Link recordset to security table

If the user is a admin the
Enable the add performances button
Enable the add users button
End If

End Subroutine

Form Name: Customers form, Customers listbox
Objective: to display the customer ID, first name and surname in a listbox

As soon as the form loads

Declare a variable as string
In the variable, store Customer ID, First name, and surname from the customers table. Store first name and surname as full name

The customer listbox will have 2 columns
The customer listbox will have "Table/Query" as a row source type
Store Customer ID, first name, and surname in the listbox
The customer listbox will store the declared variable as the row source
The customer listbox will not have any column titles

Update the listbox
End Subroutine

Form Name: Update customers
Objective: To let the user update a customer
Update button click

Create a recordset
Link recordset to customers table

Edit a record in the customers table

Title input box = title field in recordset
First name input box = first name field in recordset
Surname input box = Surname field in recordset
Date of birth input box = date of birth field in recordset

Telephone Number input box = Telephone Number field in recordset
Email input box = Email field in recordset
Residence Number input box = Residence Number field in recordset
Road input box = Road field in recordset
City input box = City field in recordset
Postcode input box = Postcode field in recordset
Card Type input box = Card Type field in recordset
Credit Card Number input box = Credit Card Number field in recordset
Expiry Date input box = Expiry Date field in recordset
Security Code input box = Security Code field in recordset

Update the record which was selected in the customer's listbox

Display a message box telling the user the record has been updated

Close the current form
Open the Customers form
End Subroutine

Form Name: Update customers (get selected listbox value)
Objective: the algorithm for displaying the selected customer in the customer's listbox on the update form

As soon as the form loads

Declare a variable as string
Declare a variable as Variant
Declare a variable as string
Create a recordset

Get the selected customer Id in the customer listbox and get all the data corresponding to that record. Get the record from the customers table

Link recordset to customers table

Title field in recordset = Title input box
first name field in recordset = First name input box
Surname field in recordset = Surname input box
Date of birth field in recordset = Date of birth input box
Telephone Number field in recordset =Telephone Number input box
Email field in recordset = Email input box
Residence Number field in recordset = Residence Number input box
Road field in recordset = Road input box
City field in recordset = City input box

Postcode field in recordset = Postcode input box
Card Type field in recordset = Card Type input box
Credit Card Number field in recordset = Credit Card Number input box
Expiry Date field in recordset = Expiry Date input box
Security Code field in recordset = Security Code input box

End Subroutine

Form Name: Create Bookings Form
Objective: To allow users to create bookings
Add button click

Create a recordset
Link recordset to Bookings table

Bookings ID input box = Bookings ID field in recordset
Customer ID input box = Customer ID field in recordset
Row input box = Row field in recordset
Seat Number input box = Seat Number field in recordset
Method of payment input box = Method of payment field in recordset
Performance ID input box = Performance ID field in recordset

Display a message box saying that the booking has been made

Close the current form
Open the Bookings form

End Subroutine

Form Name: Create Bookings Form
Objective: To allow users to create group and single bookings

Option box click

If the option box is selected then
  Show the Seat No to, text box
  Show the Seat No from, text box
  Show the row, text box
Else
  Hide the Seat No to, text box
  Hide the Seat No from, text box
  Show the row, text box
  Show the seat number, text box

End If

End Subroutine

Form Name: Create Bookings Form
Objective: To allow users to create group and single bookings part 2

As soon as the form loads

Hide the Seat No to, text box
Hide the Seat No from, text box
Show the row, text box
Show the seat number, text box

End Subroutine

Form Name: Bookings form, Bookings listbox
Objective: to be able to view all the bookings in a listbox

As soon as the form loads

Declare a variable as string
In the variable, store Customer ID, Bookings ID, Performance ID, Row, Seat number, Method of payment from the bookings table
The bookings listbox will have 2 columns
The bookings listbox will have "Table/Query" as a row source type
Store Customer ID, Bookings ID, Performance ID, Row, Seat number, Method of payment
The bookings listbox will store the declared variable as the row source
The bookings listbox will have column titles

Update the listbox

End Subroutine

CG2.1/8 – Evaluation criteria

1. **Storage Tasks**

task	Table Name	Purpose/Notes	Reference	Success
1	tblCustomer	Hold customer account details		
2	tblBookings	Holds bookings details		
3	tblSecurity	Hold user's and admin's login details		
4	tblSeats	Holds the seat details		
5	tblPerformance	Hold the performance details		

2. Input Tasks

task	Form Name	Purpose/Notes	Reference	Success
1	frmLogin	Enables to show valuable information to only trusted users		
2	frmUpdateCustomer	Allows users to edit customer details		
3	frmCreateBooking	Allows bookings to be made		
4	frmAddUser	Allows users to be created		
5	frmAddPerformance	Allows users to add extra performances		
6	frmAddCustomer	Allows users to add customer details		
7	frmRefund	Enables users to refund tickets		

3. Output Tasks

task	Form Name	Purpose/Notes	Reference	Success
1	rptIncome	Allows users to view the total income from ticket sales		
2	frmBookingDetails	Allows users to view bookings made by a customer		
3	pvtSeats	Displays all seats and shows the status of the seats		

4. Processing Tasks

task	Name	Purpose/Notes	Reference	Success
1	Password Security system (VB)	To prevent any unauthorised access		
2	Income	Calculates the total income from sales		
3	Double Bookings verifier	To prevent and double booking made by setting up validations		
4	Seat Code	A piece of code which calculates how many seats there are in a row and when this has been done it will allow the user to select a row and underneath that there will be combo box listing all the seat numbers of that specific row		

5. Quantitative performance Considerations (Volumetrics)

task	Name	Purpose/Notes	Reference	Success
1	Customer table capacity	Up to 1000 customers		
2	Bookings table capacity	Up to 5000 Bookings		
3	Access time	A single record found within 6 seconds		
4	User table capacity	Can store up to 100 users		

6. Qualitative Evaluation and Criteria (Usability and suitability)

task	Name	Purpose/Notes	Reference	Success
1	Usability	The solution is simple and easy to use		
2	Suitability	Suitable for experienced IT user		
3	Menus	Large buttons, easy to use and navigate		

7. Data security and integrity of the system

task	Name	Purpose/Notes	Reference	Success
1	Password	Can be used by user		
2	Password	Must be more than 5 characters		
3	User levels	Two levels, Standard and administrator		
4	Backup	Automatic backup routine		