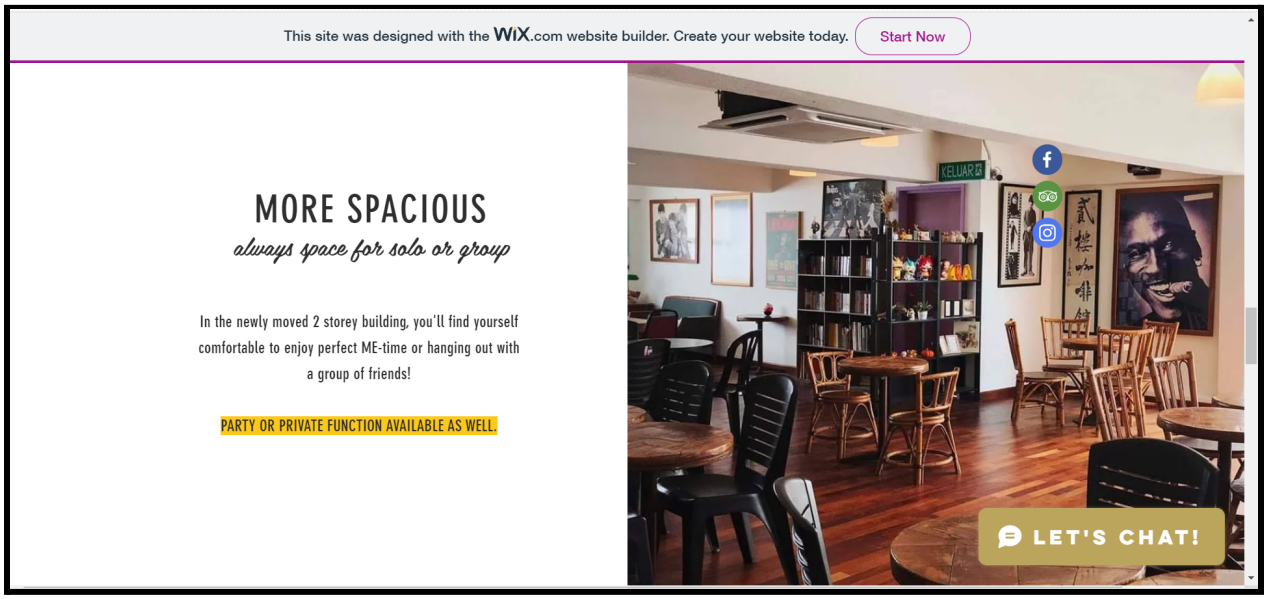
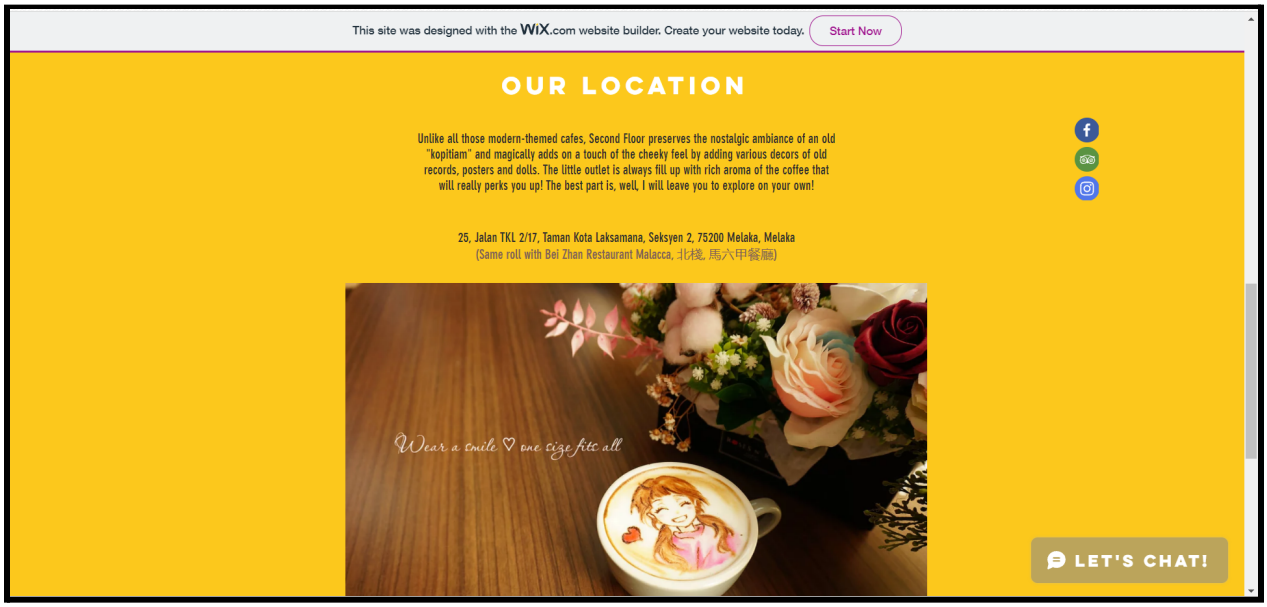
Chapter 1: Introduction to WAD

Static website: https://2ndfloorcoffeehouz.wixsite.com/2ndfloorcoffeehouse



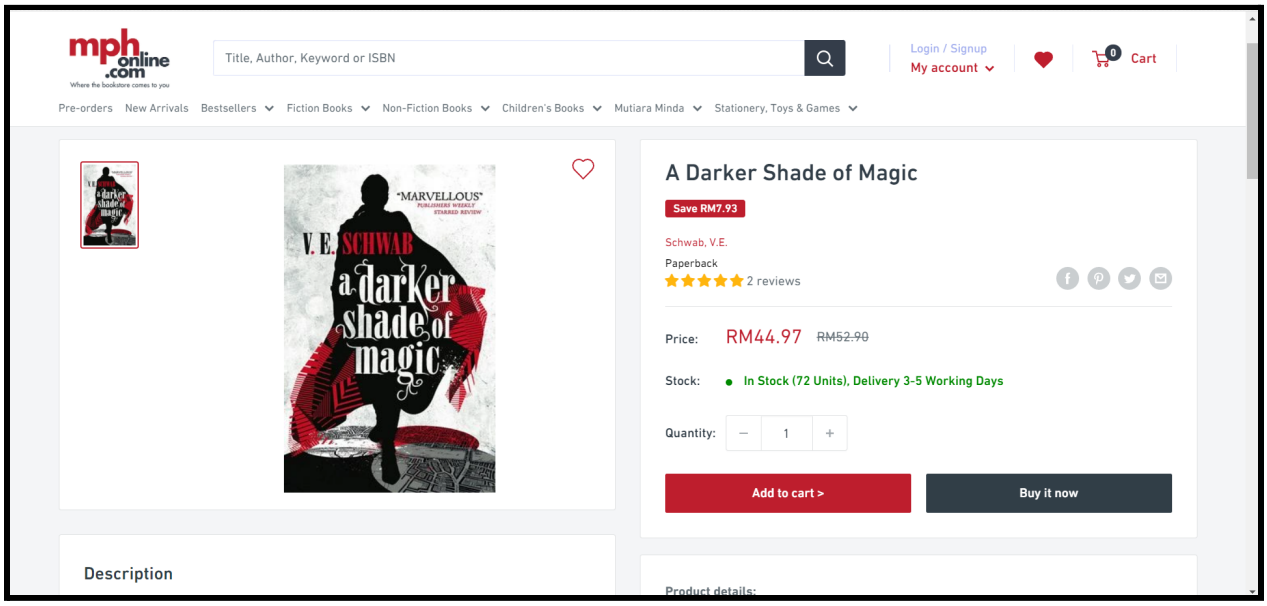
Our Story in Second Floor Cafe Website

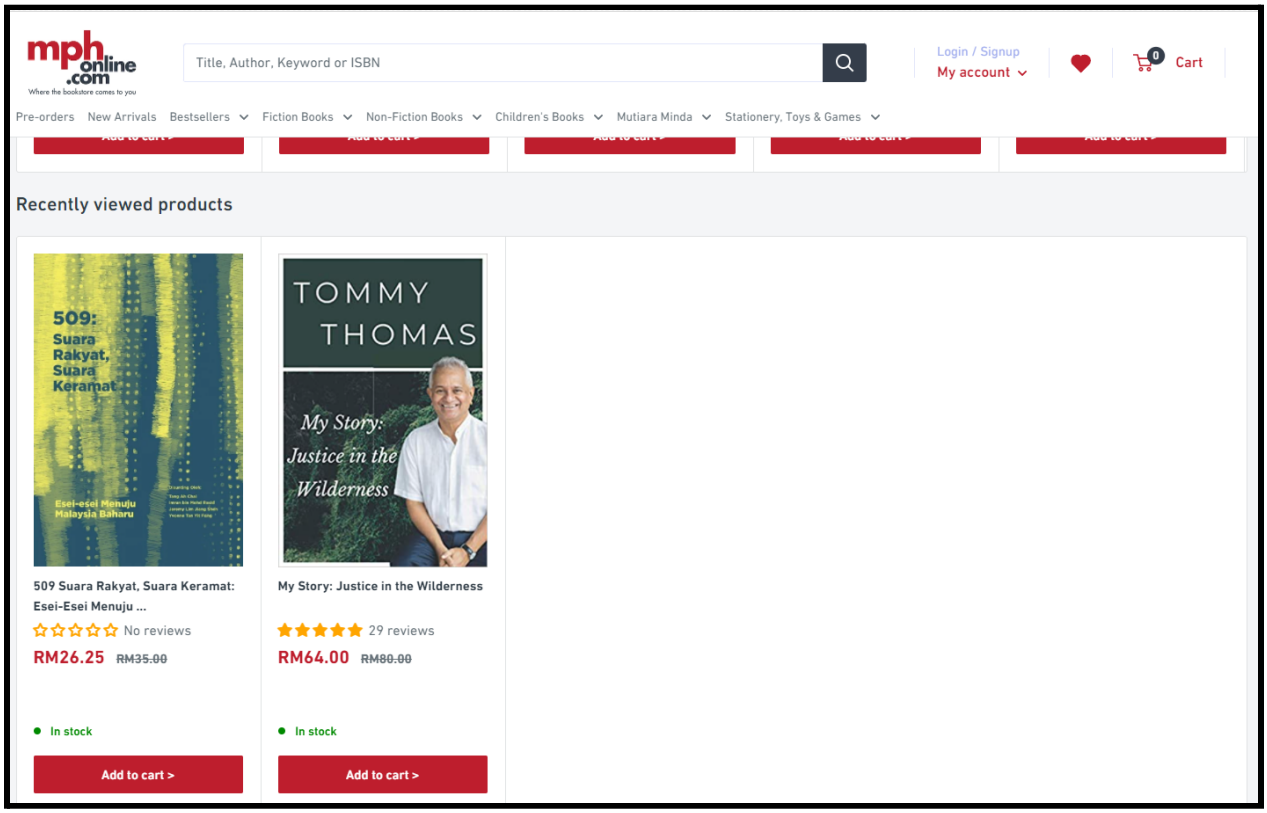


Our Location in Second Floor Cafe Website

Static page is a stable/constant content with the same text, images, hyperlinks that are displayed on a website regardless of any time. The appearance of the static page will always be the same which provides the same information to every visitor. It is simple to construct as it does not involve any database and only involves client-side HTML and CSS. Moreover, the static pages can be loaded quickly as it is pre-rendered and does not require a round trip to the server to fetch information from the database to generate dynamic content to be displayed on the web page.

Dynamic website: https://mphonline.com/





The appearance of the dynamic page will not always be the same as the web pages can be generated dynamically at run time. It involves client side and server side scripting such as JavaScript, PHP, C#, VB.NET, PHP, etc to be constructed. Moreover, it generates dynamic content which fetches the information from the database in real time. Hence, it is more powerful with

the features compare to a static web page such as user login, accept payments for e-commerce, update quantity for product, etc

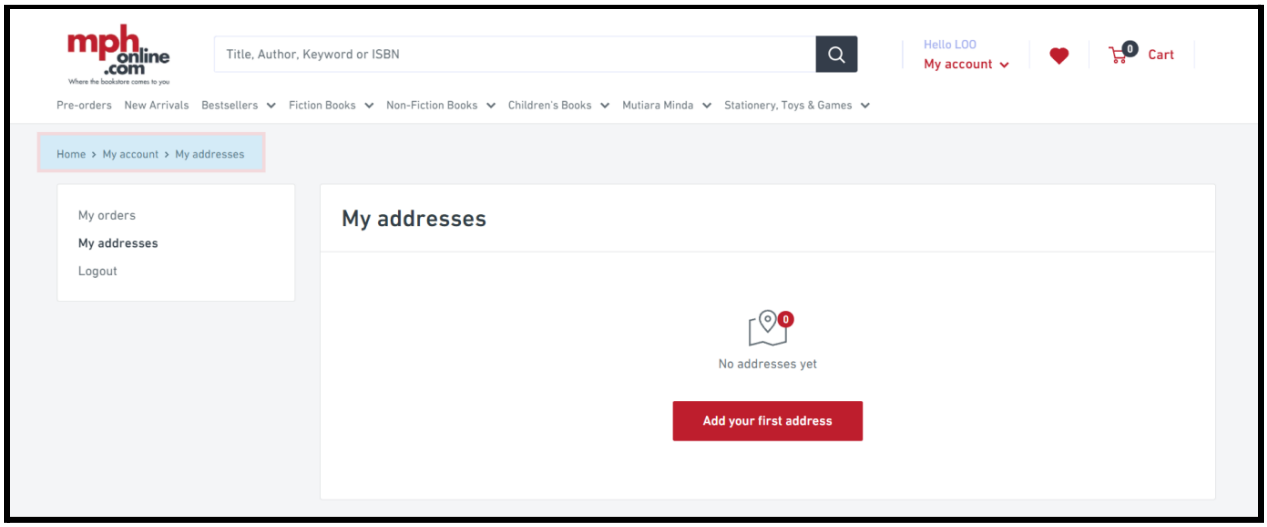
Dynamic page is better than static page because a dynamic page is more functional and interactive which can generate dynamic content based on user interaction and provide different useful features to the user such as user login and logout, accept payment for e-commerce, check order status whereas the static page only displays the same information all the time. Apart from that, content from dynamic pages are easier to maintain as they use efficient data management (database) and the developer can expand them with more functionalities in the future whereas the static pages are built for purely informational websites where the content in the static page is non-interactive and read-only and it can only be updated if the developer republish the file.

Client side scripting	**Server Side scripting**
Advantages	
Faster response time (i.e no need to go through server and process the information to return another HTML page to the browser)	Generate web pages dynamically based on user interaction (some actions triggered by user to create new page i.e. add a item to shopping cart)
Less overhead on the web server	Does not depend on browser as all the processing are performed in the server side
Ideal when the web page need to change without involve the database	Browser compatibility as the scripting is done on the server, it does not send back to the browser which prevent it from being copied thus more secure
I.e dynamically show and hide elements based on user inputs, input validation	
Disadvantages	
Not secure because anyone can look at (inspect) the code structure in the page source	Might slow at the time when the user is disconnecting from the network or the web hosting is down. It might take longer time to execute
Some of the browser does not supports scripting language (i.e JavaScript) very well	A database is required to store the dynamic data because they need to backup regularly

User can disable the client side checking	

Chapter 2: Server control and Site design

Sitemap: [MPHOnline | Malaysia's No. 1 Online Bookstore]



* Home> My account > My addresses is the sitemap (In blue background -- first image)

How sitemaps help increase the usability and accessibility of the web application?

- Sitemap provides an overall picture of the structure of the website which includes major sections and hyperlink in a serialized format. Hence, it improves the accessibility of a website which allows users to enter/access any section of the website that is listed in the sitemap instantly.
- Moreover, sitemap improves a website's usability by offering the visualization of navigation on a website. A good sitemap allows users to see all the available content on one page. If the website user interface is cluttered with content, a good sitemap probably will help the users from getting lost as the sitemap can help users to find information more easily which performs as a second navigation function on the website.

CIL and CLR -- Chapter 1

Compiler -> CIL (Common Intermediate language) -> CLR (common language runtime) -> machine code

Reason why choose this type of server control

TextBox

It is used to get the user input to be being processed further in the server code in c#

Label

Label is used because the element is being auto-generated which is only used for displaying the textual information to the user as an info and does not require any user input.

Check Box List

It allows users to select multiple options (zero, one or more) from the list of checkbox options.

Radio Button list

The Radio Button list is suitable when there are less than 5 options available to be selected. It allows users to select exactly one single item at a time from a list of options that are mutually exclusive.

Dropdown list

There is a group of items and the user can only select an item from the dropdown list predefined options which avoid any mistakes such as misspell or mistype and ensure the validity.

Calendar

It is used to display a single month calendar that can let the user click on the next ”>” button or previous button “<” to switch the month and choose the date he/she wants. This server control can ensure the date format entered by the user is correct such as 11/10/2021 before being stored inside the database.
(achieved using the calDepartDt.SelectedDate.ToShortDateString())

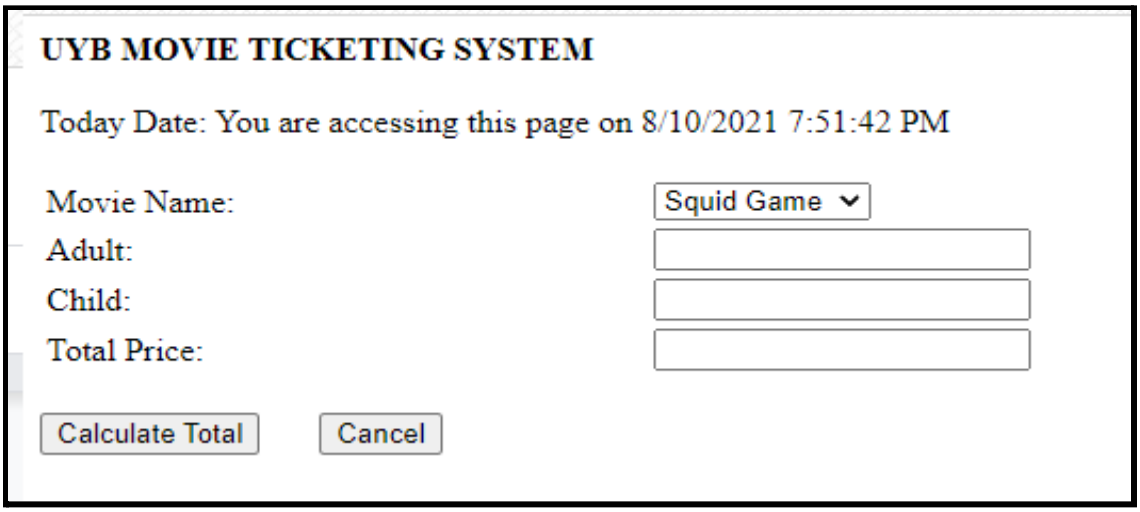
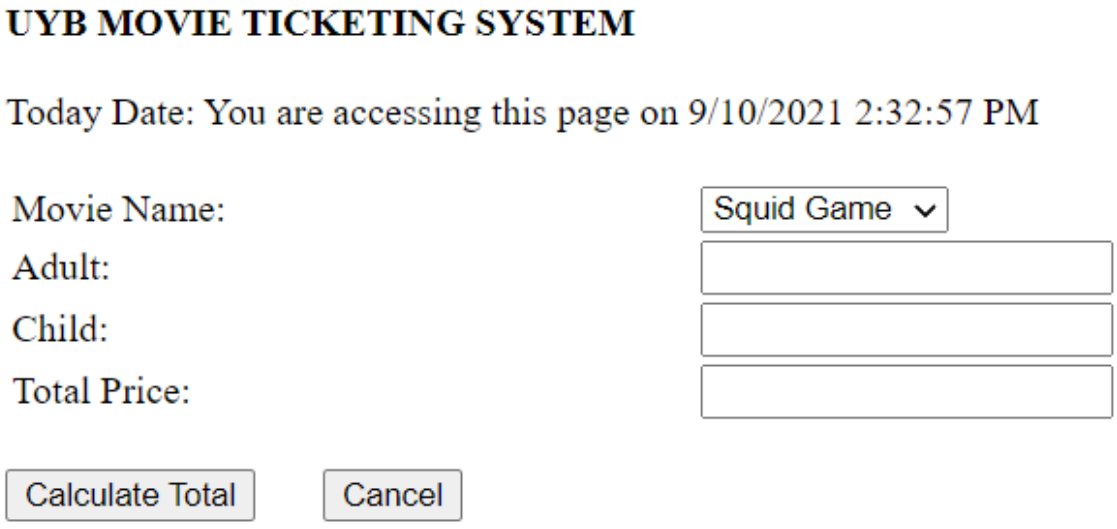
Button

It is used to perform either a client-side event to display a confirmation dialog before submitting the page or a server side event such as a click event to calculate the total payment and store it inside the database.

Chapter 3: Event-Driven Programming and Postback

3 categories of events:

- HTML event
- ASP.NET page level event
- ASP.NET server control event

<table>
<tr><th colspan="2">Event</th></tr>
<tr><td>Web form design</td><td> 
</td></tr>
<tr><td>Example of event</td><td>HTML Event
 

When mouse pointer move onto the Cancel button:</td></tr>
</table>

UYB MOVIE TICKETING SYSTEM

Today Date: You are accessing this page on 9/10/2021 2:32:57 PM

Movie Name: Squid Game

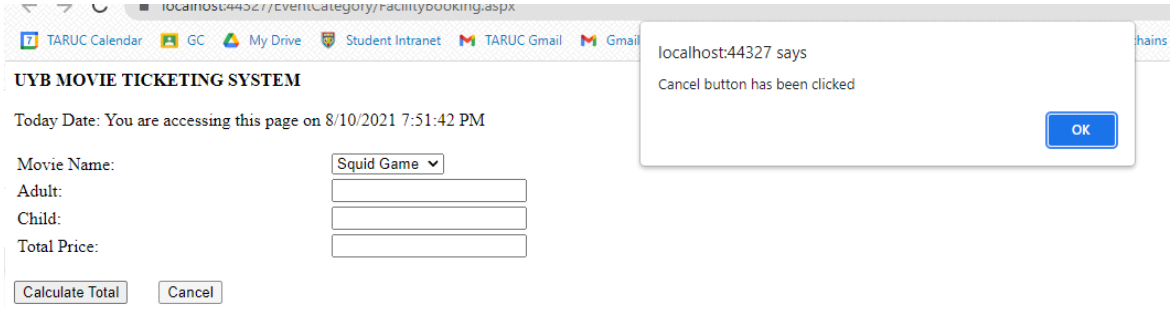
Adult:

Child:

Total Price:

Calculate Total | Click if you confirm to reset selection

When Cancel button has been clicked:



Code illustration:

```
        <asp:Button ID="btnCalculate" runat="server" Text="Calculate Total" OnClick="btnCalculate_Click" />
             
           <asp:Button ID="btnCancel" runat="server" OnClientClick="popMessage()"
            Text="Cancel" OnMouseOver="this.value='Click if you confirm to reset selection'"
            OnMouseOut="this.value='Cancel'" OnClick="btnCancel_Click" />
        <br />
        <br />
    </div>
</form>

<script>
    function popMessage() {
        alert("Cancel button has been clicked");
    }
</script>
```

Page Level Event

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        lblTime.Text = "You are accessing this page on " + DateTime.Now.ToString();
    }
}
```

	Server Control event
	`protected void btnCalculate_Click(object sender, EventArgs e)`
	`{`
	`  double adultPrice, childPrice, ttlPrice;`
	`  if (txtAdult.Text != "" && txtChild.Text != "")`
	`  {`
	`  if (!ddlMovie.Text.Equals(""))`
	`  {`
	`  adultPrice = double.Parse(txtAdult.Text) * 15;`
	`  childPrice = double.Parse(txtChild.Text) * 8;`
	`  ttlPrice = adultPrice + childPrice;`
	`  txtTotalPrice.Text = ttlPrice.ToString("C");`
	`  }`
	`  }`
	`}`
	
	Total Price:
	UYB MOVIE TICKETING SYSTEM
	Today Date: You are accessing this page on 8/10/2021 7:51:42 PM
	Movie Name: Squid Game
	Adult: 2
	Child: 1
	Total Price: RM38.00
	Calculate Total Cancel

Explanation of how each event works with code	**HTML Event (onClientClick, onmouseover, onmouseout event)** When hover the **cancel asp button**, it will trigger onMouseOver HTML event and change the text button to “Click if you confirm to reset selection” and if the user move the mouse pointer out of the cancel asp button, it will trigger onMouseOut HTML event and change the text back to “Cancel”. After that, when the **cancel asp button** has been clicked, it will trigger the event handler - JavaScript function popMessage() in the browser to handle the event which displays an alert box with a message (“Cancel button has been clicked”). **Page Level Event (page load event)** Everytime when the **page is loaded** which invoked Page.IsPostback property, it will trigger page load event and automatically update and display the latest datetime on the

	[lblTime] label **Server Control event (onclick event)** This server control event occurs when the user click on the **Calculate Total asp button**, it will then invoked the associate event handler - `protected void btnCalculate_Click(object sender, EventArgs e)` to handle the server control click event which calculate the total payment amount based on the total number of adult and children after the movie has been chosen.

Server Control Event

Server control	Event	Attribute
TextBox	**TextChanged** When a text is changed (text value change) in the TextBox control, it will trigger the event to handle the event. It will only occur when the "AutoPostBack" of the TextBox server control is set to true.	OnTextChanged
Dropdown list, radio button list, check box list	**SelectedIndexChanged** When an item is changed in the dropdown list, this event handler will be triggered. It will only occur when the "AutoPostBack" of the dropdown list server control is set to true.	OnSelectedIndexChanged
CheckBox, Radio button	**CheckedChanged** When the user clicks the checkbox to change the checked state, it will trigger this event handler to handle this click event of the checkbox. It will only occur when the "AutoPostBack" of the checkbox server control is set to true.	OnCheckedChanged
Button, image button, link button, image map	**Click** Button click event occurs when the button control has been clicked.	OnClick
Calendar	**SelectedChanged** It occurs whenever there is a change to a selection	

Chapter 4 Database Programming

Discuss the importance of choosing the correct data type and how it affects the functionality of a database using your own words

- Choosing the right data types can improve performance and data integrity which ensure the data is being stored inside the database in a correct format and value.
- If the size allocation of the data type is too large, it will waste the memory storage space.
- If the size allocation of the data type is too small, it will cause insufficient memory storage space for storing the data.
- If the data type chosen is not appropriate, it requires the developer to carry out data type conversion (i.e convert a string ‘9’ to an integer 9) which requires more work and time. Moreover, it also makes the reporting become more difficult and complex.

Chapter 5 Membership Management

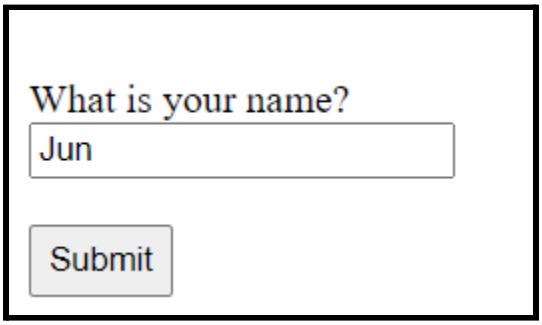
Authentication must be performed before authorization. Justify it.

- **Authentication** is the process of validating the user credentials such as email and password to verifying the identity of a user. The system or computer application will determine who you say you are by using the credentials you provided in the login form.

- After a user has been authenticated to the system, **authorization** process will be carried out to determine what user has been granted access to use the resource in the system. It is used to verify what the user is allowed to do in the system after the user has authenticated. For example, there are 2 roles which are customer and administrator which client does not have permission to access the resources/web pages under the webpages inside the admin folder.

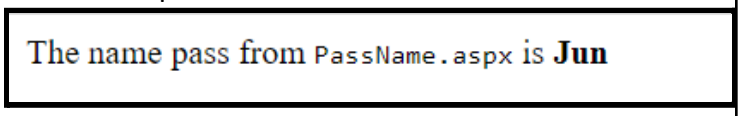
Chapter 6: State Management

<table>
<tr><th>Query String</th><th>Cookie</th></tr>
<tr><td>Query string is used to pass the information from one page to the target page. Only the target page will receive the value of the query string.</td><td>A cookie is used to store user specific data such as user preferences, login token, etc on a client's computer. All the pages within an application can retrieve the cookie value.</td></tr>
<tr><td>Real life example: URL from a web browser
https://localhost:44327/StateManagement/QueryString/GetName.aspx?name=Jun

The url redirects the user to GetName.aspx and pass 1 variables which is name=Jun</td><td>Real life example:
Website that store the user theme preferences of the client</td></tr>
<tr><td>PassName.aspx:
 

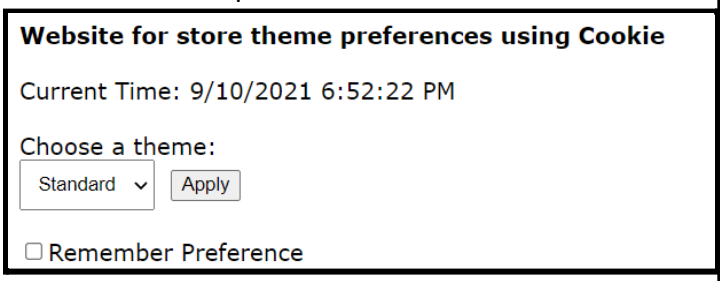
PassName.cs:

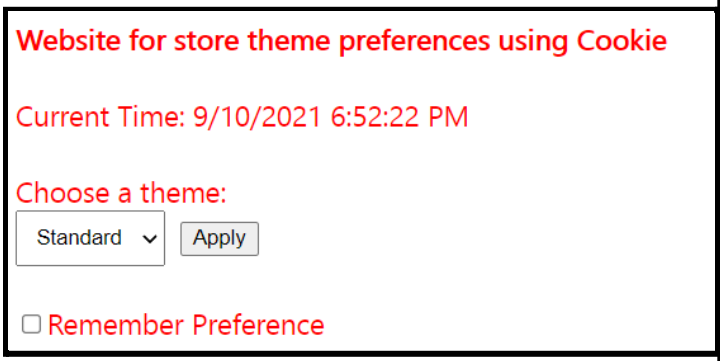
```
protected void btnSubmit_Click(object sender, EventArgs e)
{
    btnSubmit.PostBackUrl = "~/StateManagement/QueryString/GetName.aspx?name=" + txtName.Text;
}
```


GetName.aspx:
 

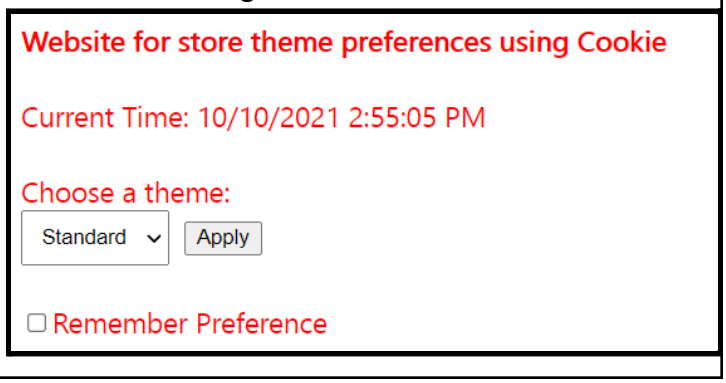
GetName.cs:

```
protected void Page_Load(object sender, EventArgs e)
{
    string name = Request.QueryString["name"];
    lblName.Text = name;
}
```


The PassName.aspx will accept the name input from TextBox and pass the “name” query string to the target page (GetName.aspx). Afterwards, the GetName.aspx</td><td>UserPreference.aspx
 

After user choose a theme called “Special.css” and click “Apply” button:
 


```
protected void chkPreference_CheckedChanged(object sender, EventArgs e)
{
    if (chkPreference.Checked)
    {
        HttpCookie cookie = new HttpCookie("theme", Session["SelectedCss"].ToString());
        Response.Cookies.Add(cookie);
        cookie.Expires = DateTime.Now.AddDays(30);
    }
}
```
</td></tr>
</table>

will retrieve the value of the name attribute and display it in the [lblName] label.	The UserPreference.aspx will let the user choose his/her favourite theme and apply it. If the user checks the *Remember Preference* checkbox, it will create a cookie called theme and store it for 30 days before expiring. Thus, the theme will be the Standard.css for 30 days when the user enters the UserPreference Page as shown below. 

Session Variable	**Application Variables**
Session variable is to store the object in any type about <u>one particular user</u> and available to all pages within the application. The session variable will persist until the session has expired or the user closes the website.	Application variable is to store the object in any types which are designed to maintain the state globally across all the pages in the entire website within an application. Application variables can be accessed by <u>all the users</u> of the application.
Real life example: Store the user specific information such as user id (i.e artist id) in a website	**Real life example:** Display tips of the day in a website
 After the user enters the login credentials which are	

```
SqlConnection con;
string strCon = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;

con = new SqlConnection(strCon);
con.Open();

string strSelect = "Select * from Artist Where email=@email AND password=@password";
SqlCommand cmdSelect = new SqlCommand(strSelect, con);
cmdSelect.Parameters.AddWithValue("@email", email);
cmdSelect.Parameters.AddWithValue("@password", password);

SqlDataReader dtrArtistInfo = cmdSelect.ExecuteReader();

if (dtrArtistInfo.HasRows)
{
    while (dtrArtistInfo.Read())
    {
        Session["artistID"] = dtrArtistInfo["ArtistID"];
        Response.Redirect("~/ArtistManagement/ArtistProfile/ArtistProfile.aspx");
    }
}
```

email and password, it will check the user authentication through the database. If the result is found, the artist ID will be stored inside the session variable and postback to the artist profile to use the Session[“artistID”] to retrieve the information of that particular artist and display it on the profile page.

```
protected void Page_Load(object sender, EventArgs e)
{
    lblTipsOfTheDay.Text = Application["TipsOfTheDay"].ToString();
}
```

```
protected void Application_Start(object sender, EventArgs e)
{
    Application["TipsOfTheDay"] = "Don't judge a book by it's cover";
}
```

In this UYB website, it will display the same tips of the day to all the users within this application. The application variable “Tips of the day” will be declared in the global.asax and a label called *[lblTipsOfTheDay]* will be declared and assigned the value from this application variable to be displayed to all the users that access this website.

Caching	**Fragment Caching**
Caching stores frequently used data in the memory so that when the next time the same information is needed to be displayed, it can directly retrieve from memory instead of requiring a server trip to generate the information.	Fragment caching allows part of a page to be cached which is achieved using user control within a web form.
Real life example: Displays a product summary of each product in a year	**Real life example:**
Output:	**Output:**

This product sales summary for each product in 1997 has been cached at **10/10/2021 3:49:15 PM** for **3600 seconds**

CategoryName	ProductName	ProductSales
Meat/Poultry	Alice Mutton	16580.8500
Condiments	Aniseed Syrup	1724.0000
Seafood	Boston Crab Meat	9796.3300
Dairy Products	Camembert Pierrot	20652.2800
Seafood	Carnarvon Tigers	15950.0000
Beverages	Chai	4887.0000
Beverages	Chang	7038.5500
Beverages	Chartreuse verte	4475.7000
Condiments	Chef Anton's Cajun Seasoning	5214.8800
Condiments	Chef Anton's Gumbo Mix	373.6300

1 2 3 4 5 6 7 8

**Assume there is a database involved called Northwind.mdf which can be found in the Practical 4 materials*

The gridview is a user control from WebUserControl.ascx
It will cache the WebUserControl.ascx for 3600 seconds

EmployeeID	LastName	FirstName	Title	HireDate
1	Davolio	Nancy	Sales Representative	1/5/1992 12:00:00 AM
2	Fuller	Andrew	Vice President, Sales	14/8/1992 12:00:00 AM
3	Leverling	Janet	Sales Representative	1/4/1992 12:00:00 AM
4	Peacock	Margaret	Sales Representative	3/5/1993 12:00:00 AM
5	Buchanan	Steven	Sales Manager	17/10/1993 12:00:00 AM
6	Suyama	Michael	Sales Representative	17/10/1993 12:00:00 AM
7	King	Robert	Sales Representative	2/1/1994 12:00:00 AM
8	Callahan	Laura	Inside Sales Coordinator	5/3/1994 12:00:00 AM
9	Dodsworth	Anne	Sales Representative	15/11/1994 12:00:00 AM

User control being cached at 10/10/2021 4:35:00 PM

WebUserControl.aspx:

Caching code:

```
<%@ OutputCache Duration="3600"
VaryByParam="none" %>
```

Other code:

```
<asp:SqlDataSource ID="SqlDataSource1"
runat="server" ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
SelectCommand="SELECT * FROM [Product Sales
for 1997]"></asp:SqlDataSource>
```

to get all the product sales information and bind to the GridView data bound control.

After we run the OutputCaching.aspx page, it will display the grid view of the product sales for each product in a table format and it will be stored inside the cache for 3600 seconds to eliminate the time needed to regenerate again the next time it's requested. Thus, it can reduce the load from the database and speed up the accessing time thus improving the application's performance.

FragmentCaching.aspx

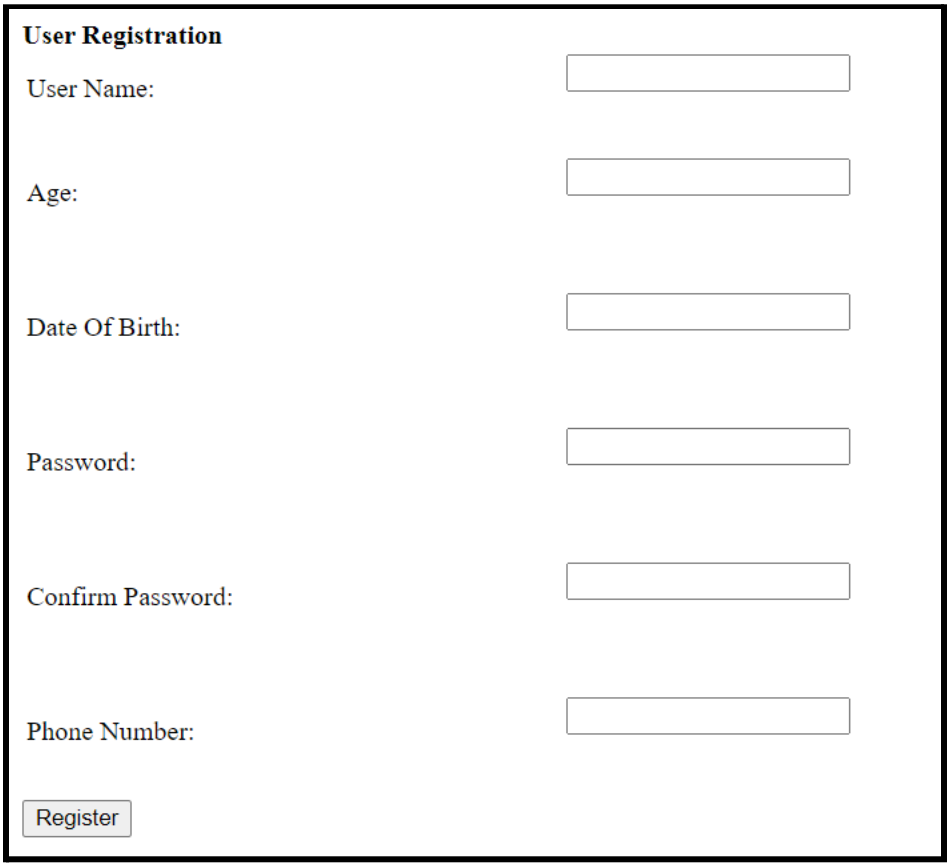
The caching will be done in the WebUserControl.ascx which will cache part of the FragmentCaching.aspx for 3600 seconds so that the user control no need to regenerate again when each time it's being requested.

Chapter 7: Validation Control

5 types of validation control:

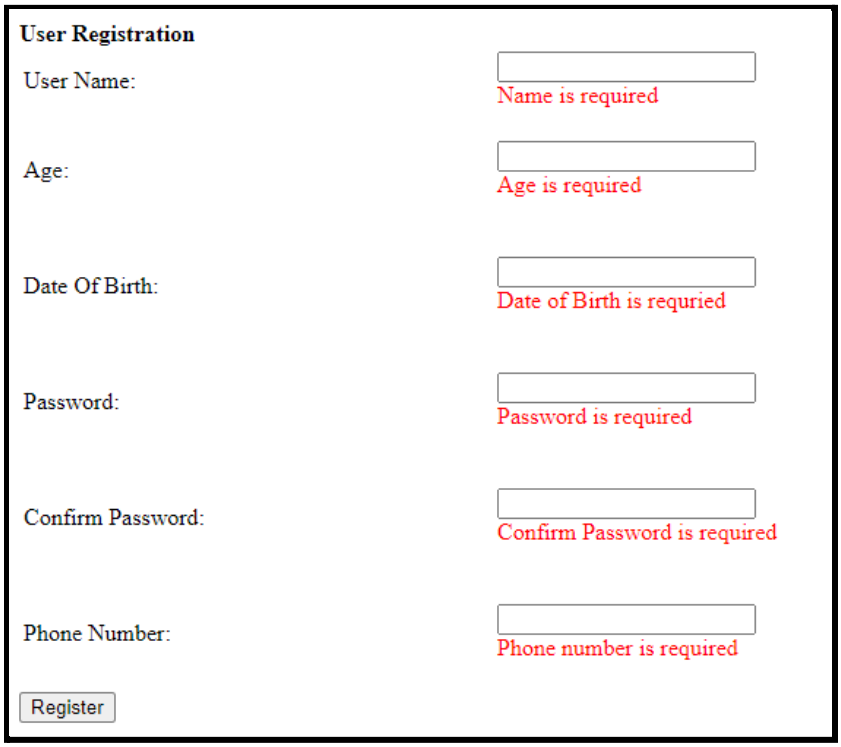
1. Required Field Validator
2. Range Validator
3. Compare Validator
4. RegularExpression Validator
5. Custom Validator

Web form design



Explanation:

RequiredFieldValidator



The **RequiredFieldValidator** is to ensure the user enters all the fields which are username, age, date of birth, password, confirm password and phone number before registering an account. To illustrate, if one of the fields is left empty, it will display the error message (i.e Name is required, etc) to inform the users so that they will fill in all the fields.

Validator code (2 examples):

- `<asp:RequiredFieldValidator ID="rfvName" runat="server" ControlToValidate="txtName" ErrorMessage="Name is required" ForeColor="Red"></asp:RequiredFieldValidator>`
- `<asp:RequiredFieldValidator ID="rfvAge" runat="server" ControlToValidate="txtAge" ErrorMessage="Age is required" ForeColor="Red"></asp:RequiredFieldValidator>`

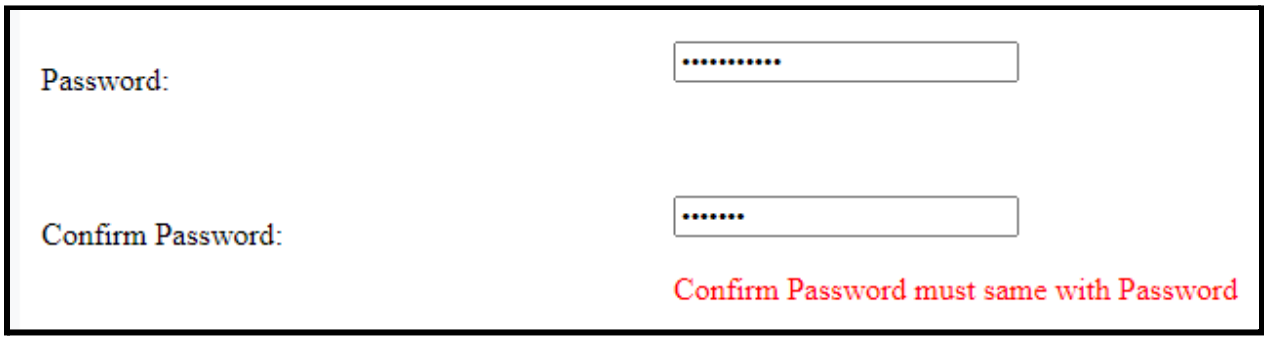
RangeValidator

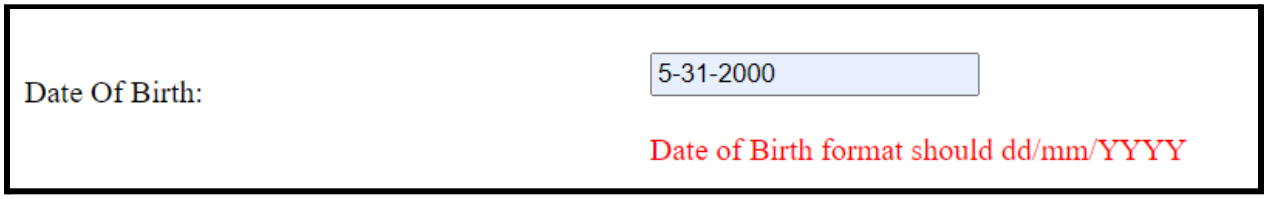
The **RangeValidator** will ensure the value entered is within the valid range. To illustrate, when the age entered is out of the range (i.e. less than 0 or greater than 99), it will display an error message which is “Age must between 1-99” to inform the user.

Validator code:

```
<asp:RangeValidator ID="rvAge" runat="server" ControlToValidate="txtAge"
ErrorMessage="Age must between 1 - 99" ForeColor="Red" MaximumValue="99"
MinimumValue="1"></asp:RangeValidator>
```

compareValidator





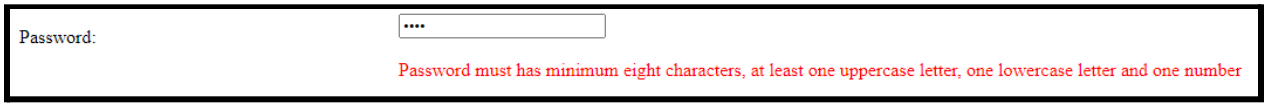
The **compareValidator** ensures that the entered value matches with another control as well as for checking data type. To illustrate, if the confirmed password entered is not the same as the password, it will display the error message which is “Confirm Password must be the same as Password ''. Another example will be to compare the data type (Date) in dd/mm/YYYY format; otherwise it will display an error message to the user.

Validator code:

- ```
  <asp:CompareValidator ID="cvCPassword" runat="server"
  ControlToCompare="txtPassword" ControlToValidate="txtCPassword"
  ErrorMessage="Confirm Password must same with Password"
  ForeColor="Red"></asp:CompareValidator>
  ```
- ```
  <asp:CompareValidator ID="cvDOB" runat="server"
  ControlToValidate="txtDOB" ErrorMessage="Date of Birth format should
  ```

```
dd/mm/YYYY" ForeColor="Red" Operator="DataTypeCheck"
Type="Date"></asp:CompareValidator>
```

RegularExpressionValidator



The **RegularExpressionValidator** will check whether the entered value matches with another control defined by a regular expression. To illustrate, if the password entered is not matched with the regular expression that is defined in the RegularExpressionValidator which is "^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$", it will display the error message so that the user can reenter the password in the correct format.

Validator Code:

```
<asp:RegularExpressionValidator ID="revPassword" runat="server"
ControlToValidate="txtPassword" ErrorMessage="Password must has minimum eight
characters, at least one uppercase letter, one lowercase letter and one number"
ForeColor="Red"
ValidationExpression="^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$"
ViewStateMode="Enabled"></asp:RegularExpressionValidator>
```

CustomValidator

```
function ClientValidatePhoneNumber() {
    var phoneNumber = document.getElementById("<%=txtPhoneNumber.ClientID%>").value
    phoneRegex1 = /^\(?([0-9]{3})\)?[-. ]?([0-9]{3})[-. ]?([0-9]{4})$/;
    phoneRegex2 = /^\d{10}$/;

    if (String(phoneNumber).match(phoneRegex1) || (String(phoneNumber).match(phoneRegex2))) {
        alert("Phone number is valid");
    } else {
        alert("Phone number is invalid");
    }
}
```

The **CustomValidator** allows the developer to define the custom codes to validate the entered value. In this case, the ClientValidatePhoneNumber() client function has been written to validate the phone number. If the phone number does not match with either phoneRegex1 or phoneRegex2, it will display an alert box with the message "Phone number is invalid"; otherwise a valid message ("Phone number is valid") will be displayed.

Validator Code:

```
<asp:CustomValidator ID="cvPhoneNumber" runat="server"
ClientValidationFunction="ClientValidatePhoneNumber"
ControlToValidate="txtPhoneNumber" ErrorMessage="CustomValidator"
ForeColor="Red" EnableClientScript="true"></asp:CustomValidator>
```

Importance of validation

- Validation is important as it can prevent invalid data from being entered in the web form to ensure the input provided by the user is in a correct format to be further processed by the server to generate meaningful output.

- Client side validation is convenient as the user can get instant feedback as the page doesn't need to postback to the server (round trip to the server) but it needs to depend on the scripting language support and browser.

- Server side validation is browser compatibility which does not depend on the browser and scripting language. Therefore, it ensures the server validation would be performed to ensure the correct data has been received before being processed by the server.

- Developers prefer server side validation because it is more secure which the developer can protect against the malicious user who can easily disable the client side validation and submit dangerous input to the server.

Chapter 8: User control

User control	Master Page
Similarity	
Both user control and master page are used to reduce code and enhance reusability	
Differences	
User control can be coded for the same content by wrapping up a small bundle of functionality with same programming logic that can be reused on multiple ASP.NET web pages Separate basic functionality from the main view which can be reused It can be drag and drop in every asp net page when the developer needs it.	Master page is a template that provides a consistent layout and shares common behaviour for multiple pages within an application. The essential elements such as header and footer will be presented on every web page within the application.
It supports fragment caching	It does not support any caching method
For example, there are 2 different groups of users such as admin and normal staff who can login to the website but both of them will see the different navigation menu after login.	For example, a common header with the menu and footers that provide copyright information of an organization will be presented on every web page within the application.

Under what circumstances a user control should be used instead of master page?

When some of the content in an ASP.NET page can be reused for multiple times which require slight changes such as the value of the properties and method defined. For instance, the user credentials Textbox (username and password) can be coded in a loginControl.ascx which can be placed in customer, admin and staff login ASP.NET pages where each of the textbox value in the user control will be passed to the cs server code accordingly for further processing. This saves time on developing the code and reduces the amount of code per page which can improve the efficiency of the code structure.

Adding a user control on a page

1. <%@ Register TagPrefix=”SuperCompany” TagName=”Header” Src=”Simple.aspx” />
2. <SuperCompany:Header ID=”ctlHeader” runat=”server” />

Custom Control	User Control
A loosely coupled control w.r.t code and UI	A tightly coupled control w.r.t code and UI
Derives from `Control`	Derives from `UserControl`
Defines UI in a `ResourceDictionary`	Defines UI as normal XAML
UI is skinable	Child controls are skinable
Has dynamic layout	Has static layout
UI can be changed in different projects	UI is fixed and can't have different looks in different project
Has full toolbox support	Can't be added to the toolbox
Defines a single control	Defines a set of controls
More flexible	Not very flexible like a Custom Control
Requires in-depth knowledge of Silverlight UI Model	Does not require indepth knowledge of the UI Model

When to Use?

Good question: "When to use a Custom Control?" Haven't you got the inner meaning of it yet? OK, read the summarized points below:

- When you have a rapid and fixed content in your UI, use `UserControl`.
- When you want to separate some basic functionality of your main view to some smaller pieces with reusability, use `UserControl`.
- When you want to use your control in different projects and each project may want to change the look, use `CustomControl`.
- When you want to implement some additional functionality for a control, create a `CustomControl` derived from the base control.
- When you want to apply themes to your controls, use `CustomControl`.
- When you want to add toolbox support for your control, so that your user will be able to do drag and drop to the designer, use `CustomControl`.

Chapter 9 Debugging and error handling

4 types of the error

- Parse error
- Compilation error
- Configuration error
- Runtime error/ logic error

Examples of the **runtime error**:

- Invalid query string
- Server failed
- Attempt to open a database connection which was not closed previously
- Out of bounds
- Null pointer exception

Examples of the **logic error**:

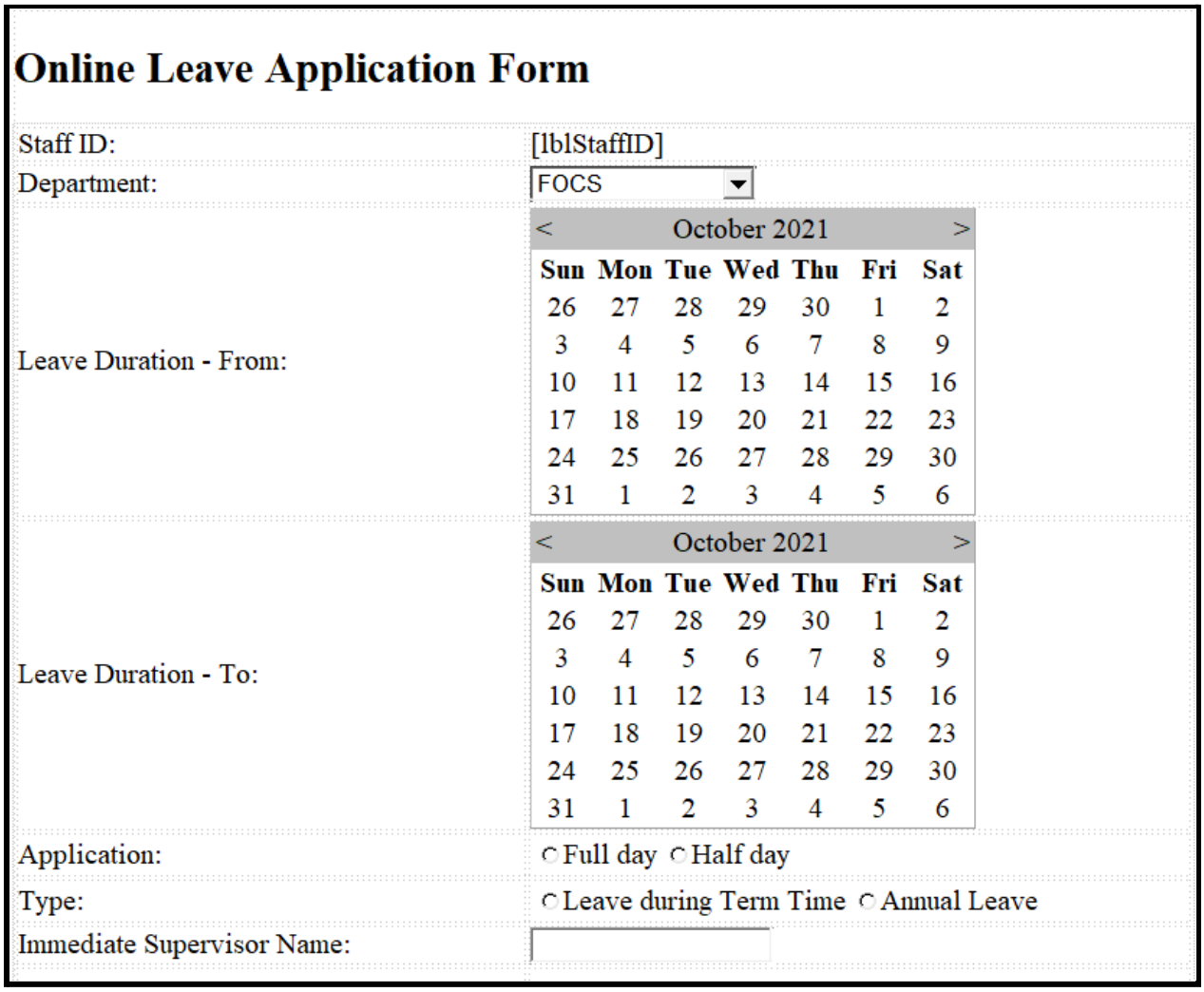
- Divided by zero
- Parsing problem

The importance of error handling. Justify it.

- Provide pleasant user experience when unexpected failure occurs within the application as the error handling methods will resolve the exception so that the application can continue or terminate gracefully without crashing.
- Error handling enables the developer to maintain the code more easily and create a application/system that are more fault-tolerant and robust
- The error message displayed when unexpected failure occurs is usually technical in which the end user may not understand. By using error handling in the application, the developer can display the human-readable, constructive messages to the end user which are more user friendly and able to suggest a solution to the user.

May 2021

a)



Staff ID: Label is used because the staff ID is being auto-generated which is only used for displaying its ID information to the user and does not require any user input.

Department: Dropdown list is used because there is a group of departments and the user can only select one department from the dropdown list predefined options which can avoid any mistakes such as misspell or mistype.

Leave Duration - From:

Calendar server control is used to display a single month calendar that can let the user click on the next ”>” button or previous button “<” to switch the month and choose the date he/she wants. This server control can ensure the date format of the leave duration - from selected by the user is correct such as 11/10/2021 before being stored inside the database.

Leave Duration - To:
Calendar server control is used to display a single month calendar that can let the user click on the next ”>” button or previous button “<” to switch the month and choose the date he/she wants. This server control can ensure the date format of the leave duration - to be selected by the user is correct such as 12/10/2021 before being stored inside the database.

Application: Radio button list is used because it allows users to select exactly one single item at a time from a list of two options (radio button list will be used when there are less than or equal to 5 options) which are full day or half day that are mutually exclusive.

Type: Radio button list is used because it allows users to select exactly one single item at a time from a list of two options (radio button list will be used when there are less than or equal to 5 options) which are leaving during term time or annual leave that are mutually exclusive.

Immediate supervisor name: TextBox is used because it is used to get the user input which is the supervisor name to be being processed further in the server code in c#

Dynamic web pages is more suitable as it can provide more functionality and powerful than the static web pages. To illustrate, by using dynamic pages, it can generate dynamic content based on user interaction. For example, when the user had enter and select all the required fields and click submit, the leave application record will be stored inside the database which will then retrieved by the staff's computer from the database. The staff can then approve the leave application and update the record status in the database to be viewed by the applicant. Therefore, a dynamic page is more appropriate as it requires a database, client-side scripting as well as server side scripting to write sql statements and update the record.