

Event Management System

Group Project

Table of Contents

Chapter 1	Introduction.....	5
Chapter 2	Interfaces.....	6
Chapter 3	Validation and Error Handling.....	12
Chapter 4	Conclusion	16

List of Figures

Figure 1: Login Interface	6
Figure 2: Dashboard.....	7
Figure 3: Customer Interface	8
Figure 4: Booking Interface	9
Figure 5: Event Interface	10
Figure 6: Due Handling Interface	11
Figure 7: Validation 1.....	12
Figure 8: Validation 2.....	12
Figure 9: Error Handling 1.....	14
Figure 10: Error Handling 2.....	15

The Event Management System is a software tool created with Windows Forms in C# using the .NET framework. Its main goal is to simplify the management of event bookings for various occasions, including weddings, parties, and corporate events. The system enables users to manage all event details efficiently, such as customer information, specific event requirements, additional services, and financial transactions.

This application connects with a SQL Server database to store and access booking details. It offers a user-friendly interface for inputting booking information, calculating costs, and managing payments. Additionally, the system includes a powerful data retrieval feature, allowing users to view and manage all event bookings through a detailed data grid interface.

The Event Management System's key features include real-time validation of user input, robust error handling, and flexible data entry options, making it a reliable and easy-to-use tool. The project is designed to make event management more efficient and organized for both event planners and clients.



Figure 1: Login Interface

This is (Figure 1) the login screen where users are required to enter their username and password to access the Event Management System. The interface includes options to show the password and a login button to submit the credentials.



Figure 2: Dashboard

This interface (Figure 2) serves as the main navigation hub for the system. It allows users to access different sections, such as Customer, Booking, and Events. Also, can logout from the system.

Customer

CUSTOMER REGISTRATION

Name

ADD

NIC

UPDATE

Address

DELETE

Phone number

RESET

Email

	cusId	Name	NIC	Address	PhoneNumber	Email
▶	2	Sasindu	992872772V	Colombo	725103137	sasindu@gmail.com
	3	Avishka	200257561479	Malwana	746784356	avishka@gmail.com
	1002	Sithumya	200279103518	Kotahena	712118456	sithumya@gmail.c...
*						

HOME

Figure 3: Customer Interface

This interface (Figure 3) is used to manage customer details within the system. Users can add new customers, update existing records, delete customers, or reset the form fields. The interface also displays a grid listing all registered customers along with their details such as name, NIC, address, phone number, and email.

BOOKING REGISTRATION

Customer ID: Name:

Event: Date / Time: No. of Persons:

Dishes

☐ Menu 1 ☐ Menu 2

Plate price (LKR)

Beverages

	Qty.	Price
<input type="checkbox"/> Soda	<input type="text"/>	<input type="text"/>
<input type="checkbox"/> Wine	<input type="text"/>	<input type="text"/>
<input type="checkbox"/> Beer	<input type="text"/>	<input type="text"/>
<input type="checkbox"/> Coca Cola	<input type="text"/>	<input type="text"/>
<input type="checkbox"/> Juice	<input type="text"/>	<input type="text"/>

Additional Events

☐ DJ

☐ Decoration

☐ Wedding Car

☐ Poruwa Ceremony

☐ Cake

Payments

Net Total:

Discount:

Gross Total:

Payment Option

1st Installment:

Due Amount:

Buttons: ADD, RESET, Calculate, HOME

Figure 4: Booking Interface

This interface (Figure 4) is used for creating and managing event bookings. Users can enter details such as the customer ID, event type, date and time, number of persons, menu options, beverages, additional events, and payment options. When adding the booking details, the system automatically calculates the net total. After applying any discounts, it further calculates the gross total. Customers have the option to pay directly or in installments, and the system also calculates the due amount accordingly. This provides a comprehensive tool for booking management, ensuring all financial aspects are accurately handled.

Events

VIEW BOOKING

BookingID	CustomerName	Event	EventDate	NoOfPersons	Dishes	PlatePrice	Beverages	AdditionalEven	PaymentOption	PaymentStatus	FirstInstalment	DueAmount	NetTotal	Discount	GrossTotal	CustomerID
2	Anshika	Party	5/25/2024	140		1200	Soda	DJ	Installment	Paid	100000	0	169300	0	169300	3
3	Saanshu	Party	5/17/2024	100		1000			Installment	Paid	50000	0	100000	0	100000	2
4	Anshika	Party	5/20/2024	130		1300			Direct Paym...	Paid	0	169000	169000	0	169000	3
5	Saanshu	Party	5/31/2024	134	40	1300			Direct Paym...	Paid	0	0	174200	40	104520	2
6	Sithumya	Wedding	5/22/2024	100	20	2500	Soda, Wine...	DJ, Decorati...	Direct Paym...	Paid	0	0	315250	20	252200	1002
7	Anshika	Party	5/25/2024	200	20	2000	Soda, Wine	DJ	Installment	Paid	100000	0	417000	20	333600	3

Figure 5: Event Interface

This interface (Figure 5) allows users to view and manage all event bookings. It displays a comprehensive data grid with details such as booking ID, customer name, event type, event date, number of persons, menu options, beverages, additional services, payment status, and other financial details. Users can also delete bookings or navigate to the home screen from this interface.

DUE HANDLING

Name

Gross Total

First Installment

Pay

Back

	BookingID	CustomerName	FirstInstallment	DueAmount	GrossTotal
*					

Figure 6: Due Handling Interface

This interface (Figure 6) is designed to manage payment dues for events. Users can select a booking directly from the grid view, which displays only the customers who have outstanding payments. Upon selection, the system automatically fills in the relevant text boxes, including the due amount. The interface then provides options to proceed with payment or navigate back to the previous screen, streamlining the process of handling due payments efficiently.

Chapter 3

Validation and Error Handling

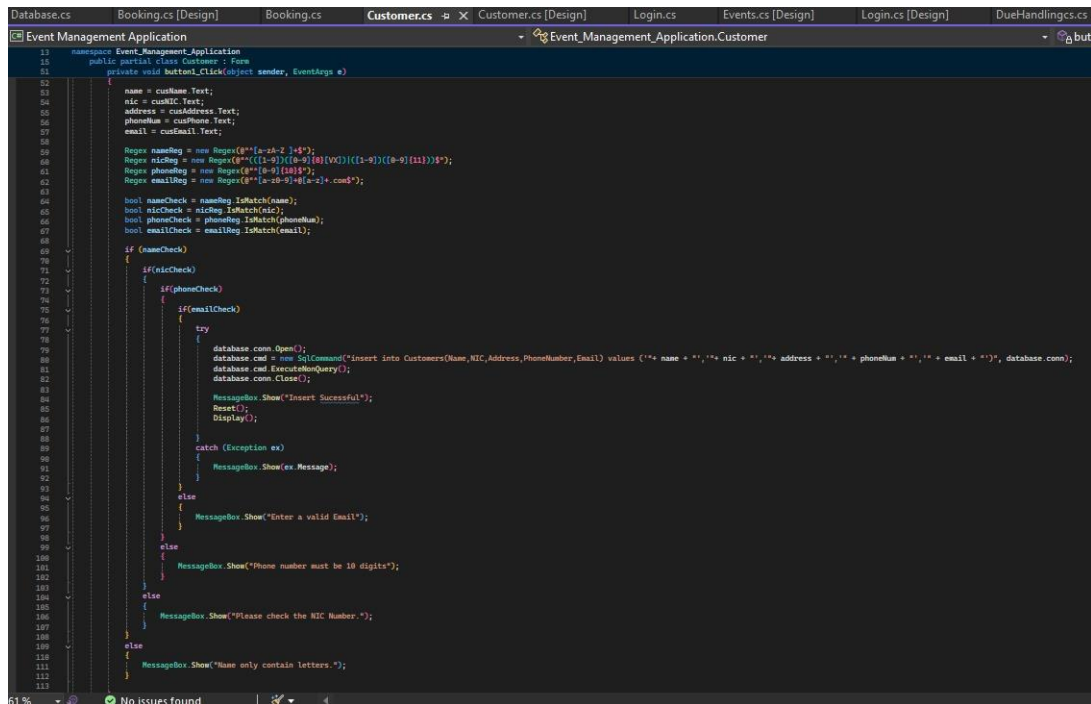
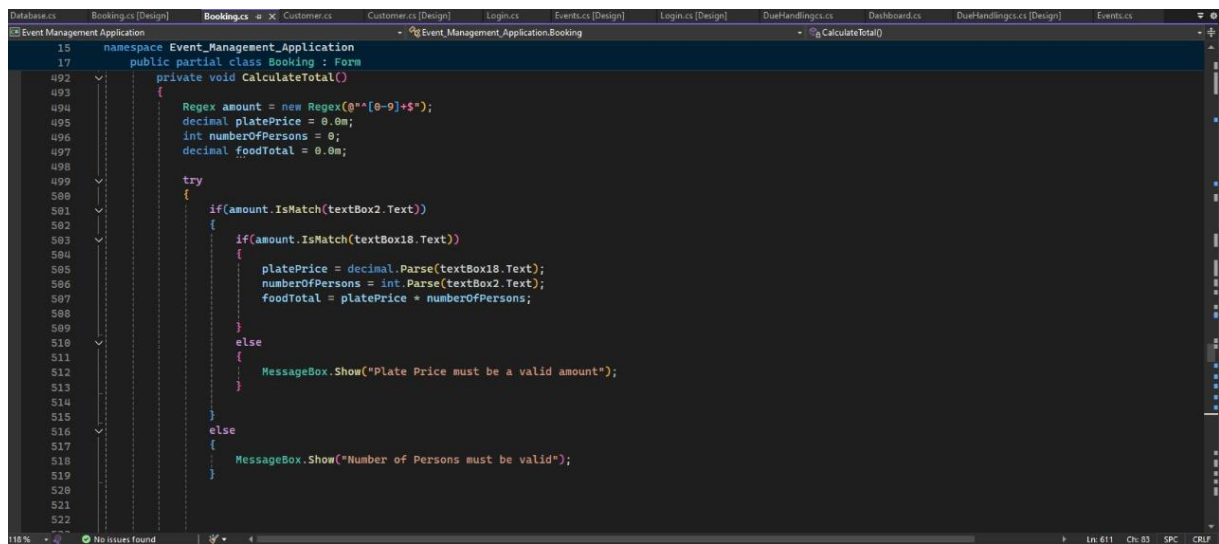


Figure 7: Validation 1

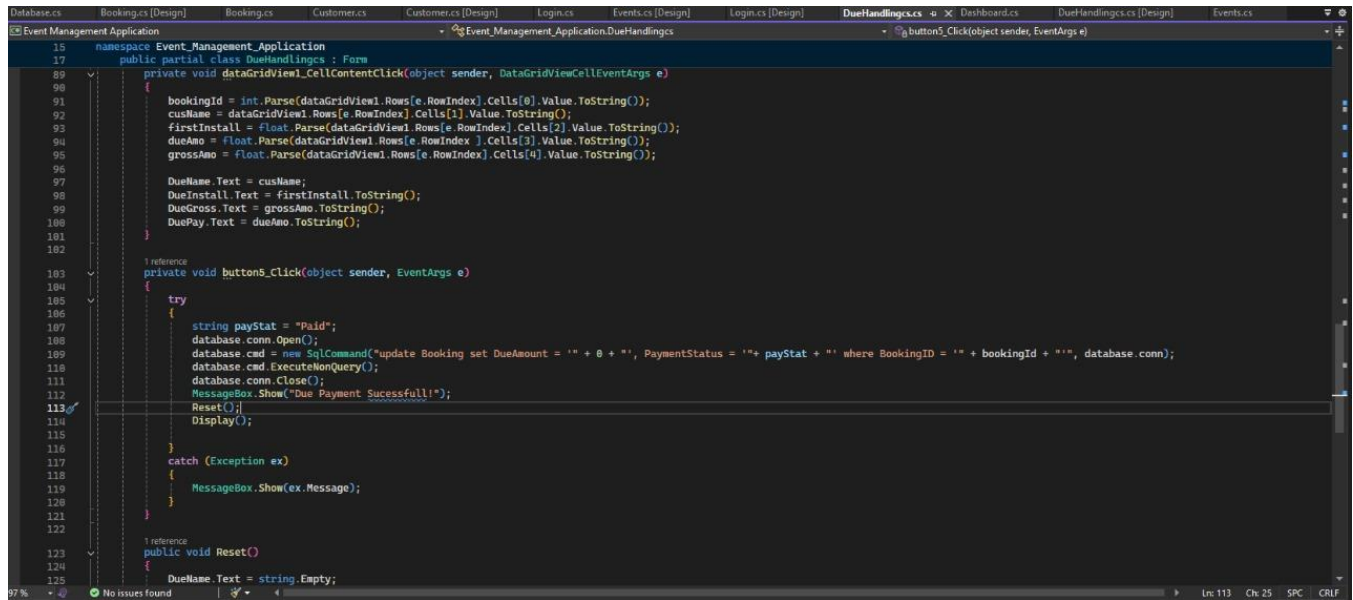
- **Name Validation:** The system checks (Figure 7) that the name contains only letters using a Regex. If the name contains invalid characters, an error message "Name only contains letters" is displayed.
- **NIC Validation:** The NIC number is validated against a specific format using Regex. If the format is incorrect, the system prompts the user with "Please check the NIC Number."
- **Phone Number Validation:** The system ensures that the phone number consists of exactly 10 digits. If the number is invalid, a message "Phone number must be 10 digits" is shown.
- **Email Validation:** A Regex is used to validate the email format. If the email is not valid, the system displays "Enter a valid email."



```
15 namespace Event_Management_Application
16 {
17     public partial class Booking : Form
18     {
19         private void CalculateTotal()
20         {
21             Regex amount = new Regex(@"^[0-9]+$");
22             decimal platePrice = 0.0m;
23             int numberOfPersons = 0;
24             decimal foodTotal = 0.0m;
25
26             try
27             {
28                 if (amount.IsMatch(textBox2.Text))
29                 {
30                     if (amount.IsMatch(textBox18.Text))
31                     {
32                         platePrice = decimal.Parse(textBox18.Text);
33                         numberOfPersons = int.Parse(textBox2.Text);
34                         foodTotal = platePrice * numberOfPersons;
35                     }
36                     else
37                     {
38                         MessageBox.Show("Plate Price must be a valid amount");
39                     }
40                 }
41                 else
42                 {
43                     MessageBox.Show("Number of Persons must be valid");
44                 }
45             }
46             catch { }
47         }
48     }
49 }
```

Figure 8: Error Handling 1

- **Plate Price Validation:** A Regex pattern (Figure 8) is used to ensure that the plate price entered by the user is a valid numeric value. If the input does not match the expected format, the system shows a message box with the error: "Plate Price must be a valid amount."
- **Number of Persons Validation:** The system validates that the number of persons entered is a valid integer. If the input is invalid, the user is notified with the message: "Number of Persons must be valid."



```
15 namespace Event_Management_Application
16 {
17     public partial class DueHandlings : Form
18     {
19         private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
20         {
21             bookingId = int.Parse(dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString());
22             cusName = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();
23             firstInstall = float.Parse(dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString());
24             dueAmo = float.Parse(dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString());
25             grossAmo = float.Parse(dataGridView1.Rows[e.RowIndex].Cells[4].Value.ToString());
26
27             DueName.Text = cusName;
28             DueInstall.Text = firstInstall.ToString();
29             DueGross.Text = grossAmo.ToString();
30             DuePay.Text = dueAmo.ToString();
31         }
32
33         1 reference
34         private void button5_Click(object sender, EventArgs e)
35         {
36             try
37             {
38                 string payStat = "Paid";
39                 database.conn.Open();
40                 database.cmd = new SqlCommand("update Booking set DueAmount = '' + @ + '', PaymentStatus = ''+ payStat + '' where BookingID = '' + bookingId + ''", database.conn);
41                 database.cmd.ExecuteNonQuery();
42                 database.conn.Close();
43                 MessageBox.Show("Due Payment Successful!");
44                 Reset();
45                 Display();
46             }
47             catch (Exception ex)
48             {
49                 MessageBox.Show(ex.Message);
50             }
51         }
52
53         1 reference
54         public void Reset()
55         {
56             DueName.Text = string.Empty;
57         }
58     }
59 }
```

Figure 9: Error Handling 1

The try-catch block (Figure 9) is used when updating the booking details in the database after processing a payment. If any error occurs during this process, the exception is caught, and an error message is shown.

The Event Management System developed in this project effectively meets the critical requirements for organizing and managing events, providing a comprehensive set of tools for customer registration, booking management, and financial tracking. The system's user-friendly interfaces, such as Customer Registration, Booking Registration, and Due Handling, simplify the event planning process by ensuring that all data is accurately recorded and easily accessible.

Key functionalities, including automated calculations, payment processing, and data visualization via grid views, contribute to enhanced operational efficiency and informed decision-making. This project not only fulfills the initial objectives but also offers a scalable and robust solution adaptable to various event management scenarios. The successful implementation of this system highlights its potential to significantly improve event management, making it an invaluable resource for event coordinators and planners.

Our Group Members

L.G Dinithi kawya : UGC0122030

Avishka Udara : UGC0122017

Sasindu Jayawardana : UGC0122032