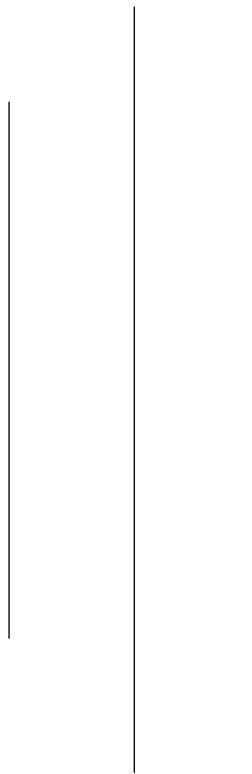




**Everest Engineering College  
Affiliated to Pokhara University  
Sanepa-2, Lalitpur, Nepal**



**Lab Manual On Web Technology**

**By:**

**Er. Santosh Bist**

**Computer and Information Technology Department**

**Everest Engineering College**

**sanepa-2, Lalitpur, Nepal**

## Guide Lines

- Web Technology is based on Design and Development of Website.
- This Manual is useful only for HTML, CSS , JavaScript and PHP.
- Students must read and practice as much as you can .
- In this manual there are all together **Twelve Lab** where **eight** of them include Assignment and are named as **Project** too.
- Those projects must be submitted on week after that lab.
- Marks Distribution out of 20:
  - Attendance – 5 Marks
  - Projects – 5 Marks
  - Viva and Lab Exam – 10 Marks

## Requirements:

Including your PC of any platform ,following tools are needed.

- Text Editor – is used to edit plain text(program). Example: Notepad, Sublime Text, Visual Studio Code, Atom etc.
- Browser – is used to browse web pages content that make interact to user. Example: Chrome, Firefox, Uc-Browser etc.
- Server – is used to provide services. Example: xampp, wampp etc.

**Note:** Here I used Sublime Text and Notepad Text Editor,Google Chrome Browser and XAMPP local server.

## Table of Contents

<b>Lab 1: Introduction and Basic Structure of HTML</b>	<b>1</b>
<b>Lab 2: HTML Lists, Table and Frame</b>	<b>4</b>
1. Unordered Lists:	4
2. Ordered Lists:	4
3. Definition Lists:	4
<b>Lab 3:HTML Form and Image Mapping</b>	<b>10</b>
Forms	10
<b>Lab 4: Introduction to Cascading Style Sheets(CSS)</b>	<b>13</b>
Introduction and Levels of style sheets	13
CSS Syntax	13
CSS Selectors	14
1. The element selectors	14
2. Class selectors	14
3. Id selectors	14
4. Universal selector	15
5. Pseudo Classes	15
<b>Lab 5: How to use CSS properties and value to HTML elements.</b>	<b>16</b>
CSS Backgrounds	16
CSS Colors	17
CSS Text Properties	17
CSS max-width property	18
CSS Layout – The <i>position</i> Property	18
<b>Lab 6: Designing webpage Layout(Including Header, Navigation Bar, content , Footer) using CSS</b>	<b>23</b>
Box Model:	23
Outline:	24
Float:	25
<b>Lab 7: Introduction to JavaScript</b>	<b>27</b>
JavaScript Variable:	27
JavaScript Function:	27
JavaScript Objects:	27
JavaScript Array:	27
JavaScript Loop:	27
<b>Lab 8: Event handling and <i>this</i> keyword in JavaScript</b>	<b>31</b>
<i>this</i> keyword	31

<b>Event and Event Handling</b> .....	31
<b>Lab 9: Prompt, Display and Visibility and Regular Expression in JavaScript</b> .....	36
<b>Popup Boxes:</b> .....	36
<b>Display and Visibility</b> .....	37
<b>Regular Expression(Pattern Matching)</b> .....	37
<b>Lab 10: Introduction to PHP</b> .....	40
<b>Introduction:</b> .....	40
<b>Variable:</b> .....	40
<b>Data types:</b> .....	40
<b>PHP Arrays:</b> .....	40
<b>PHP Function:</b> .....	42
<b>Lab 11: String Function, <i>each()</i> function, Form Handling and Cookies and Session</b> .....	43
<b>String Function in PHP:</b> .....	43
<b><i>each()</i> Function in PHP</b> .....	43
<b>Form Handling in PHP</b> .....	43
<b>Cookies and Session in PHP</b> .....	44
<b>Session in PHP</b> .....	45
<b>Lab 12: MySQL in PHP</b> .....	46

## Lab 1: Introduction and Basic Structure of HTML

**Objective:** To be Familiar with Basic Structure and Some basic elements in Html.

### Theory:

#### What is Web Technology?

Web Technology refers the methods that communicate computers with each other through the use of markup language and multimedia packages.

#### What is Markup Language?

A markup language is a computer language that uses tags to define elements within document. It is human-readable and markup files contain standard words , rather than typical programming syntax.

#### What do you mean by multimedia packages?

Multimedia is content that uses a combination of different content forms such as text, audio, images, animations, video and interactive content.

#### What do you mean by HTML?

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web.

#### Basic Syntax:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Basic structure of HTML</title>
  </head>
  <body>
    <h1>First Heading</h1>
    <p>This tag is used for paragraph.</p>
  </body>
</html>
```

- The <!DOCTYPE html>declaration defines this document to be HTML5
- The <html>element is the root element of an HTML page
- The <head>element contains meta information about the document
- The <title>element specifies a title for the document
- The <body>element contains the visible page content
- The <h1>element defines a large heading
- The <p>element defines a paragraph

#### Basic Text Formatting

- <b>- element defines **bold** text, without any extra importance.
- <strong>- element defines **strong** text, with added semantic "strong" importance.
- <i>- element defines *italic* text, without any extra importance.
- <em>- element defines *emphasized* text, with added semantic importance.
- <mark>- element defines marked or highlighted text.
- <small>- element defines smaller text .
- <del>- element defines ~~deleted~~ (removed) text.

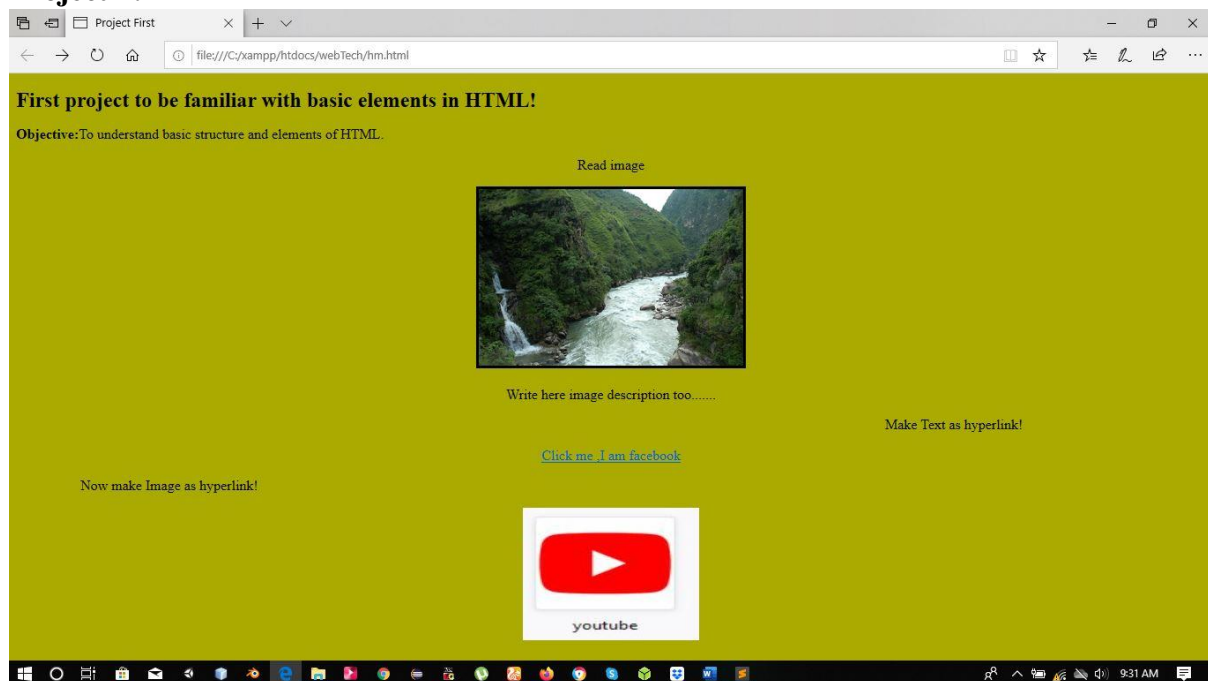
- `<ins>`- element defines inserted (added) text.
- `<sub>`- element defines subscripted text.
- `<sup>`- element defines superscripted text.

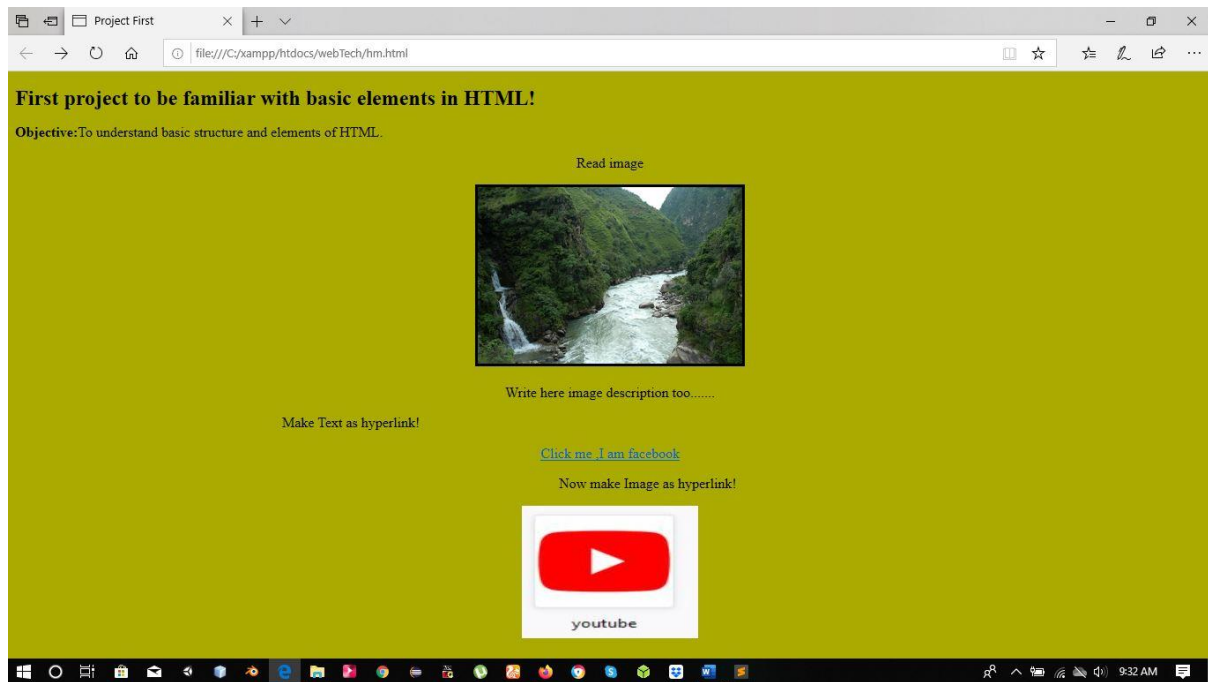
### Example:

```
<!DOCTYPE html>
<html>
<head>
    <title>Basic Elements</title>
</head>
<body bgcolor="grey">
    <b>This text is bold</b><br/>
    <strong>This text is strong</strong>
    <i>This text is italic</i>
    <em>This text is emphasized</em>
    <h2>HTML <small>Small</small> Formatting</h2>
    <h2>HTML <mark>Marked</mark> Formatting</h2>
    <p>My favorite color is <del>blue</del> red.</p>
    <p>My favorite <ins>color</ins> is red.</p>
    <p>This is <sub>subscripted</sub> text.</p>
    <p>This is <sup>superscripted</sup> text.</p>
    <marquee Onmouseover="this.stop()" Onmouseout="this.start()" bgcolor="blue" width="500"
height="200" direction="right" scrollamount="10" > hello</marquee>
    <marquee scrolldelay="400" bgcolor="yellow" hspace="200" vspace="20" ><font color="red">
namaste</font></marquee><hr width="500">
    <image src="karnali.jpg" alt="Karnali Image" height="200" width="200"></image>
<a href="https://www.gmail.com" title="Go to gmail" target="_blank">visit for mail to gmail</a>
</body>
</html>
```

**Assignment:** The following assignment is named as Project 1 so you need to complete it and present your output on next week.

### Project 1:





### Description:

- All above content are same as they appear but some have changes.
- First , describe about image you use(example: I used karnali river image) , at section “Write here image description too.....” .
- “Make Text as hyper link!” and “Now make Image as hyperlink” are defined with in marquee element , where first one move from right-left and second one from left-right.
- Section “Click me, I am facebook” is the text hyperlink having title “Facebook” and “Image of youtube ” is the image hyperlink having title “This image link to YOUTUBE”.
- Both hyperlink must open with new window , when clicking on that link.

## Lab 2: HTML Lists, Table and Frame

**Objective:** To understand Lists, Table and Frame in HTML

**Theory:**

### Lists:

- There are mainly three sorts of html lists.( Unordered Lists, Ordered List and Definition Lists).

#### 1. Unordered Lists:

- The `<ul>` tag , which is a block tag , creates an unordered list.
- Each item in a list is specified with an `<li>` tag.
- Any tag can appear in list item , including nested lists with in the `<ul>` tag.
- Items are displayed , each list item is implicitly preceded by a bullet.
- Also we can change type of Unordered list .( .i.e. *type=square or circle or disc or none*)

#### 2. Ordered Lists:

- The `<ol>` tag , which is a block tag , creates an ordered list.
- Each item in a list is specified with an `<li>` tag.
- Any tag can appear in list item , including nested lists with in the `<ol>` tag.
- Items are displayed , each list item is implicitly preceded by a numbers(1, 2, 3....).
- Also we can change type of Unordered list .( .i.e. *type= i / a/A/I*)

#### 3. Definition Lists:

- Definition lists are used to specify lists of terms and their definitions.
- A definition list is given as the content of a `dl` element , which is a block element.
- Each term to be defined in the definition list is given as the content of a `dt` element.
- Definitions themselves are specified as the content of `dd` elements .

### Example: Unordered lists , Ordered lists and Definition lists.

```
<!DOCTYPE html>
<html>
  <head>
    <title>About Lists</title>
  </head>
  <body>
    <h2>Unordered List with Bullets(as default)</h2>
    <p>You can change type as needed. Unordered types (square , circle , disc and none)</p>
    <ul type="">
      <li>Coffee</li>
      <li>Tea</li>
      <li>Milk</li>
    </ul>

    <h2>Ordered List with Numbers(as default)</h2>
    <p>You can change type as needed.</p>
    <ol type="i">
      <li>Coffee</li>
      <li>Tea</li>
      <li>Milk</li>
    </ol>
```



```

</ol>
<h3>Definition Lists</h3>
<dl>
    <dt>C</dt>
    <dd>Procedure Programming language</dd>
    <dt>C++</dt>
    <dd>Object Oriented programming language</dd>
</dl>
</body>
</html>

```

### Table:

- Tables provide a highly effective way of presenting many kinds of information.
- A table is a matrix of cells.
- The cells in the top row often contain column labels , those in the leftmost column often contain row labels , and most of the rest of the cells contain the data of the table.
- The content of a cell can be almost any document element , including text, a heading , a horizontal rule , an image , or a nested table.
- An HTML table is defined with the `<table>` tag.
- Each table row is defined with the `<tr>` tag.
- A table header is defined with the `<th>` tag.
- By default, table headings are bold and centered.
- A table data/cell is defined with the `<td>` tag.

### Example:

```

<!DOCTYPE html>
<html>
<head>
    <title>Creating table</title>
    <style>
        table, th, td {
            border: 1px solid black;
            border-collapse: collapse;
        }
        th, td {
            padding: 15px;
        }
        th {
            text-align: left;
        }
    </style>
</head>
<body>
    <h2>Basic HTML Table</h2>
    <table width="400">
        <tr>
            <th>Firstname</th>
            <th>Lastname</th>
            <th>Age</th>
        </tr>
        <tr>
            <td>Ram</td>
            <td>Saud</td>

```

```
 <td>50</td> </tr> <tr>  <td>Shyam</td> <td>Chaudhary</td> <td>94</td> </tr> <tr>  <td>Bharat</td> <td>Rokaya</td> <td>80</td> </tr> </table> <center> <table> <tr>  <th>C</th>  <th>C++</th>  <th>Web</th> </tr> <tr>  <td>1</td>  <td>Hari</td>  <td>50</td>  <td>60</td>  <td>90</td> </tr> <tr>  <td>2</td>  <td>Ram</td>  <td>50</td>  <td>60</td>  <td>90</td> </tr> </table> </center> </body> </html> | | | | | | | | | | | | | | | | | |
```

- Use the CSS border-spacing property to set the spacing between cells
- Use the colspan attribute to make a cell span to many columns
- Use the rowspan attribute to make a cell span to many rows

#### Frame:

- The <frame> tag is not supported in HTML5.
- Instead of using <frame> tag in HTML5 we use <iframe> tag.
- <iframe> tag is used to define inline frame.

#### Example:

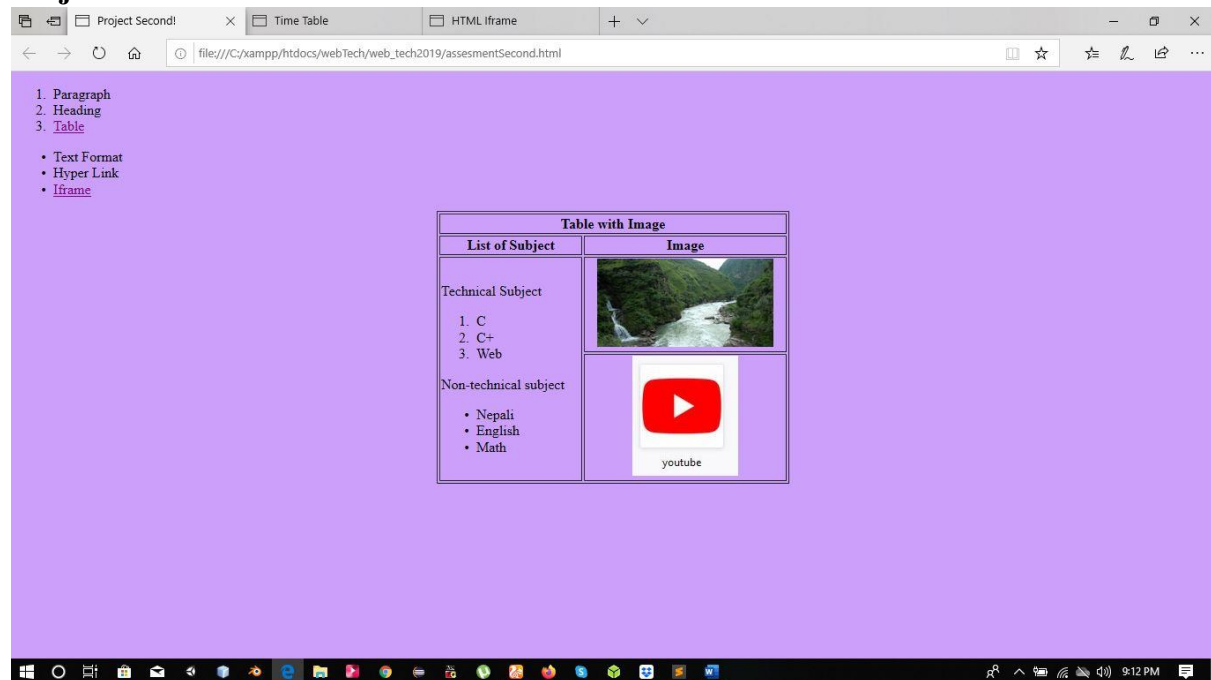
```

<!DOCTYPE html>
<html>
<head>
    <title>HTML Iframe</title>
</head>
<body bgcolor="#aaafff">
    <iframe src="karnali.jpg" height="200" width="400">
        <p>This is not support on your browser</p>
    </iframe>
</body>
</html>

```

**Assignment:** The following assignment is named as Project 2 so you need to complete it and present your output on next week.

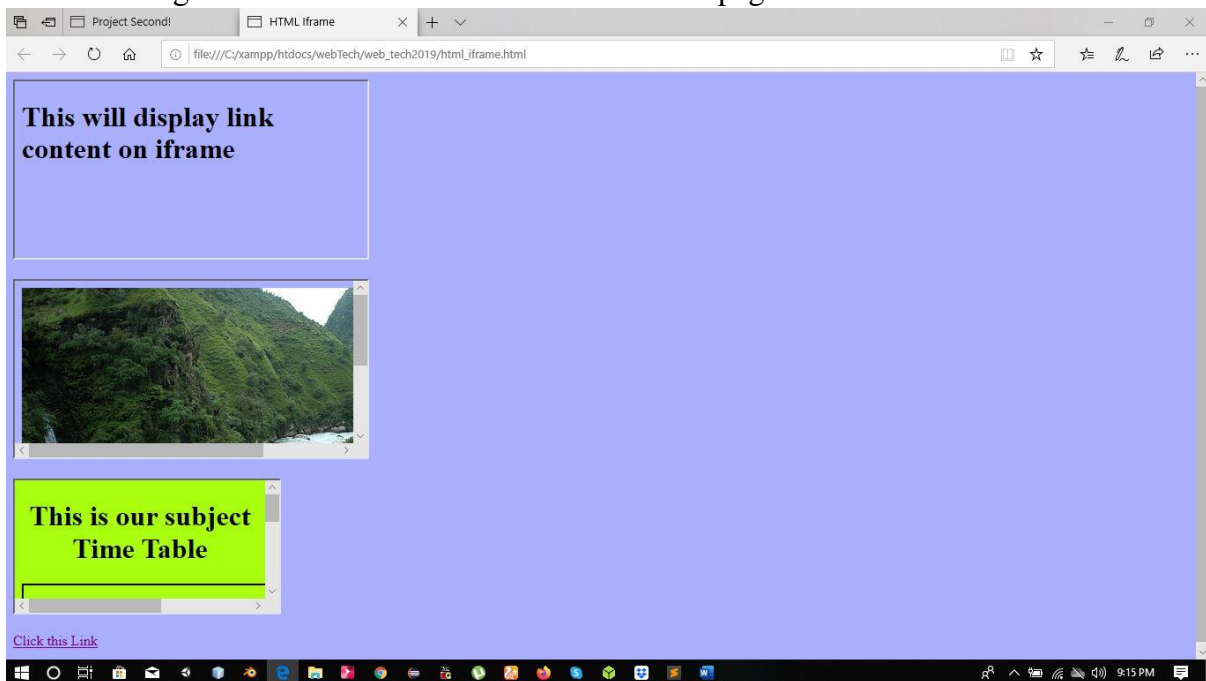
### Project 2:



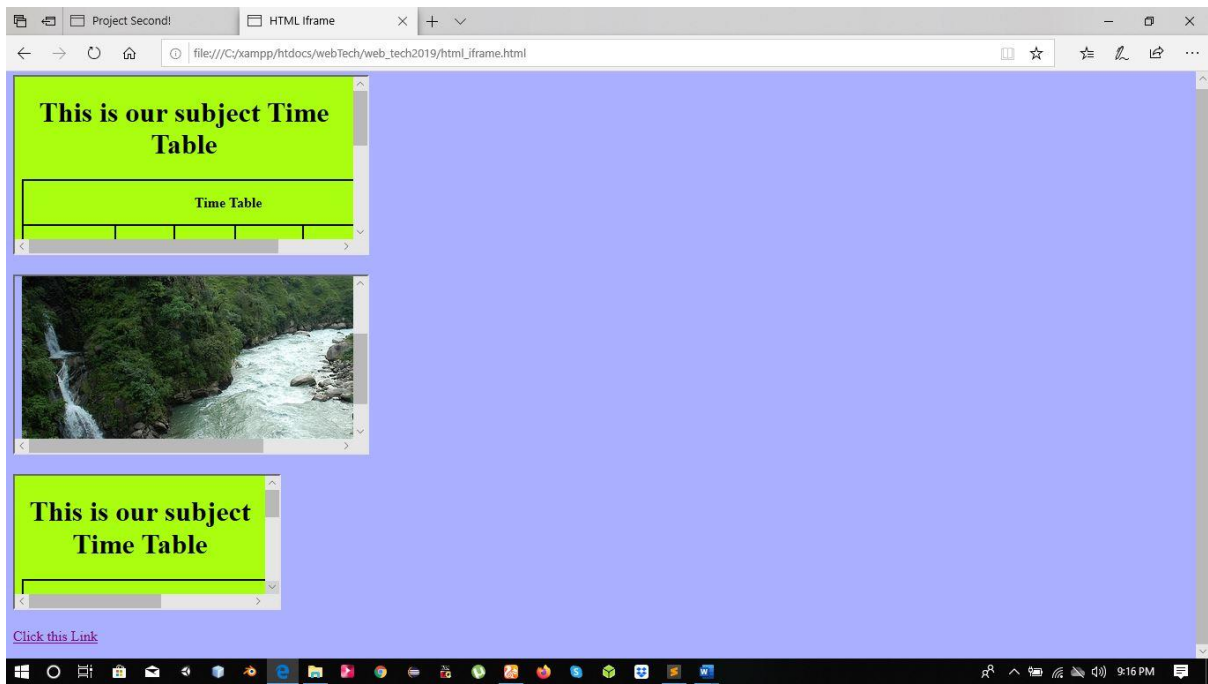
- First of all , you need to design page like above . Where you see lists(i.e ordered list and un-ordered list) and table.
- On Ordered list “[Table](#)” is the text hyper-link that link to next html page having title “link to table” and content on that page as below:



- Similarly, On Un-ordered list “**Iframe**” is the text hyper-link that link to next html page having title “link to iframe” and content on that page as below:



- Now, on the above page you see all of them are iframe but at the bottom-left corner there is the hypertext link “**Click this Link**”.
- When you clicked to that link some content will be visible at the section “**This will display link content on iframe**”. As shown below:



## Lab 3:HTML Form and Image Mapping

**Objective:** To map an image and design HTML Forms.

**Theory:**

### Forms

- Form is used to communicate information from a web browser to the server.
- Also form is a window or screen i.e. used to gathering information(data) and also visualized with different , numerous field or spaces to enter data.
- Generally , form is used to enter data.
- In HTML5 form is defined within the `<form>` tag and having different control attributes.

**Example:**

```
<!DOCTYPE html>
<html>
<head>
    <title> HTML Form</title>
</head>
<body>

    <h1>Create New account</h1>
    <form action="" name="" target="" method="post">
        Name<input type="text" name="name"><br><br>
        Username<input type="text" name="username"><br><br>
        Address<input type="text" name="address"><br><br>
        Gender
        <input type="radio" name="gender" value="male" checked>Male
        <input type="radio" name="gender" value="female">Female
        <input type="radio" name="gender" value="others">Others<br><br>
        Subjects:
        <input type="checkbox" name="sbject" value="math" checked>Math
        <input type="checkbox" name="sbject" value="C">C-program
        <input type="checkbox" name="sbject" value="c++">C++<br><br>
        Password<input type="text" name="pass"><br><br>
        Confirm-Password<input type="text" name="confirm_password"><br><br>
        <input type="submit" name="submit" value="Submit">
    </form>
</body>
</html>
```

**Attributes that are used to control form:**

- *action* – is used to define action to be performed when the form is submitted.
- *method* – is used to specifies HTTP method(post and get) when submitting the form.
- *name* – is used to sent data to all (where needed). Also used to identify the form.
- *target* – is used to specifies the target address.

**Some elements that are used within the form:**

- *input* element – is used to take input from user on browser i.e. it provides input field. Input fields are defined by an attribute called *type(text, checkbox, radio , password, email, reset, submit, image, button , color , etc)*.
- *fieldset* – is used to group related data in a form.
- *legend* – is used to define caption for `<fieldset>` element.

**Input type checkbox** – defines checkbox let a user select ZERO or MORE options of a limited number of choices.

**Radio button** – let user to chose / select one of a limited choice

**Input type reset** – defines reset button that is used to reset form and set all form values to default.

### Image Mapping:

- To create an image as a map, **<map>** tag is used and is linked to the image by using the **name** attribute.
- The **name** attribute must have same value to **usemap** attribute.

### Example:

```
<!DOCTYPE html>
<html>
<head>
    <title>Image Mapping</title>
</head>
<body>
    <p>Image map is to define map with clickable areas!</p>
    
    <map name="img-maping">
        <area shape="circle" coords="53,40,30" href="htmlForm.html" target="_blank">
        <area shape="rect" coords="110,28,170,69" href="htmlForm.html" target="_blank">
        <area shape="poly" coords="86,86,64,111,88,134,111,110" href="htmlForm.html"
target="_blank">
    </map>

</body>
</html>
```

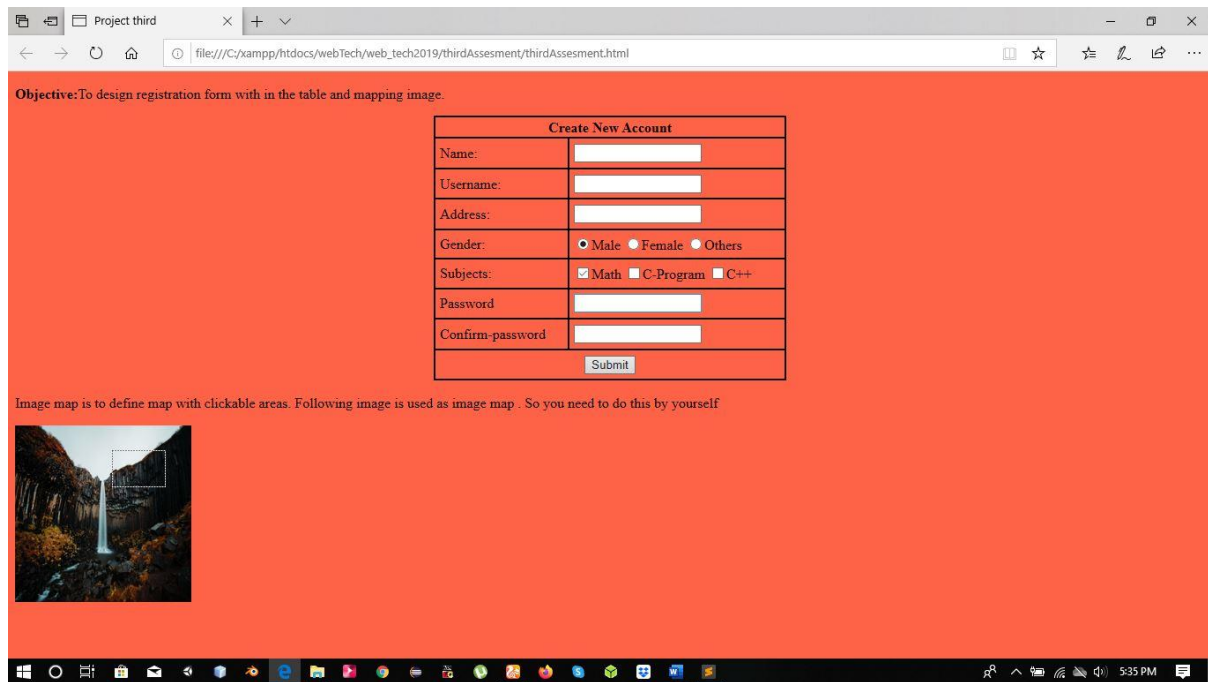
### elements and attribute used on above example:

- **<area>** - this element is used to define clickable area on image used.
- **shape** – attribute define the shape of area you want.
  - ✚ circle – circular region
  - ✚ rect – rectangular region
  - ✚ poly – polygonal region
  - ✚ default – entire region
- **coords** – attribute define coordinates to be able to place clickable on the image.
  - ✚ For circle: “100,50,30” where (100,100) is center point and (30) is radius .
  - ✚ For rect: “100,100,200,150” (100,100) is one point of diagonal and (200,150) is next one point.

**Assignment:** The following assignment is named as Project 3 so you need to complete it and present your output on next week.

### Project 3:

**Design following Form and image Mapping:**



### Description:

- First we have to design **form** to “*Create New Account*” within the table. Where **Male** and **Math** is checked.
- Second one image is mapped image where if you clicked on that image some areas are clickable and addressed to other pages.



## Lab 4: Introduction to Cascading Style Sheets(CSS)

**Objective:** To be familiar with Cascading Style Sheets.

**Theory:**

### Introduction and Levels of style sheets

- CSS stands for “Cascading Style Sheets”.
- Cascading Style Sheets is used to format the layout of web pages.
- CSS helps web developers to create uniform look across several pages on websites.
- Using CSS makes task easier to define common layout across the pages.
- CSS is specially used to style pages, including the design, layout and variations in display for different devices and screen sizes. .

**There are three levels of style sheets:**

(The style specification formats : The format of a style specification depends on the level of style sheet .)

#### 1. Inline

- is specified for a specific occurrence of a tag and apply only to that tag.
- appears in the *tag* itself.
- is fine-grain style, which defeats the purpose of style sheets - uniform style.

#### 2. Document-level style sheets

- apply to the whole document in which they appear
- appear in the *head* of the document

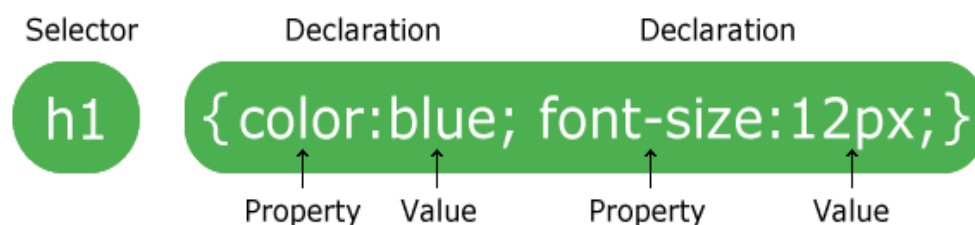
#### 3. External style sheets

- can be applied to any number of documents
- are in separate files, potentially on any server on the Internet
- use a `<link>` tag to let the browser fetch and use an external style sheet file
- `<link rel = "stylesheet" type = "text/css" href = "style.css"></link>`
- The file should not contain any html tags.
- The style sheet file must be saved with a .css extension.

Note: So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

### CSS Syntax

- A CSS rule-set consists of a selector and a declaration block:



- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a CSS property name and a value, separated by a colon.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.
- In the above example all `<h1>` elements will be of font color blue, with font size 12px.

**Example:**

```

<!DOCTYPE html>
<html>
<head>
//Internal level css
<style>
body {
                background-color: linen;
    }
    h1 {
                color: maroon;
                margin-left: 40px;
    }
</style>
//External level css
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <p style="color: red;">This is also paragraph</p> //Inline Level css
</body>
</html>
/*(style.css )This is an external file named style.css*/
h1{ font-size: 25px;}

```

**Note:**

*Do not add a space between the property value and the unit (such as margin-left: 25 px; ). The correct way is : margin-left: 25px;*

**CSS Selectors**

- CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more.

**Types of selectors:****1. The element selectors**

- The element selector selects elements based on the element name.
- The simplest selector form is a single element name ,such as *h1*.
- The property values in the rule apply to all occurrences of the named element.
- The selector could be a list of element names separated by commas (*h1,h2,h3, ....*).

**2. Class selectors**

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the name of the class.
- Class selectors are used to allow different occurrence of the same tag to use different style specifications.

**3. Id selectors**

- The id selector uses the *id* attribute of an HTML element to select a specific element.

- The id of an element should be unique within a page, so the id selector is used to select one unique element!
- To select an element with a specific id, write a hash (#) character, followed by the id of the element.

#### 4. Universal selector

- CSS Universal selector selects any type of elements in HTML page.
- CSS Universal selector is defined by using asterisk (i.e. \*).
- CSS Universal selector is also followed by selector.
- Universal selector is useful to style all elements in HTML pages or used to style all elements with in element.

#### 5. Pseudo Classes

- Sometime we need to apply style when something happen , rather than because the target element simply exists, i.e. specify by pseudo classes.

#### Example:

```
/* Single '*' is used to set up for all body elements */
*{
    color: blue; /*defines blue color for all font*/
    background-color: green; /*define green background color */
    font-size: 20px; /*defines font size for all font*/
}
/*Following use of asterisk is used to style all elements of div element*/
div *{
    color: yellow; /*define yellow color to all fonts with in div elements*/
    font-size: 30px; /*define font size to all fonts with in div elements*/
}
```

## Lab 5: How to use CSS properties and value to HTML elements.

**Objective:** Implementation of CSS properties and value.

**Theory:**

### CSS Backgrounds

- The CSS backgrounds are specified for background effects on elements.
- The CSS background properties are:
  - + **background-color** – specifies background color of an element.
  - + **background-image** – specifies background image of an element(*background-image: url("karnali.jpg").* **Note** – if necessary give path.
  - + **background-attachment** - property sets whether a background image scrolls with the rest of the page, or is fixed. (Values are: *scroll, fixed, inherit, initial and local*)
  - + **background-position** - sets the starting position of a background image.(Values are: *left top, left center, left bottom, right top, right center, right bottom, center top, center bottom, and center center*)
  - + **background-repeat** - sets if/how a background image will be repeated. (Values are: *no-repeat, repeat-y, repeat-x, space, and round*)
  - + **background-size** - specifies the size of the background images.(Values are: *auto, cover, contain, length, percentage, initial, and inherit*)
  - + **background-origin** - specifies the origin position (the background positioning area) of a background image.(Values are: *padding-box, content-box, border-box, inherit and initial*)
  - + **background-clip** - defines how far the background (color or image) should extend within an element. (Values are: *padding-box, content-box, border-box, inherit and initial*)
  - + **background-blend-mode** - defines the blending mode of each background layer (color and/or image).(Values are: *normal, multiply, screen, overlay, darken, lighten, color-dodge, saturation, color, luminosity;*)

#### Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
#myDIV {
    width: 200px;
    height: 200px;
    background-repeat: no-repeat, repeat;
    background-image: url("bas2.jpg"), url("karnali.jpg");
    background-blend-mode: lighten;
}
</style>
</head>
<body>
    <div id="myDIV"></div>
    <p><b>Note:</b> Edge/Internet Explorer do not support the
        background-blend-mode property.</p>
</body>
</html>
```

## CSS Colors

- Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.
- You can add color to text, border and background.
- There are different method to assign colors. Which are given below:

✚ **RGB Value** - In HTML, a color can be specified as an RGB value, using this formula: **rgb(red, green, blue)**. Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.

✚ **HEX Value** - In HTML, a color can be specified using a hexadecimal value in the form: **#rrggbb** Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

✚ **HSL Value** - In HTML, a color can be specified using hue, saturation, and lightness (HSL) in the form: **hsl(hue, saturation, lightness)** Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue. Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color. Lightness is also a percentage, 0% is black, 50% is neither light or dark, 100% is white.

✚ **RGBA** – RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color. An RGBA color value is specified with: **rgba (red, green, blue, alpha)** The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

✚ **HSLA** - are an extension of HSL color values with an alpha channel - which specifies the opacity for a color.

## CSS Text Properties

- **color** – is defined to set color for text.
- **text-decoration** – is used to set or remove decorations from text.(Values are: none, underline, overline and line-through)
- **text-align** – is used to set horizontal text alignment.(Values are: left, center, right and justify)
- **text-transform** – is used to specify uppercase and lowercase letters in text.(Values are: uppercase, lowercase and capitalize)
- **text-indent** – is used to specify indentation of the first line of a text.
- **line-height** – is used to define space between lines.
- **direction** – is used to change the text direction of an element. (Value is *rtl* i.e. right to left)
- **word-spacing** – is used to specify the space between the words in a text.
- **text-shadow** – add shadow to text.

### Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
    .changeColor{
        color: green;
    }
    .txtAlignment{
        text-align: center;
    }
    a{
```

```

        text-decoration: none; /*remove decoration to text*/
    }
    .uppercase{
        text-transform: uppercase;
    }
    .txtIndent{
        text-indent: 50px;
    }
    h2 {
        letter-spacing: 7px; /*Also can use -ve number.*/
    }
    .txtShadow{
        text-shadow: 3px 2px red;
    }
</style>
</head>
<body>
    <h1 class="changeColor">This is heading 1</h1>
    <h1 class="txtAlignment">You can align as you like</h1>
    <a href="https://www.google.com">Hello</a>
    <p class="uppercase">You Can use other Transform.</p>
    <p class="txtIndent"> It start text line after 50px indentation.</p>
    <h2>This is heading 2</h2>
    <h1 class="txtShadow">Text-shadow effect</h1>
</body>
</html>

```

## CSS max-width property

- The max-width is used to set the maximum width of an element.
- The max-width can be specified in *length values*, like px, cm, etc., or in percent (%) of the containing block, or set to none (this is default. Means that there is no maximum width).
- The max-width property handle the browser if the content of element is larger then browser.

### Example:

```

<html>
<head>
<style>
div {
    max-width: 500px;
        height: 100px;
        background-color: powderblue;
    }
</style>
</head>
<body>
<h2>Set the max-width of an element</h2>
<div></div>
<p>Resize the browser window to see the effect.</p>
</body>
</html>

```

## CSS Layout – The *position* Property

- specifies the type of positioning method used for an element (*static*, *relative*, *fixed*, *absolute* or *sticky*).
- position ***static*** – this is default value .

- position **relative** – relative to its normal position.
- position **fixed** – is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled.
- position **absolute** – is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).
- position **sticky** – is positioned based on the user's scroll position .A sticky element toggles between relative and fixed , depending on the scroll position.

**Example:**

```
<!DOCTYPE html>
<html>
<head>
<style>
div.shift{
    top: 100px;
    padding-top: 25px;
}
div.relative {
    position: relative;
    width: 400px;
    height: 200px;
    border: 3px solid #73AD21;
}
div.absolute {
    position: absolute;
    top: 80px;
    right: 0px;
    width: 200px;
    height: 100px;
    border: 3px solid #73AD21;
}
img {
    position: absolute;
    left: 0px;
    top: 0px;
    z-index: -1;
}
</style>
</head>
<body>
    <h1>This is a heading</h1>
    
    <p>Because the image has a z-index of -1, it will be placed behind the text.</p>
    <div class="shift"><h2>position: absolute;</h2>
    <p>An element with position: absolute; is positioned relative to the nearest positioned
    ancestor (instead of positioned relative to the viewport, like fixed):</p>
    <div class="relative">This div element has position: relative;
    <div class="absolute">This div element has position: absolute;</div>
</div>
</div>
</body>
</html>
```

## Example: Using CSS(style.css) to style table and Un-ordered list

**/\*tableStyle.html\*/**

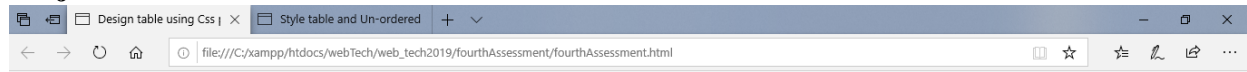
```
<!DOCTYPE html>
<html>
<head>
    <title>Style table and Un-ordered list using
    CSS properties</title>
    <link rel="stylesheet" type="text/css"
href="style.css">
</head>
<body>
    <table id="tbl1">
        <tr>
            <th>Name:</th>
            <th>Roll NO:</th>
            <th>Address:</th>
            <th>Faculty:</th>
        </tr>
        <tr>
            <td>Ram</td>
            <td>4</td>
            <td>Kathmandu</td>
            <td>BE-Computer</td>
        </tr>
        <tr>
            <td>Hari</td>
            <td>7</td>
            <td>Kathmandu</td>
            <td>BE-Computer</td>
        </tr>
        <tr>
            <td>Shyam</td>
            <td>3</td>
            <td>Bhaktapur</td>
            <td>BE-IT</td>
        </tr>
        <tr>
            <td>Sita</td>
            <td>9</td>
            <td>Lalitpur</td>
            <td>BE-CIVIL</td>
        </tr>
    </table><br><br>
    <table id="tbl2">
        <tr>
            <th>Name:</th>
            <th>Roll NO:</th>
            <th>Address:</th>
            <th>Faculty:</th>
        </tr>
        <tr>
            <td>Ram</td>
```

/\*style.css\*/
table,th,td{
 border: 1px solid black;
 border-collapse: collapse;
}
tr:nth-child(even){
 background-color: lightgray;
}
tr:hover{
 background-color: red;
}
#tbl2{
 margin-top: 0px;
 margin-left: 400px;
 position: relative;
}
#tbl1{
 margin-top: 20px;
 position: absolute;
}
ul{
 list-style-image: url("test2.gif");
}



**Assignment:** The following assignment is named as Project 4 so you need to complete it and present your output on next week.

## Project 4:



### This is our subject Time Table

Time Table 1				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	C++	DSA
Monday	logic Circit		C	C++
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

### This is our subject Time Table

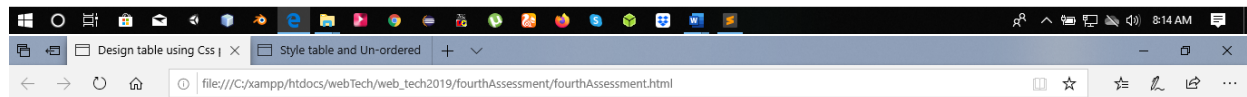
Time Table 3				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	Java	DSA
Monday	logic Circit		C	C++
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

### This is our subject Time Table

Time Table2				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	C++	DSA
Monday	logic Circit		C	C++
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

### Unordered list

- 🖼 Hello
- 🖼 Namaste
- 🖼 How do you do?



### This is our subject Time Table

Time Table 1				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	C++	DSA
Monday	logic Circit		C	C++
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

### This is our subject Time Table

Time Table 3				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	Java	DSA
Monday	logic Circit		C	C++
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

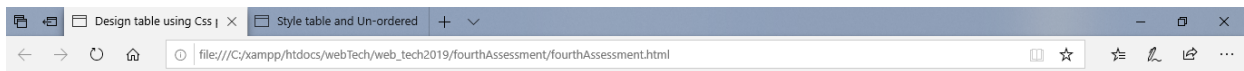
### This is our subject Time Table

Time Table2				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	C++	DSA
Monday	logic Circit		C	C++
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

### Unordered list

- 🖼 Hello
- 🖼 Namaste
- 🖼 How do you do?





**This is our subject Time Table**

Time Table 1				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	C++	DSA
Monday	logic Circit	C	C++	
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

**This is our subject Time Table**

Time Table 3				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	Java	DSA
Monday	logic Circit	C	C++	
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

**This is our subject Time Table**

Time Table2				
Days	7:00-8:40	8:40-10:20	11:10-12:50	12:50-2:30
Sunday	Web	C	C++	DSA
Monday	logic Circit	C	C++	
Tuesday	C	C++	logic Circit	
Wednesday	C++	C	Web	DSA

### Unordered list

- 🖼 Hello
- 🖼 Namaste
- 🖼 How do you do?



### Description:

- Above project shows three tables and Un-ordered list with image as a list-type.
- All table have same design and effects.
  - First** effect is to define background-color of even rows only.
  - Second** effect is to make rows hover ( when you have mouse over to row ,that shows background-color changed.
- “**Time Table 3**” is designed just before the “**Unordered list**” in HTML file but make it to visible just right to the “**Time Table 1**” on browser using CSS properties.
- Un-ordered list** is designed with image as a list-type

## Lab 6: Designing webpage Layout(Including Header, Navigation Bar, content , Footer) using CSS

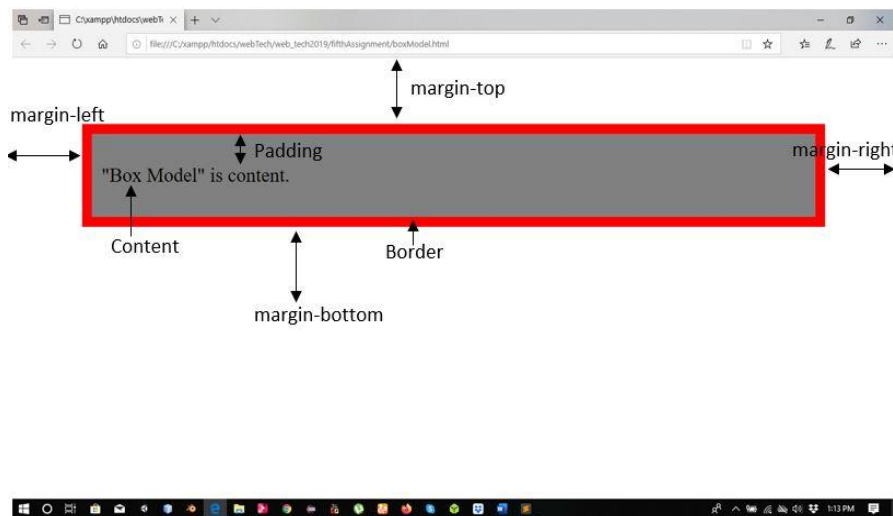
**Objective:** To design simple webpage Layout.

**Theory:**

- Web design is the process of creating websites with several different aspects including webpage layout, content and graphic design.
- Web design and Web development are interchangeable term but define differently. While Web design is technically subset of border category of web development.

**Now here are some properties :**

**Box Model:**



In HTML, all elements can be considered as boxes but in CSS “Box Model” is considered on design and layout.

Box Model consist of margins, borders , padding and the actual content. Here we define css box model to wrap around every HTML content.

**content** – The content of the box, where text and images appear.

**padding**(-top,-right,-bottom and -left) – Clears an area around the content. The padding is transparent.

**border** – A border that goes around the padding and content.

**margins**(-top,-right,-bottom and -left) – area outside the border.

**Syntax:**

<b>Padding:</b> shorthand rule padding: 15px;(all sides 15px) padding: 5px 10px 15px 20px;( top-5px, right-10px, bottom-15px, left-20px)	<b>Margin:</b> shorthand rule margin: 15px;(all sides 15px) margin: 5px 10px 15px 20px;( top-5px, right-10px, bottom-15px, left-20px)	<b>Border:</b> shorthand rule border: 15px solid black; (border-width:15, border-style:solid, border-color:black)
--	---	---

**Example:**

```
<!DOCTYPE html>
<html>
<head>
```

```

<title>Box Model</title>
<style type="text/css">
    div{
        background-color: gray;
        font-size: 30px;
        padding: 15px;
        border:15px solid red;
        margin: 100px;
    }
</style>
</head>
<body>
    <div>
        <p>"Box Model" is content.</p>
    </div>
</body>
</html>

```

### Outline:

An outline is a line that is drawn around elements, OUTSIDE the borders, to make the element "stand out".

CSS has the following outline properties:

- **outline-style**
- **outline-color** (outline-color: red;)
- **outline-width** – (outline-width: 15px/thick;)
- **outline-offset** – The outline-offset property adds space between an outline and the edge/border of an element. The space between an element and its outline is transparent.
- **outline** – shorthand rule( outline: thick ridge pink;)

#### CSS Outline Style

The outline-style property specifies the style of the outline, and can have one of the following values:

- **dotted** - Defines a dotted outline
- **dashed** - Defines a dashed outline
- **solid** - Defines a solid outline
- **double** - Defines a double outline
- **groove** - Defines a 3D grooved outline
- **ridge** - Defines a 3D ridged outline
- **inset** - Defines a 3D inset outline
- **outset** - Defines a 3D outset outline
- **none** - Defines no outline
- **hidden** - Defines a hidden outline

### Example:

```

<!DOCTYPE html>
<html>

```

```

<head>
  <title>Box Model</title>
  <style type="text/css">
    div{
      background-color: gray;
      font-size: 30px;
      padding: 15px;
      border:15px solid red;
      margin: 100px;
      outline: thick ridge pink;
    }
  </style>
</head>
<body>
  <div>
    <p>"Box Model" is content.</p>
  </div>
</body>
</html>

```

### Float:

The float property is used for positioning and formatting content e.g. let an image float left to the text in a container.

The float property can have one of the following values:

- left - The element floats to the left of its container
- right - The element floats to the right of its container
- none - The element does not float (will be displayed just where it occurs in the text). This is default
- inherit - The element inherits the float value of its parent

### Example:

```

<!DOCTYPE html>
<html>
<head>
  <title>Box Model</title>
  <style type="text/css">
    div{
      background-color: gray;
      font-size: 20px;
      padding: 10px;
      height: 150px;
      width: 100%;
    }
    img{
      float: right;
      margin-left: 15px;
      padding: 10px;
    }
    p{
      float: right;
    }
  </style>
</head>
<body>

```

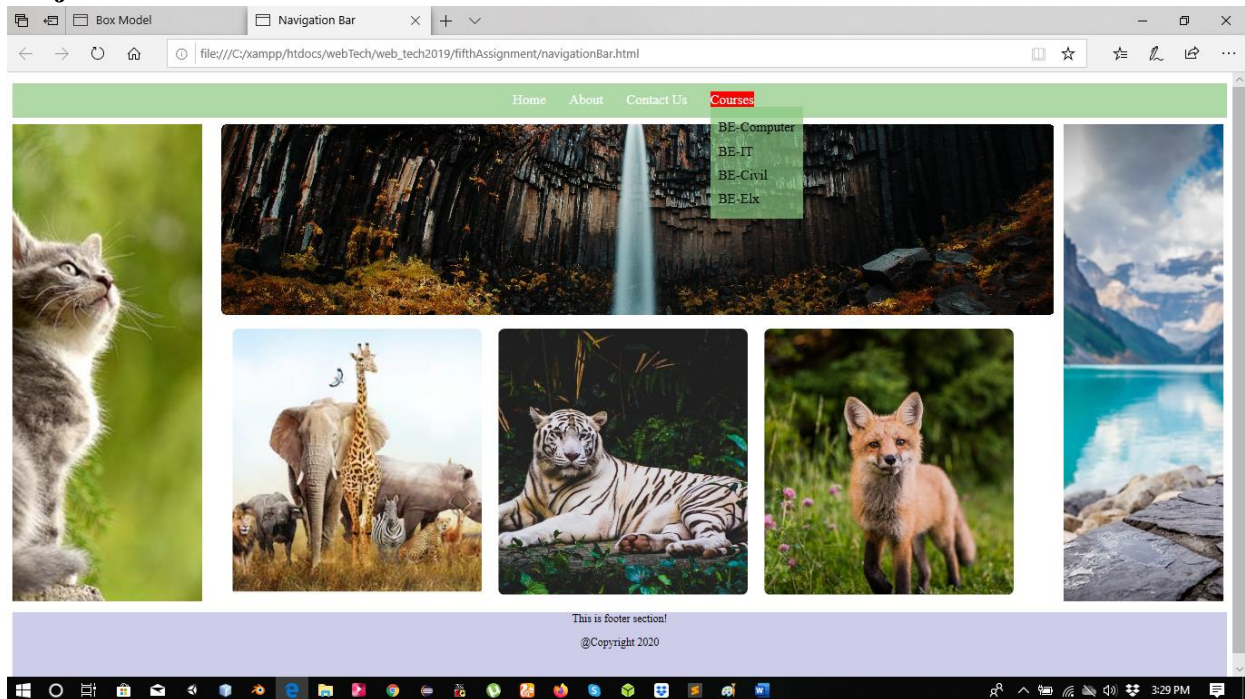
```

<div>
    
    <p>This is the image of Karnali river. It belongs to Karnali district.</p>
</div>
</body>
</html>

```

**Assignment:** The following assignment is named as Project 5 so you need to complete it and present your output on next week.

### Project 5:



### Description:

Above figure represent simple but basic web page layout include following section:

- First, **Navigation Bar**(Home, About, Contact Us and Courses) with **height-100px** and **width-100%** , background-color and Courses as **dropdown** Menu.
- **Content** Area only have images (**format**: left-column content, main content and right-column content).
- Last one , **footer** section with background-color and few information.

**Note:** *This page must be responsive.*

## Lab 7: Introduction to JavaScript

### Theory:

- JavaScript is a programming language for the web , can update and change both HTML and CSS, also can calculate , manipulate and validate data.
- JavaScript is divided into three parts: Core, Client-side and Server-side .
- Here, we are going to study basic on Client-side JavaScript.
- Now a days JavaScript is almost used in every sector.
- Everything JavaScript are Objects.

### JavaScript Variable:

In JavaScript variable is declared using **var** keyword.

#### Syntax:

```
var variableName;
```

### JavaScript Function:

**function** keyword is used to define Function .

#### Syntax:

```
function functionName()  
{  
}
```

### JavaScript Objects:

**Objects** are variables too and can hold many values.

#### Syntax:

```
var objectName = {  
    objectProperty1: "propertyValue1",  
    objectProperty2: "propertyValue2",  
    objectProperty3: "propertyValue3", .....  
}
```

Access using **objectName.objectProperty** (eg: *objectName.objectProperty1*)

### JavaScript Array:

**Array** is defined as special type of variable that hold multiple values .

#### Syntax:

```
var arrayName = ["element1", "element2", "element3", "element4", .....];  
OR  
var arrayName = new Array(["element1", "element2", "element3", .....]);
```

### JavaScript Loop:

**Loops** are used to execute code of block.

#### Different Kinds of Loops:

JavaScript supports different kinds of loops:

- for - loops through a block of code a number of times
- for/in - loops through the properties of an object
- for/of - loops through the values of an iterable object
- while - loops through a block of code while a specified condition is true
- do/while - also loops through a block of code while a specified condition is true

#### Syntax:

```
for(initialization; Condition; Increment/Decrement){  
}
```

**Example 1:** Illustrate JavaScript **variable** declaration, JavaScript **function** and JavaScript **Objects**.

```
<!DOCTYPE html>
<html>
<head>
    <title>Introduction to JavaScript</title>
</head>
<body>
    <p id="demo1"></p>
    <p class="demo1"></p>
    <p id="demo2"></p>
    <p id="demo3"></p>
    <p id="demo4"></p>
    <script type="text/javascript">
        var a=5;
        var b=6;
        var c=a+b;
        document.getElementById('demo1').innerHTML=c;
        document.write('Hello world');

        //function without parameter/argument
        function myFun(){
            return "Hello";
        }
        var x=document.getElementsByClassName('demo1');
        x[0].innerHTML=myFun();

        // function with single parameter/argument
        function myFunction(value){
            return "Welcome "+value;
        }
        var y=document.getElementById("demo2");
        y.innerHTML=myFunction("Ram");

        //creating Object
        function createObject(){
            var detail={
                name:"Hari",
                lastName:"Rawal",
                age:"25",
                address:"Kanchanpur"
            };
            return "My name is "+detail.name+" and surname is "+detail.lastName+".I am "+detail.age+" years old and from "+detail.address+";";
        }
        var x=document.getElementById('demo3');
        x.innerHTML=createObject();

        //ObjectProperty as function
        var details={
            name:"Shyam",
            lastName:"saud",
```



```

        age:"29",
        address:"Kanchanpur",
        info: function(){
            return "My name is "+this.name+" and surname is "+this.lastName+".I
am "+this.age+" years old and from "+this.address+";
        }
    };
    var f=document.getElementById('demo4');
    f.innerHTML=details.info();
</script>
</body>
</html>

```

### Example 2: Illustration of JavaScript Array and Looping.

```

<!DOCTYPE html>
<html>
<head>
    <title>JavaScript Array and Looping</title>
</head>
<body>
    <p id="demo1"></p>
    <p id="demo5"></p>
    <p id="demo2"></p>
    <p id="demo6"></p>
    <p id="demo3"></p>
    <p id="demo7"></p>
    <p id="demo4"></p>
    <p id="demo8"></p>
    <script type="text/javascript">
        var fruits = ["mango","apple","Orange"];
        document.getElementById('demo1').innerHTML="<b>fruits </b>"+ " is an array.";
        document.getElementById('demo5').innerHTML=fruits;

        //for loop
        var len = fruits.length; //array property
        var i,text="";
        document.getElementById('demo2').innerHTML="<h1>"+ "This is for loop."+"</h1>";
        for(i=0;i<len;i++){
            var a=i+1;
            text += a+" list of fruits "+fruits[i]+"<br>";
        }
        document.getElementById('demo6').innerHTML=text;
        //for/in loop
        document.getElementById('demo3').innerHTML="<h1>"+ "This is for/in loop."+"</h1>";
        var detail={
            name:"Hari",
            lastName:"Rawal",
            age:"25",
            address:"kathmandu"
        };
        var txt ="";
        var x;
        for(x in detail){

```

```

        txt +=detail[x]+" ";
    }
    var x=document.getElementById('demo7');
    x.innerHTML=txt;

    //for/of loop
    document.getElementById('demo4').innerHTML="<h1>"+ "This is for/of loop."+"</h1>";
    var y;
    for(y of fruits){
        document.write(y + "<br >");
    }
</script>
</body>
</html>

```

## Lab 8: Event handling and *this* keyword in JavaScript

### Theory:

#### *this* keyword

- In a function definition, *this* refers to the "owner" of the function.
- The JavaScript *this* keyword refers to the object it belongs to.
- It has different values depending on where it is used.
  - ✚ In a method, *this* refers to the **owner object**.
  - ✚ Alone, *this* refers to the **global object**.
  - ✚ In a function, *this* refers to the **global object**.
  - ✚ In a function, in strict mode, *this* is **undefined**.
  - ✚ In an event, *this* refers to the **element** that received the event.
  - ✚ Methods like **call()** and **apply()** can refer *this* to **any object**.

#### Example:

```
<html>
<body>
  <h2>The JavaScript <b>this</b> Keyword</h2>
  <p>In this example, <b>this</b> represents the <b>person</b> object.</p>
  <p>Because the person object "owns" the fullName method.</p>
  <p id="demo"></p>
  <p>In this example <strong>this</strong> refers to person2, even if it is a method of person1:</p>
  <p id="demo1"></p>
  <button onclick="this.style.display='none'">Click to Remove Me!</button>
</script>
  // Create an object:
  var person = {
    firstName: "John",
    lastName : "Doe",
    id      : 5566,
    fullName : function() {
      return this.firstName + " " + this.lastName;
    }
  };
  // Display data from the object:
  document.getElementById("demo").innerHTML = person.fullName();
  //using method call()
  var person1 = {
    fullName: function() {
      return this.firstName + " " + this.lastName;
    }
  }
  var person2 = {
    firstName:"Ram",
    lastName: "Prasad",
  }
  var x = person1.fullName.call(person2);
  document.getElementById("demo1").innerHTML = x;
</script>
</body></html>
```

#### Event and Event Handling

- **Events** are action that are detected by JavaScript.
- JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page.

- To execute code when a user clicks on an element, add JavaScript code to an HTML event attribute.
- There are many events . But some Examples of HTML events are:
  - ✚ When a user clicks the mouse
  - ✚ When a web page has loaded
  - ✚ When an image has been loaded
  - ✚ When the mouse moves over an element
  - ✚ When an input field is changed
  - ✚ When an HTML form is submitted
  - ✚ When a user strokes a key
- When an event occur then this action get response from script , this process is called **event handling**. Some of them are listed below:

***onclick event type:***

This is the most frequently used event type which occurs when a user clicks the left button of his mouse.

**Example:**

```
<html>
  <head>
    <title>JS event </title>
  </head>
  <body>
    <p>Click the following button and see result</p>
    <button id="myBtn">Try it</button>
    <p id="demo"></p>
    <form>
      <input type = "button" onclick = "sayHello()" value = "Say Hello" />
    </form>
    <script type = "text/javascript">
      function sayHello() {
        alert("Hello World")
      }
      document.getElementById("myBtn").onclick = displayDate;
      function displayDate() {
        document.getElementById("demo").innerHTML = Date();
      }
    </script>
  </body>
</html>
```

***onmouseover and onmouseout event type:***

The onmouseover event triggers when you bring your mouse over any element and the onmouseout triggers when you move your mouse out from that element.

**Example:**

```
<!DOCTYPE html>
<html>
  <body>
    <div onmouseover="mOver(this)" onmouseout="mOut(this)"
      style="background-color:#D94A38;width:120px;height:20px;padding:40px;">
      Mouse Over Me</div>
    <script>
      function mOver(obj) {
        obj.innerHTML = "Release Me";
      }
      function mOut(obj) {
```

```

    obj.innerHTML = " Thank You ";
}
</script>
</body>
</html>

```

#### ***onmousedown and onmouseup event type:***

The ***onmousedown***, ***onmouseup***, and ***onclick*** events are all parts of a mouse-click. First when a mouse-button is clicked, the ***onmousedown*** event is triggered, then, when the mouse-button is released, the ***onmouseup*** event is triggered, finally, when the mouse-click is completed, the ***onclick*** event is triggered.

#### **Example:**

```

<!DOCTYPE html>
<html>
<body>
    <div onmousedown="mDown(this)" onmouseup="mUp(this)"
        style="background-color:#D94A38;width:90px;height:20px;padding:40px;">
        Click Me</div>
<script>
    function mDown(obj) {
        obj.style.backgroundColor = "#1ec5e5";
        obj.innerHTML = "Release Me";
    }
    function mUp(obj) {
        obj.style.backgroundColor="#D94A38";
        obj.innerHTML="Thank You";
    }
</script>
</body>
</html>

```

#### ***onchange event type:***

The ***onchange*** event is often used in combination with validation of input fields.

#### **Example:**

```

<html>
<head>
<script>
    function myFunction() {
        var x = document.getElementById("fname");
        x.value = x.value.toUpperCase();
    }
</script>
</head>
<body>
    Enter your name: <input type="text" id="fname" onchange="myFunction()">
    <p>When you leave the input field, the input text transfer to upper case.</p>
</body>
</html>

```

#### ***onblur and onfocus event type:***

The ***onblur*** event occurs when an object loses focus. The ***onblur*** event is most often used with form validation code (e.g. when the user leaves a form field).

The ***onfocus*** event occurs when an element gets focus. The ***onfocus*** event is most often used with ***<input>***, ***<select>***, and ***<a>***.

#### **Example:**

```

<!DOCTYPE html>

```

```

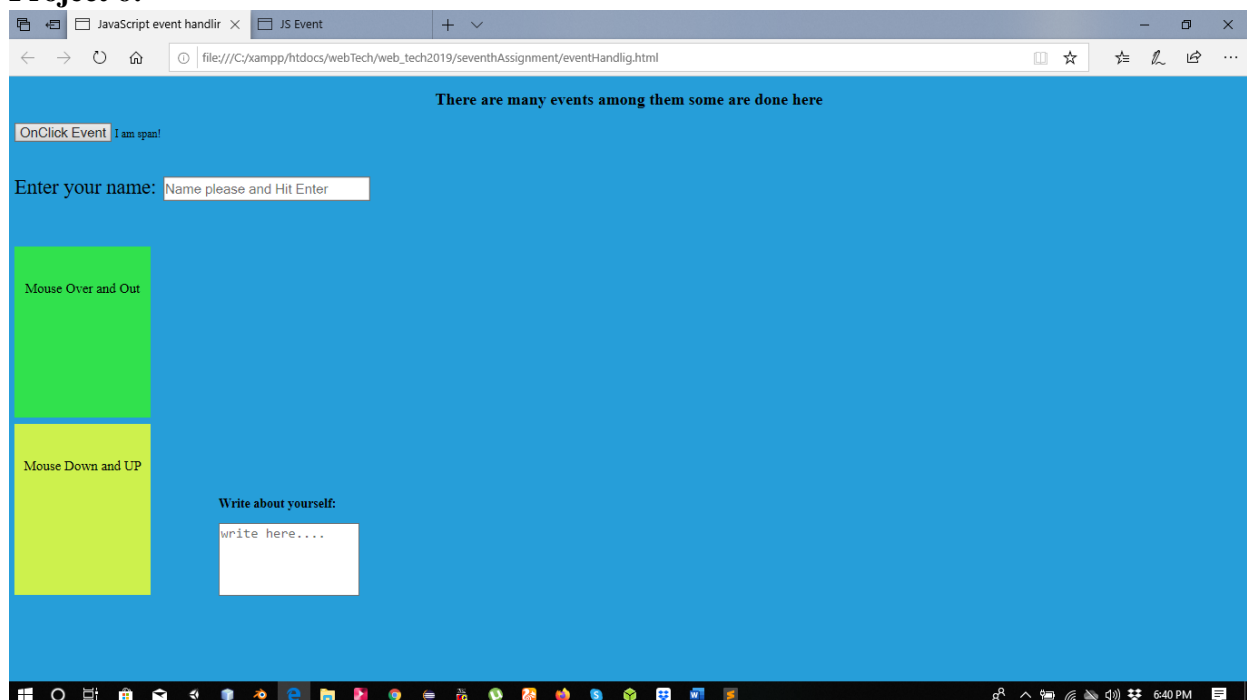
<html>
<body>
  <p>This example uses the HTML DOM to assign an "onfocus" event to an input element.</p>
  <!-- When click for input it get focus but when click out the input field it loses focus i.e. blur. -->
  Enter your name: <input type="text" id="fname">
<script>
  document.getElementById("fname").onfocus = function() {myFunction()};
  function myFunction() {
    document.getElementById("fname").style.backgroundColor = "red";
  }

  document.getElementById("fname").onblur = function() {myFunction1()};
  function myFunction1() {
    document.getElementById("fname").style.backgroundColor = "yellow";
  }
</script>
</body>
</html>

```

**Assignment:** The following assignment is named as Project 6, so you need to complete it and present your output on next week.

### Project 6:



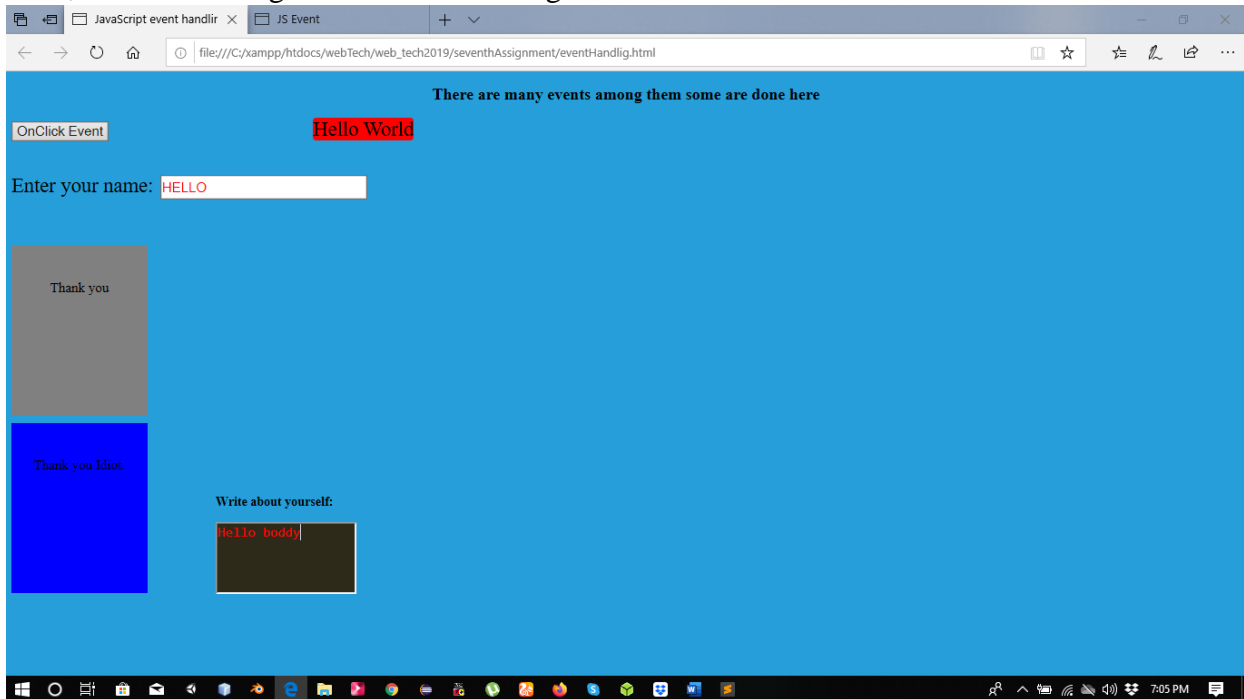
### Description:

On above image all content are based on event and handling them too.

- “**OnClickEvent**” is a button and “I am span!” is a content in span. When click this button it shows some changes . (Event used: **onclick**)
- white space after “Enter your name:” is an **input text** area with **placeholder**. Input text will change on pressing “Enter” button on your PC keyboard.( Event used: **onchange**)
- “**Mouse Over and Out**” is a **div** section with **height**, **width** and **background-color** properties. When mouse-over and mouse-out on this section it changes with some properties .(Events used: **onmouseover** and **onmouseout**)

- “**Mouse Down and Up**” is also **div** section with **height,width** and **background-color** properties. When clicked or mouse-down and un-click or mouse-up on this section it changes with some properties .(Events used: *onmousedown* and *onmouseup*)
- “Write about yourself:” and “write here.... ” with white background(i.e. **textarea**) is to write about you. When you clicked inside the white backgrounded section it is **focused** and clicked out side make **blur**.

Now, all above changes are shown on image below:



Here we see changes, so to do above task first read manual clearly and use those event individual to understand well.

## Lab 9: Prompt, Display and Visibility and Regular Expression in JavaScript

### Theory:

#### Popup Boxes:

Popup Boxes are used to display messages or notification to the user. There are three types of popup boxes: alert, confirm and prompt.

#### Example: Illustrating **alert**, **confirm** and **prompt**

```
<!DOCTYPE html>
<html>
<head>
    <title>Popup Boxes</title>
    <style type="text/css">
        button{
            font-size: 30px;
            font-style: italic;
            padding: 10px;
            margin-top: 15px;
            outline: 2px ridge red;
        }
        span,p{
            font-size: 25px;
            margin-left: 10px;
        }
    </style>
</head>
<body>
    <button onclick="alertFunction()">Alert Button</button><br>
    <button onclick="confirmFunction()">Confirm Button</button><span id="spn"></span><br>
    <button onclick="promptFunction()">Prompt Button</button>
    <p>The sum is :<span id="result"></span></p>
    <script type="text/javascript">
        function alertFunction(){
            alert("Hi,I am alert. Are you all right? Then read.")
        }
        function confirmFunction(){
            var txt;
            if(confirm('Are you OK? Press OK button.')){
                txt = "You pressed OK button";
            }
            else{
                txt = "You pressed cancel button";
            }
            document.getElementById('spn').innerHTML = txt;
        }
        function promptFunction(){
            var num1 = prompt('Enter the first number!');
            var num2 = prompt('Enter the second number!');
            var sum = Number(num1)+Number(num2);
            document.getElementById('result').innerHTML = " "+sum;
        }
    </script>
</body> </html>
```



## Display and Visibility

The **display** property sets or returns the element's display type. The **visibility** property sets or returns whether an element should be visible. Both properties allow author to show or hide an element. However, if you set **display: none**, it hides the entire element, while **visibility: hidden** means that the contents of the element will be invisible, but the element stays in its original position and size.

### Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Display and Visibility Properties</title>
</head>
<body>
  <p id="hideShow"> click HIDE button to hide and SHOW button to show</p>
  <button onclick="hideDisplay()">Hide</button>
  <button onclick="showDisplay()">SHOW</button>
  <p id="hideShow1"> click HIDDEN button to hide and VISIBLE button to show</p>
  <button onclick="hideVisibility()">HIDDEN</button>
  <button onclick="showVisibility()">VISIBLE</button>
  <script type="text/javascript">
    var disp=document.getElementById('hideShow').style;
    function hideDisplay(){
      disp.display="none";
    }
    function showDisplay(){
      disp.display="block";
    }

    var visi=document.getElementById('hideShow1').style;
    function hideVisibility(){
      visi.visibility="hidden";
    }
    function showVisibility(){
      visi.visibility="visible";
    }
  </script>
</body>
</html>
```

## Regular Expression(Pattern Matching)

A regular expression is an object that describes a pattern of characters. The JavaScript **RegExp** class represents regular expressions, and both String and **RegExp** define methods that use regular expressions to perform powerful pattern-matching and search-and-replace functions on text.

### Syntax:

```
var pattern=new RegExp(pattern, attributes);
or simply
var pattern=/pattern/attributes;
```

Here is the description of the parameters –

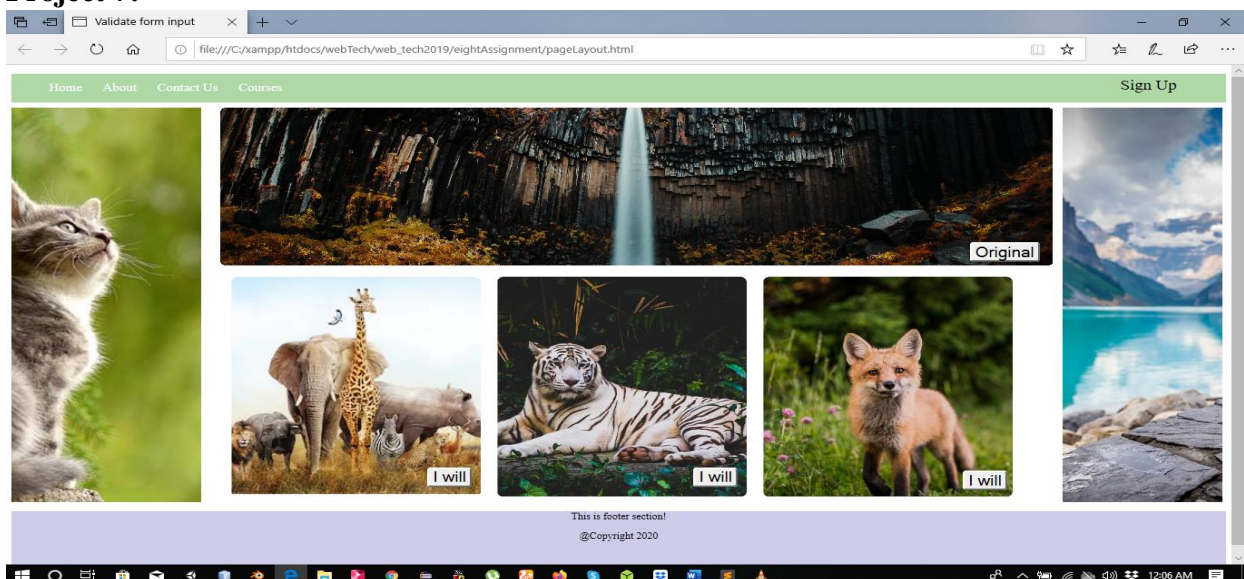
- **pattern** – A string that specifies the pattern of the regular expression or another regular expression.
- **attributes** – An optional string containing any of the "g", "i", and "m" attributes that specify global, case-insensitive, and multi-line matches, respectively.

### Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Validation of character or string</title>
  <style type="text/css">
    p{
      font-size: 20px;          }
    input{
      margin-left: 15px;        }
    span{
      margin-left: 15px;        }
    button{
      font-size: 30px;
      border-radius: 5px;
      box-shadow: 5px 10px 5px grey ;    }
  </style>
</head>
<body>
  <p>Enter User Name:<input type="text" id="user"><span id="spn"></span></p><br>
  <button onclick="validateFunction()">click it!</button>
  <script type="text/javascript">
    function validateFunction(){
      var correct = /^[A-Za-z]+$;/
      var val=document.getElementById('user').value;
      if (val.match(correct)) {
        document.getElementById('spn').innerHTML = "Perfect";
      }
      else{
        document.getElementById('spn').innerHTML = "Invalid!Please character only";
      }
    }
  </script>
</body>
</html>
```

**Assignment:** The following assignment is named as Project 7, so you need to complete it and present your output on next week.

### Project 7:



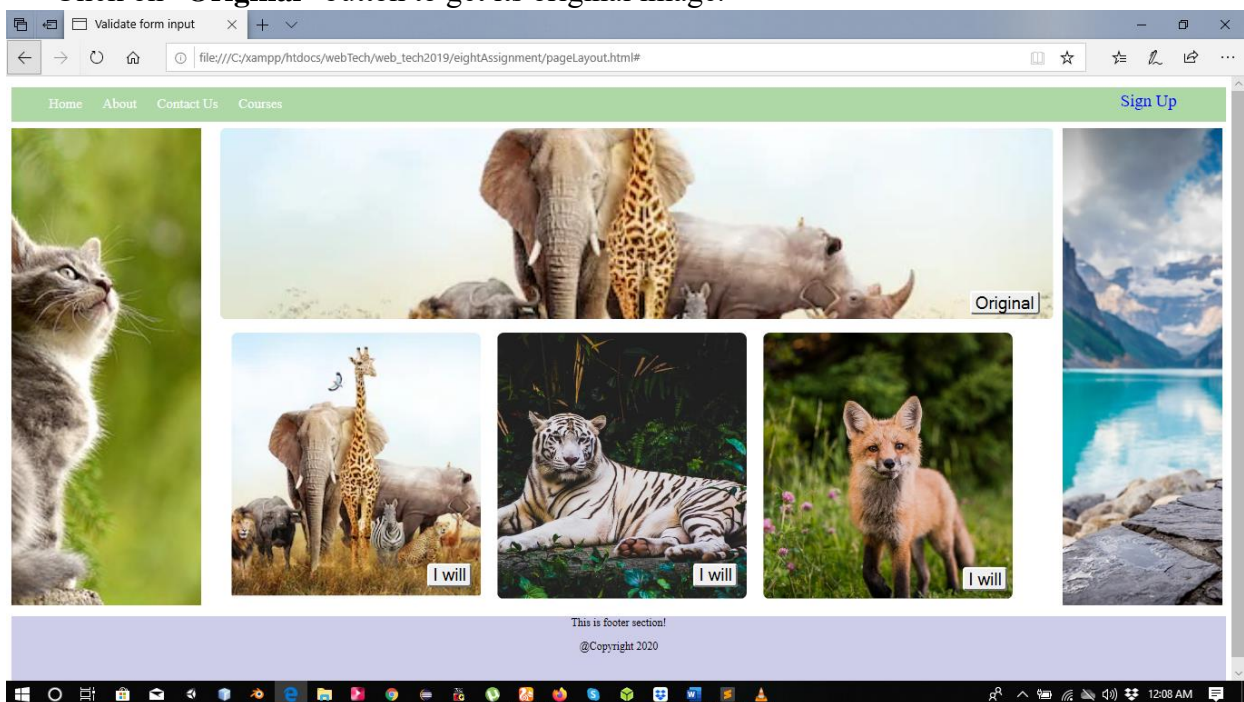
## Description:

- Above Image is a webpage similar to **project 5**. But have some additional elements. Where button named “I will” and “Original” have **onclick** event.
- Also on navigation bar “Sign Up” is used to link form i.e. when you click on “Sign Up” the following form will be appear.
- After the form below you need to fill up with appropriate information and submit it (Form must validate every input. *For example: Name never be number or you cannot submit form without value*).

The screenshot shows a web browser window with the address bar displaying a file path. The page content includes a form with the following elements:

- Name:** A text input field.
- Roll NO:** A text input field.
- Address:** A text input field.
- Phone No:** A text input field.
- Submit**: A button at the bottom of the form.

- When you click on **buttons** they embed their own **image** to section where “**Original**” button is located.
- Image below show that , I clicked on first “**I will**” button and made changes to section where “**Original**” button is located.
- Click on “**Original**” button to get its original image.



## Lab 10: Introduction to PHP

### Theory:

#### Introduction:

- PHP is a server-side scripting language whose scripts are embedded in HTML documents - Similar to JavaScript, but on the server side .
- It takes a PHP document file as input and produces an HTML document file.
- When the php processor finds markup code in the input file, it simply copies it to the output file.
- PHP code is embedded in document by enclosing it between `<?php` and `?>` tags.
- PHP files can contain text, HTML, CSS, JavaScript, and PHP code.
- PHP code is executed on the server, and the result is returned to the browser as plain HTML.
- PHP files have extension ".php".
- Every variable name begins with a dollar sign (\$).
- PHP variable names are case sensitive .
- But, neither reserved words nor function names are case sensitive (no difference between while, WHILE, While, and wHile).
- Single line comments to be specified either with # or with // .
- Multiple line comments are specified with /\* and \*/ .
- PHP statements are terminated with semicolons.
- Braces are used to form compound statements for control statements.

#### Variable:

- Variables are "containers" for storing information. A variable starts with the \$ sign, followed by the name of the variable.
- A variable name must start with a letter or the underscore character.
- A variable name cannot start with a number.
- Variables are case sensitive (\$name and \$Name are different case).

#### Syntax:

*\$variableName = assignValue;*

#### Data types:

- PHP is loosely coupled programming language , don't need to define data types.
- Variable value define the type of data.

PHP supports the following data types:

- String
- Integer
- Float (floating point numbers - also called double)
- Boolean
- Array
- Object
- NULL
- Resource

#### Example:

*\$txt = "Hello"; //Here variable \$txt is a String type data*

*\$a = 5; //Here variable \$a is a Integer type data*

#### PHP Arrays:

- Array is a special types of variable that store multiple values in single variable.

- There are three types of arrays: Indexed Array, Associative Array and Multi-dimensional Array.

### Syntax:

*\$arrayName* = **array**(*item1,item2,.....*);

### Example: PHP Variable, PHP Array

```
<!DOCTYPE html>
<html>
<head>
    <title>Basic about PHP</title>
</head>
<body>
    <h1>Variable declaration.</h1>
    <?php
        $txt1 = "Hello Students!";
        $a = 5;
        $b = 2;
        echo "<b>$txt1</b>";
        echo "<br><b>Sum is ".$a+$b."</b>";
        echo "<br><b>Subtraction is <b>".$a-$b."</b>";
        echo "<br><b>Multiplication is ".$a*$b."</b>";
        echo "<br><b>Division is ".$a/$b."</b>";
    ?>
    <h1>PHP Array</h1>
    <?php

        $fruits = array("Mango","Orange","Apple");
        $arrLen = count($fruits);
        echo "Total items are ".$arrLen;
        echo "<br>Array elements are ".$fruits[0].",".$fruits[1]." $fruits[2].";
        echo "<br><h3>Indexed array!</h3>";
        for ($i=0; $i < $arrLen ; $i++) {
            echo $fruits[$i];
            echo "<br>";
        }
    ?>
    <?php
        echo "<h3>Associative Array</h3>";
        $person = array("Ram"=>"12","Hari"=>"45","Shyam"=>"30");
        echo "Ram is ".$person['Ram']." years old.";
        echo "<br>";
        foreach ($person as $name => $age) {
            echo $name." is ".$age." years old.";
            echo "<br>";
        }
    ?>
    <?php
        echo "<h3>Multi-Dimensional Array</h3>";
        $info = array(
            array("Suresh","Kathmandu",23),
            array("Keshav","Kanchanpur",25),
            array("Ramesh","Kailali",28)
        );
```

```

        echo $info[0][0]." from ".$info[0][1]." is ".$info[0][2]." years old.<br>";
        for ($row=0; $row < 3; $row++) {
            echo "Information of row number ".$row;
            echo "<ul>";
            for ($col=0; $col < 3 ; $col++) {
                echo "<li>";
                echo $info[$row][$col]."<br>";
                echo "</li>";
            }
            echo "</ul>";
        }
    ?>
</body>
</html>

```

### PHP Function:

- A function is a block of statements and is reusable in a program.
- A function will not execute immediately when a page loads.
- A function will be executed by a call to the function.
- A user-defined function declaration starts with the word *function*:

#### Syntax:

```

function functionName() {
    //code to be executed
}

```

#### Example:

```

<!DOCTYPE html>
<html>
<head>
    <title>Basic about PHP</title>
</head>
<body>
<?php
    echo "Php function<br>";
    function firstFun($name){
        echo "Given name is <b>".$name."</b>";
        echo "<br>";
    }
    firstFun('Ram');
    firstFun('Shyam');
    echo "Return Function<br>";
    function secondFun(int $len=50):int{
        return 5*$len;
    }
    echo "Value returned ".secondFun(6);
    echo "Value Multiplied ".secondFun(4);
    ?>
</body>
</html>

```

## Lab 11: String Function, *each()* function, Form Handling and Cookies and Session

**Objective:** To be familiar with String Method, *each()* function and Form Handling in PHP.

**Theory:**

**String Function in PHP:**

- *strlen()* – returns the length of string.
- *str\_word\_count()* – function counts the number of words in a string.
- *strrev()* – reverse the string.
- *strpos()* – function searches for a specific text within a string.
- *str\_replace()* – function replaces some characters with some other characters in a string.
- *strcmp()* – The strcmp() function compares two strings.

**Example:**

```
<!DOCTYPE html>
<html>
<body>
<?php
    echo strlen("Hello world!");
    echo str_word_count("Hello world!");
    echo strrev("Hello world!");
    echo strpos("Hello world!", "world");
    echo str_replace("world", "Dolly", "Hello world!");
    echo strcmp("Hello world!", "Hello world!");
?>
    <p>If strcmp() function returns 0, the two strings are equal.</p>
</body>
</html>
```

**each() Function in PHP**

- *each()* function returns current element key and value and moves the internal pointer forward .

**Example:**

```
<?php
    $student=array("Ram","Shyam","Rabin","Sugam");
    echo "Current item is:".current($student)."<br/>";
    echo "Next item is:".next($student)."<br/>";
    echo "Last item is:".end($student)."<br/>";
    echo "Previous item is:".prev($student)."<br/>";
    echo "Current item is:".current($student)."<br/>";
?>
<?php
    $marks=array("Amrit"=>50,"Shyam"=>48,"Rabin"=>38,"Sugam"=>49);
    while ($student=each($marks)) {
        $name=$student["key"];
        $mark=$student["value"];
        echo "$name"." obtained $mark "."mark in Web<br/>";
    }
?>
```

**Form Handling in PHP**

An example below shows simple form and make response after submission.

**Example:****<!-- formHandling.php -->**

```

<!DOCTYPE html>
<html>
<head>
    <title>Form Handling</title>
</head>
<body>
    <form method="POST" action="formResponse.php" target="_blank">
        <fieldset>
            <legend>Fill Form below and Submit</legend>
            Name:<input type="text" name="nam"><br/><br/>
            Address:<input type="text" name="add"><br/><br/>
            <input type="submit" name="sub" value="Save">
        </fieldset>
    </form>
</body>
</html>

```

**<!-- formResponse.php -->**

```

<!DOCTYPE html>
<html>
<head>
    <title>Response after Form submitted</title>
</head>
<body>
    <?php
        echo "<h3>Thank you!</h3>";
        $name=$_POST['nam'];
        $address=$_POST['add'];
        echo "Your name is <b>".$name."</b> and you are from <b>".$address."</b>.";
    ?>
</body>
</html>

```

**Cookies and Session in PHP**

Cookies are text files stored on the client computer and they are kept of use tracking purpose. PHP transparently supports HTTP cookies.

**Example:**

```

<!DOCTYPE html>
<?php
    $cookie_name = "user";
    $cookie_value = "Santosh Bist";
    setcookie($cookie_name, $cookie_value, time() + (60 * 1), "/");
    // 86400 = 1 day
?>
<html>
<body>
    <?php
        if(!isset($_COOKIE[$cookie_name])) {
            echo "Cookie named '" . $cookie_name . "' is not set!";
        } else {
            echo "Cookie '" . $cookie_name . "' is set!<br>";
        }
    }

```



```

        echo "Value is: " . $_COOKIE[$cookie_name];
    }
?>
    <p><strong>Note:</strong> You might have to reload the page to see the value of the cookie.</p>
</body>
</html>

```

## Session in PHP

- PHP Session is an alternative way to make data accessible across the various pages of an entire website.
- A session creates a file in a temporary directory on the server where registered session variables and their values are stored.

### Example:

#### <!-- phpSession.php -->

```

<?php
    session_start();
    if( isset( $_SESSION['counter'] ) ) {
        $_SESSION['counter'] += 1;
    }else {
        $_SESSION['counter'] = 1;
    }

    $msg = "You have visited this page ". $_SESSION['counter'];
    $msg .= " in this session.";
?>
<html>
    <head>
        <title>Setting up a PHP session</title>
    </head>
    <body>
        <?php echo ( $msg ); ?>
    </body>
</html>

```

## Lab 12: MySQL in PHP

Database is used to store information and can manipulate them in manageable ways.

### Example: Create Database

**<!-- createDatabase.php -->**

```
<?php
    $host='localhost';
    $user='root';
    $pass="";
    $con=mysqli_connect($host,$user,$pass);
    if (!$con) {
        die("Connection failed: " . mysqli_connect_error());
    } //database connection
    $sql = "CREATE DATABASE demoDatabase"; //create database
    if (mysqli_query($con, $sql)) {
        echo "Database created successfully";
    } else {
        echo "Error creating database: " . mysqli_error($con);
    }
    mysqli_close($con);
?>
```

### Example: Create Table

**<!-- createTable.php -->**

```
<?php
    $host='localhost';
    $user='root';
    $pass="";
    $db='demodatabase';
    $con=mysqli_connect($host,$user,$pass,$db);
    if(!$con){
        echo "Not connected".mysqli_connect_error();
    }
    $sql="CREATE TABLE STUDENT(id INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY,name
VARCHAR(20) NOT NULL,address VARCHAR(30) NOT NULL,email VARCHAR(50))"; //create table (STUDENT)
    if(mysqli_query($con,$sql)){
        echo "Table created successfully";
    }
    else{
        echo "Unsuccessful".mysqli_error($con);
    }
    mysqli_close($con);
?>
```

### Example: Delete Database

**<!-- deleteDatabase.php -->**

```
<?php
    $host='localhost';
    $username="root";
    $pass="";
    $con=mysqli_connect($host,$username,$pass);
    if (!$con) {
        echo "Not connected!".mysqli_connect_error();
    }
}
```

```

$sql="DROP DATABASE test";
if(mysqli_query($con,$sql)){
    echo 'Database deleted successfully!';
}
else{
    echo "Error deleting Database.".mysqli_error($con);
}

```

?>

### Example: Insert Data in table

<!-- insertData.php -->

```

<?php
    $server='localhost';
    $user='root';
    $pass='';
    $db='demodatabase';
    $con=mysqli_connect($server,$user,$pass,$db);
    if(!$con){
        echo "Not connected".mysqli_connect_error();
    }
    $sql="INSERT INTO STUDENT(name,address,email) VALUES ('hari','kathmandu','hari98@gmail.com')";
    if(mysqli_query($con,$sql)){
        echo "Data inserted in STUDENT table.";
    }
    else{
        echo "Error: ".$sql(mysqli_error($con));
    }
    mysqli_close($con);

```

?>

### Example: Insertion of data to database through form

<!-- dataForm.php -->

```

<?php include('dataFromForm.php')?>
<!DOCTYPE html>
<html>
<head>
    <title>Fill up the form below</title>
</head>
<body>
    <form method="POST">
        <fieldset>
            <legend>Can you please fill the form</legend>
            Name:<br/>
            <input type="text" name="nam" placeholder="Enter your name" required=""><br/>
            Email:<br/>
            <input type="text" name="email" placeholder="Enter your email" required=""><br/>
            Address:<br/>
            <input type="text" name="address" placeholder="enter your address"
required><br/><br/>
            <input type="submit" name="submit" value="Submit">
        </fieldset>
    </form>
</body>
</html>

```

```
<!-- dataFromForm.php -->
```

```
<?php
```

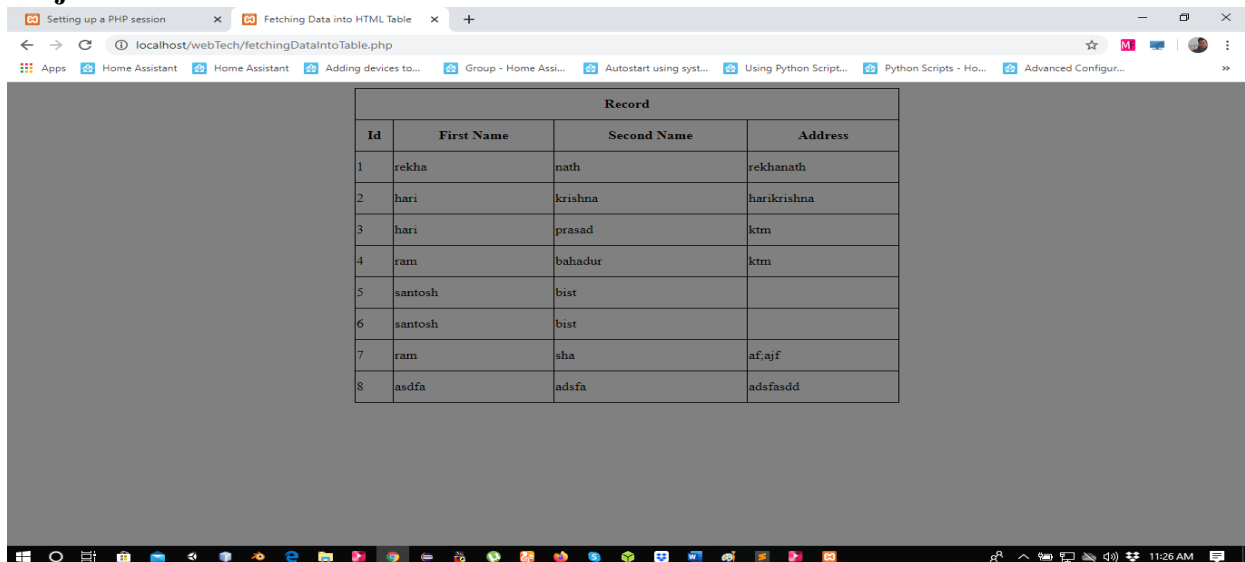
```
    if(isset($_POST['submit'])){
        $name=$_POST["nam"];
        $email=$_POST["email"];
        $address=$_POST["address"];

        $server='localhost';
        $user='root';
        $pass="";
        $db='demodatabase';

        $con=mysqli_connect($server,$user,$pass,$db);
        if(!$con){
            echo "Not Connected :".mysqli_connect_error();
        }
        $sql="INSERT INTO TEACHER(name,email,address) VALUES
('".$name."','".$email."','".$address."')";
        if(mysqli_query($con,$sql)){
            echo "Data inserted Successfully!";
        }
        else{
            echo "Error occured:".$sql(mysqli_error($con));
        }
        mysqli_close($con);
    }
?>
```

**Assignment:** The following assignment is named as Project 8, so you need to complete it and present your output on next week.

### Project 8:



Record			
Id	First Name	Second Name	Address
1	rekha	nath	rekhanath
2	hari	krishna	harikrishna
3	hari	prasad	ktm
4	ram	bahadur	ktm
5	santosh	bist	
6	santosh	bist	
7	ram	sha	af,ajf
8	asdfa	adsfa	adsfasdd

### Description:

Fetch data from database and display them in Table format , as shown in above image.

Thank You 😊