

**MS806 – Business Application Programming**

**Student Name: Jayakarthi Boovendran**

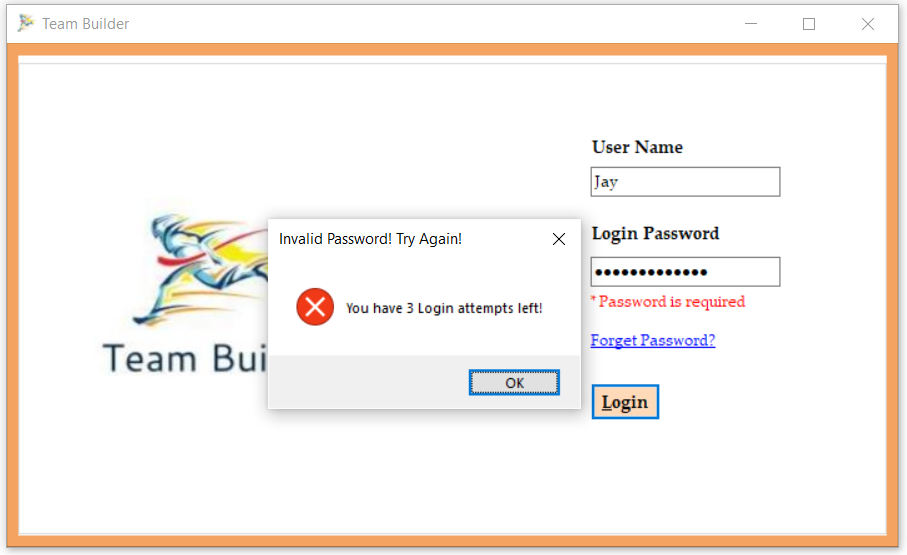
**NUI Galway ID Number: 19230487**

**Assignment:** **3**

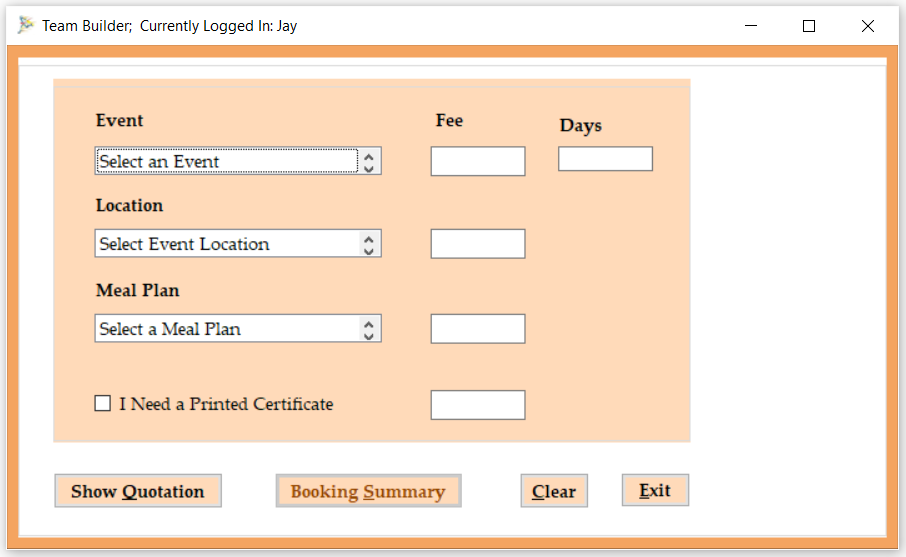
**Date of Submission:** **25/10/2019**

**Assignment III: Team Builder Ltd - Event Booking**

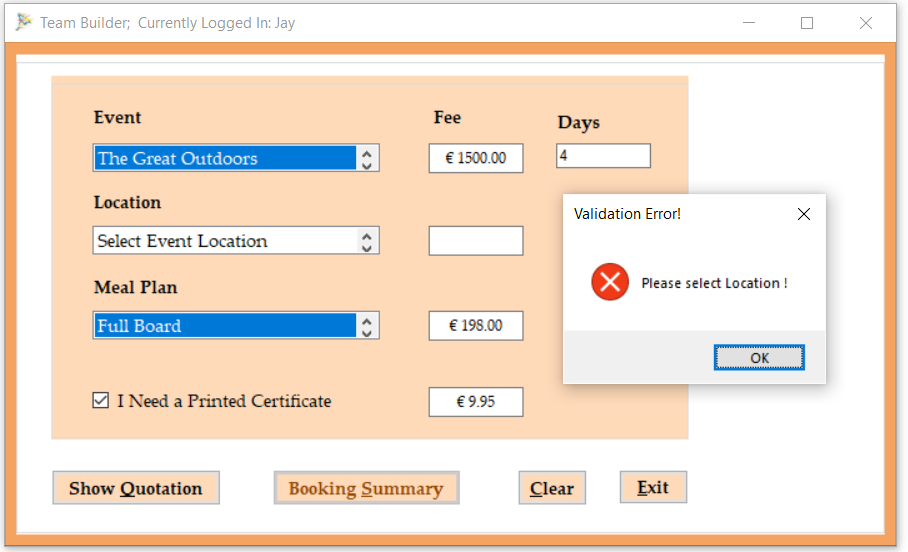
**Screenshots of Application at Runtime:**



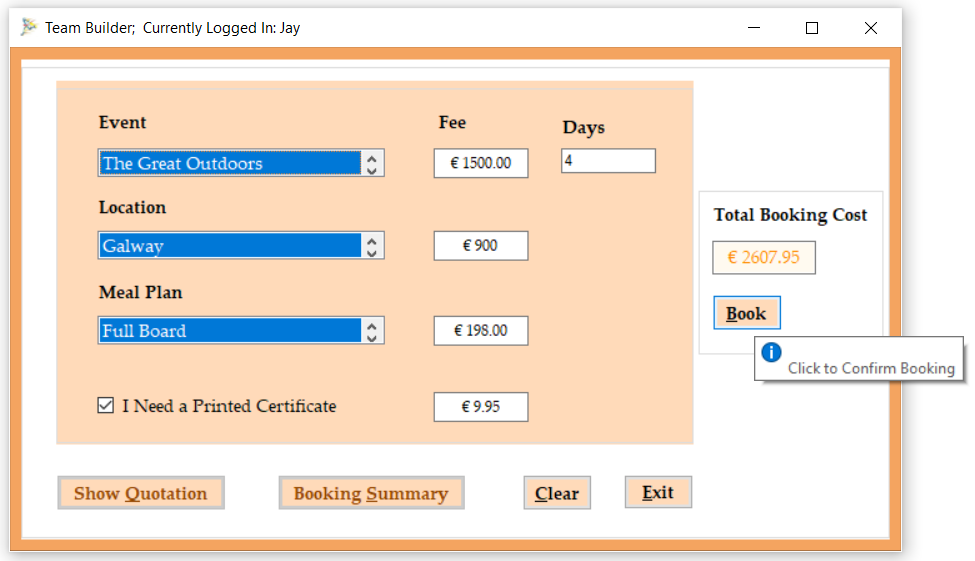
**Fig 1: User Login Validations**



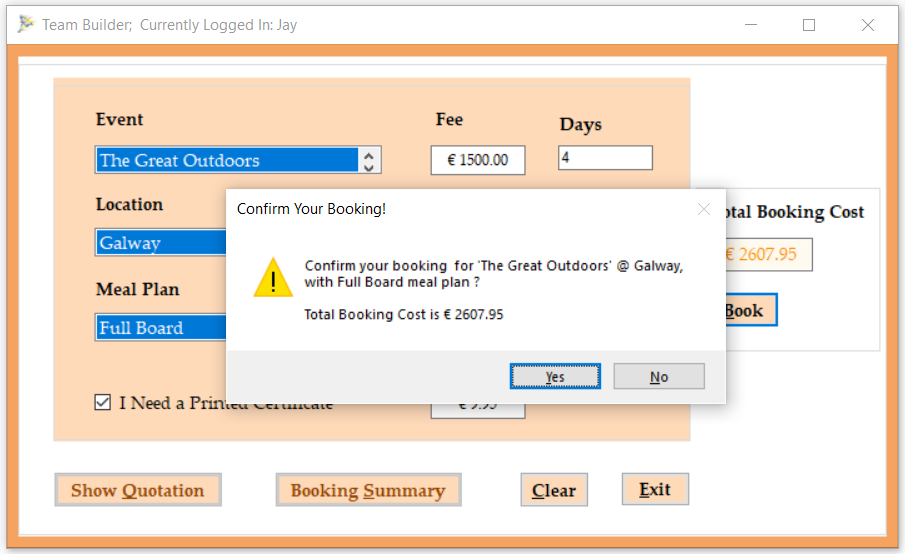
**Fig 2: Events Booking Page – Pristine State**



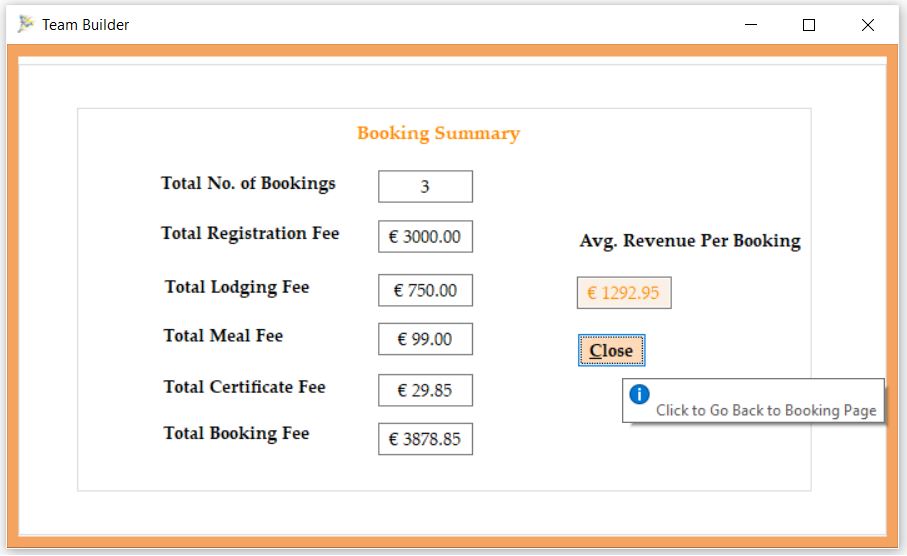
**Fig 3: User Input Validation**



**Fig 4: Display functionality on the click of ‘Show Quotation’ button**



**Fig 5: Confirm Booking pop-up on the click of ‘Book’ button**



**Fig 6: Booking Summary Page**

**Application Source Code:**

/\*

## MS806 - Business Application Programming

## Assignment No : 3

## Name : Jayakarthi Boovendran | ## StudentID : 19230487

## Date of Submission : October 25, 2019

## Assignment Description : To Build a Window Form for 'Team Builder Ltd' to enable users to book for events

and to compute Company's overall booking summary.

## The Form takes in user's info like Event Name, Event Location, Meal Plan and Certificate options as input,

and associated Receipts are computed and reported as Output. \*/

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Windows.Forms;

using Newtonsoft.Json;

namespace TeamBuilder

{

public partial class TeamBuilderForm : Form

{

//Login Retry Count= 4

int RetryCount = int.Parse(ConfigurationManager.AppSettings.Get("RetryCount").ToString());

// To hold Form's Meta Information

MetaEvent MetaData = new MetaEvent();

// To hold individual bookings

Quotation quotation = new Quotation();

// To hold booking summary

BookSummary Summary = new BookSummary();

public TeamBuilderForm()

{

InitializeComponent();

//Loads login page

InitializeLoginPage();

}

// Loads Login Page

public void InitializeLoginPage()

{

TeamBuilderLoginGroupBox.Visible = true;

TeamBuilderEventsGroupBox.Visible = false;

SummaryGroupBox.Visible = false;

UserNameTextBox.Text = string.Empty;

LoginPassword.Text = string.Empty;

UserNameTextBox.Focus();

}

// Loads Event Booking Page

public void ShowEventsPage()

{

TeamBuilderLoginGroupBox.Visible = false;

TeamBuilderEventsGroupBox.Visible = true;

BookingGroupBox.Visible = false;

EventsListBox.Focus();

}

#region Load Form Meta Data

public void LoadFormMetaData()

{

try

{

// Event, Location and MealPlan Info JSONs are fetched from App.Config

MetaData.EventData = JsonConvert.DeserializeObject<IList<Event>>(ConfigurationManager.AppSettings.Get("EventsJSON").ToString());

MetaData.EventLocationData = JsonConvert.DeserializeObject<IList<EventLocation>>(ConfigurationManager.AppSettings.Get("LocationJSON").ToString());

MetaData.MealData = JsonConvert.DeserializeObject<IList<MealInfo>>(ConfigurationManager.AppSettings.Get("MealInfoJSON").ToString());

//The Loaded MetaData is inserted into their respective listboxes

// Event

foreach (int EventID in MetaData.EventData.Select(x => x.EventID))

{

EventsListBox.Items.Insert(EventID, MetaData.EventData.Where(y => y.EventID == EventID).Select(x => x.EventName).FirstOrDefault());

}

// Location

foreach (int LocationID in MetaData.EventLocationData.Select(x => x.LocationID))

{

EventLocationListBox.Items.Insert(LocationID, MetaData.EventLocationData.Where(y => y.LocationID == LocationID).Select(x => x.Location).FirstOrDefault());

}

// Meal Plan

foreach (int MealID in MetaData.MealData.Select(x => x.MealID))

{

MealPlanListBox.Items.Insert(MealID, MetaData.MealData.Where(y => y.MealID == MealID).Select(x => x.MealType).FirstOrDefault());

}

// Clear listbox selections

EventsListBox.SelectedIndex = 0;

EventsListBox.ClearSelected();

EventLocationListBox.SelectedIndex = 0;

EventLocationListBox.ClearSelected();

MealPlanListBox.SelectedIndex = 0;

MealPlanListBox.ClearSelected();

PrintedCertificateCheckBox.Checked = false;

// Clears 'Fee' info

EventRegFeeTextBox.Text = string.Empty;

EventDaysTextBox.Text = string.Empty;

LodingFeeTextBox.Text = string.Empty;

MealPriceTextBox.Text = string.Empty;

CertificatePriceTextBox.Text = string.Empty;

//Booking Summary button

if (Summary.Summary.Count > 0) { BookSummaryButton.Enabled = true; } else { BookSummaryButton.Enabled = false; }

}

catch (Exception ex)

{

MessageBox.Show("Error:" + ex.Message, "Error Loading Meta Data!", MessageBoxButtons.OK, MessageBoxIcon.Error);

InitializeLoginPage();

}

}

#endregion

#region User Login

private void Login\_Click(object sender, EventArgs e)

{

try

{

//Checks user credentials

If(LoginPassword.Text.Equals(ConfigurationManager.AppSettings.Get("LoginPassword").ToString()) && RetryCount > 0)

{

this.Text = this.Text + "; Currently Logged In: " + UserNameTextBox.Text;

//Load Event Booking Page

LoadFormMetaData();

ShowEventsPage();

}

else

{

RetryCount--;

if (RetryCount < 0) RetryCount = 0;

if (RetryCount == 0)

{

// User has exceeded the max no.of.login retries

throw new LoginException();

}

else

{

// Invalid Credentials, prompts for retry

throw new RetryLoginException(RetryCount);

}

}

}

catch (LoginException ex)

{

//Closes the application

ShowLoginErrorPopup(ex.Message);

Close();

}

catch (RetryLoginException ex)

{

// promts for login retry

ShowRetryLoginPopup(ex.Message);

LoginPassword.Text = string.Empty;

LoginPassword.Focus();

}

}

// User-defined Exceptions for Login Validation

public class LoginException : Exception

{

public LoginException() : base(String.Format("Crash!, Login Error!"))

{

}

}

public class RetryLoginException : Exception

{

public RetryLoginException(int LoginAttempts) : base(String.Format("You have {0} Login attempts left!", LoginAttempts))

{

}

}

// Login Error pop-ups

public void ShowLoginErrorPopup(string Message)

{

MessageBox.Show(Message, "Login Error!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

public void ShowRetryLoginPopup(string Message)

{

MessageBox.Show(Message, "Invalid Password! Try Again!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

#endregion

#region Exit

// Closes the application

private void ExitButton\_Click(object sender, EventArgs e)

{

Close();

}

#endregion

#region User Input Validation

// Required Fields : Event Name, Location

// Shows Error Message if the required fields are not selected

public bool InputValidation()

{

bool ValidationResult = true;

string FieldName = string.Empty;

try

{

// Event Validation

FieldName = "Event";

// Error is thrown if, Event is not selected or 'Select An Event' option is selected

if (EventsListBox.SelectedIndex == -1 || EventsListBox.SelectedIndex == 0)

{

ValidationResult = false;

EventsListBox.Focus();

// InputValidation Exception is thrown

throw new InputValidationException(FieldName);

}

// Location Validation

FieldName = "Location";

if (EventLocationListBox.SelectedIndex == -1 || EventLocationListBox.SelectedIndex == 0)

{

ValidationResult = false;

EventLocationListBox.Focus();

throw new InputValidationException(FieldName);

}

return ValidationResult;

}

catch (InputValidationException ex)

{

// Exception handling: Error Message is Displayed

ShowInputValidationErrorPopup(ex.Message);

return ValidationResult;

}

}

//Custom-exception for Input validation

public class InputValidationException : Exception

{

public InputValidationException(string FieldName) : base(String.Format("Please select {0} !", FieldName))

{

}

}

// Error message pop-up

public void ShowInputValidationErrorPopup(string Message)

{

MessageBox.Show(Message, "Validation Error!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

#endregion

#region Display

// Displays Booking Info like Total booking cost and cost associated with each user selection

private void DisplayButton\_Click(object sender, EventArgs e)

{

int eventdays = 0;

try

{

if (InputValidation())

{

// quotation object is formed based on the user selection

quotation.EventInfo = MetaData.EventData.Where(x => x.EventID == EventsListBox.SelectedIndex).Select(x => x).FirstOrDefault();

eventdays = quotation.EventInfo.Days;

quotation.LocationInfo = MetaData.EventLocationData.Where(x => x.LocationID == EventLocationListBox.SelectedIndex).Select(x => x).FirstOrDefault();

quotation.LocationInfo.LodgingFee = quotation.LocationInfo.LodgingFee \* eventdays;

// if meal plan is not selected ; 'No meal' is selected by default

if (MealPlanListBox.SelectedIndex == -1) { MealPlanListBox.SelectedIndex = 0; }

quotation.MealInfo = MetaData.MealData.Where(x => x.MealID == MealPlanListBox.SelectedIndex).Select(x => x).FirstOrDefault();

quotation.MealInfo.Price = quotation.MealInfo.Price \*eventdays;

// Certificate price

Certificate certificate = new Certificate();

quotation.CertificateInfo = certificate;

quotation.CertificateInfo.IsCertificateRequired = PrintedCertificateCheckBox.Checked;

if (quotation.CertificateInfo.IsCertificateRequired) quotation.CertificateInfo.CertificatePrice = 9.95m; else quotation.CertificateInfo.CertificatePrice = 0.0m;

// Total Booking Cost

quotation.TotalBookingCost = quotation.EventInfo.RegistrationFee + quotation.LocationInfo.LodgingFee + quotation.MealInfo.Price + quotation.CertificateInfo.CertificatePrice;

TotalBookingCostTextBox.Text = "€ " + quotation.TotalBookingCost.ToString("f2");

BookingGroupBox.Visible = true;

DisplayButton.Enabled = false;

}

}

catch (Exception)

{

MessageBox.Show("Error in showing quotations, Contact the IT team", "Computational Error!", MessageBoxButtons.OK, MessageBoxIcon.Error);

LoadFormMetaData();

ShowEventsPage();

}

}

#endregion

#region Book Event

// Makes a booking for the selected event

private void BookButton\_Click(object sender, EventArgs e)

{

// gets user's confirmation on booking

DialogResult dialogResult = MessageBox.Show("Confirm your booking for '" + quotation.EventInfo.EventName + "' @ " + quotation.LocationInfo.Location + ", with " + quotation.MealInfo.MealType + " meal plan ?" + "\n\nTotal Booking Cost is € " + quotation.TotalBookingCost, "Confirm Your Booking!", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (dialogResult == DialogResult.Yes)

{

// Adds the quotation object formed during 'Display' to the Summary object

Summary.Summary.Add(quotation);

quotation = new Quotation();

MessageBox.Show("Your Booking is Confirmed!", "Confirmed!", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

// Clear Selection

LoadFormMetaData();

ShowEventsPage();

DisplayButton.Enabled = true;

}

else

{

BookingGroupBox.Visible = false;

DisplayButton.Enabled = true;

DisplayButton.Focus();

}

if (Summary.Summary.Count > 0) { BookSummaryButton.Enabled = true; } else { BookSummaryButton.Enabled = false; }

}

#endregion

#region Clear

// Clears form data

private void ClearButton\_Click(object sender, EventArgs e)

{

LoadFormMetaData();

ShowEventsPage();

}

#endregion

#region Auto-Fee Load on List box selection

// Loads 'Fee' info automatically based on the user's selection

// Events

private void EventsListBox\_SelectedIndexChanged(object sender, EventArgs e)

{

int days = 0;

decimal price = 0.0m;

switch (EventsListBox.SelectedItem)

{

case "Murder Mystery Weekend": price = 600.00m; days = 2; break;

case "CSI Weekend": price = 1000.00m; days = 3; break;

case "The Great Outdoors": price = 1500.00m; days = 4; break;

case "The Chase": price = 1800.00m; days = 6; break;

case "Digital Refersh": price = 599.00m; days = 2; break;

}

EventRegFeeTextBox.Text = "€ " + price;

EventDaysTextBox.Text = days.ToString();

}

// Location

// Location price = Location price per day \* no.of.event days

private void EventLocationListBox\_SelectedIndexChanged(object sender, EventArgs e)

{

LodingFeeTextBox.Text = "€ " + (MetaData.EventLocationData.Where(x => x.LocationID == EventLocationListBox.SelectedIndex).Select(x => x.LodgingFee).FirstOrDefault()\*int.Parse(EventDaysTextBox.Text)).ToString();

}

// Meal Plan

// Meal price = daily meal \* no.of.event days

private void MealPlanListBox\_SelectedIndexChanged(object sender, EventArgs e)

{

MealPriceTextBox.Text = "€ " + (MetaData.MealData.Where(x => x.MealID == MealPlanListBox.SelectedIndex).Select(x => x.Price).FirstOrDefault() \* int.Parse(EventDaysTextBox.Text)).ToString();

}

// Certificate

private void PrintedCertificateCheckBox\_CheckedChanged(object sender, EventArgs e)

{

if (PrintedCertificateCheckBox.Checked) CertificatePriceTextBox.Text = "€ 9.95"; else CertificatePriceTextBox.Text = "€ 0.00";

}

#endregion

#region Booking Summary

// Computes and displays booking summary

private void BookSummaryButton\_Click(object sender, EventArgs e)

{

TeamBuilderLoginGroupBox.Visible = true;

EventsGroupBox.Visible = false;

BookingGroupBox.Visible = false;

SummaryGroupBox.Visible = true;

TotalBookingsTextBox.Text = Summary.Summary.Count().ToString();

TotalRegFeeTextBox.Text = "€ " + Summary.Summary.Select(x => x.EventInfo.RegistrationFee).Sum().ToString("f2");

TotalLodgeFee.Text = "€ " + Summary.Summary.Select(x => x.LocationInfo.LodgingFee).Sum().ToString("f2");

TotalMealFee.Text = "€ " + Summary.Summary.Select(x => x.MealInfo.Price).Sum().ToString("f2");

TotalCertificateFeeTextBox.Text = "€ " + Summary.Summary.Select(x => x.CertificateInfo.CertificatePrice).Sum().ToString("f2");

TotalBookingFeeTextBox.Text = "€ " + Summary.Summary.Select(x => x.TotalBookingCost).Sum().ToString("f2");

AvgRevenueTextBox.Text = "€ " + (Summary.Summary.Select(x => x.TotalBookingCost).Sum() / Summary.Summary.Count()).ToString("f2");

SummaryCloseButton.Focus();

}

#endregion

// Closes Booking Summary page and redirects to Events Booking Page

private void SummaryCloseButton\_Click(object sender, EventArgs e)

{

TeamBuilderLoginGroupBox.Visible = false;

EventsGroupBox.Visible = true;

BookingGroupBox.Visible = false;

SummaryGroupBox.Visible = false;

LoadFormMetaData();

ShowEventsPage();

}

}

#region Modals

public class Event

{

public int EventID { get; set; }

public string EventName { get; set; }

public int Days { get; set; }

public decimal RegistrationFee { get; set; }

}

public class EventLocation

{

public int LocationID { get; set; }

public string Location { get; set; }

public decimal LodgingFee { get; set; }

}

public class MealInfo

{

public int MealID { get; set; }

public string MealType { get; set; }

public decimal Price { get; set; }

}

public class MetaEvent

{

public IList<Event> EventData = new List<Event>();

public IList<EventLocation> EventLocationData = new List<EventLocation>();

public IList<MealInfo> MealData = new List<MealInfo>();

}

public class Certificate

{

public bool IsCertificateRequired { get; set; }

public decimal CertificatePrice { get; set; }

}

public class Quotation

{

public Event EventInfo { get; set; }

public EventLocation LocationInfo { get; set; }

public MealInfo MealInfo { get; set; }

public Certificate CertificateInfo { get; set; }

public decimal TotalBookingCost { get; set; } = 0.0m;

}

public class BookSummary

{

public IList<Quotation> Summary = new List<Quotation>();

}

#endregion

}

**JSONs Used:**

**Events JSON**

[

{

"EventID":"1",

"EventName":"Murder Mystery Weekend",

"Days":"2",

"RegistrationFee":"600"

},

{

"EventID":"2",

"EventName":"CSI Weekend",

"Days":"3",

"RegistrationFee":"1000"

},

{

"EventID":"3",

"EventName":"The Great Outdoors",

"Days":"4",

"RegistrationFee":"1500"

},

{

"EventID":"4",

"EventName":"The Chase",

"Days":"6",

"RegistrationFee":"1800"

},

{

"EventID":"5",

"EventName":"Digital Refresh",

"Days":"2",

"RegistrationFee":"599"

}

]

**Event Location JSON**

[

{

"LocationID":"1",

"Location":"Cork",

"LodgingFee":"250"

},

{

"LocationID":"2",

"Location":"Dublin",

"LodgingFee":"165"

},

{

"LocationID":"3",

"Location":"Galway",

"LodgingFee":"225"

},

{

"LocationID":"4",

"Location":"Belmullet",

"LodgingFee":"305"

},

{

"LocationID":"5",

"Location":"Belfast",

"LodgingFee":"95"

}

]

**Meal Type JSON**

[

{

"MealID":"0",

"MealType":"No Meal",

"Price":"0"

},

{

"MealID":"1",

"MealType":"Full Board",

"Price":"49.50"

},

{

"MealID":"2",

"MealType":"Half Board",

"Price":"37.50"

},

{

"MealID":"3",

"MealType":"Breakfast Only",

"Price":"12"

}

]

**App.Config**

