Assignment No. 1

1. **Title: -**

   Website Case Study

2. **Objectives: -**

   Understand about different Website designs and its issues.

3. **Problem Statement: -**

   Case study: Before coding of the website, planning is important, students should visit different websites (Min. 5) for the different client projects and note down the evaluation results for these websites, either good website or bad website in following format:

Sr. No.	Website URL	Purpose of Website	Things Liked in the website	Things disliked in the website	Overall evaluation of the website (Good/Bad)

   From the evaluation, students should learn and conclude different website design issues, which should be considered while developing a website.

4. **Problem Outcome: -**

   Students will be able to understand

   1. Find out different Website Designs
   2. Find out various issues in various websites.
   3. How to design a fine good-looking website.

5. **Hardware & Software Requirement: -**

Hardware Requirements	Software Requirements
At least 64 MB RAM, 350 MB Free fixed disk space	Windows/ Linux OS, web browser

6. **Theory Concepts: -**

   Parameters to check Website Quality:

   1. Site Purpose
   2. Target Audience
   3. Responsive/Mobile Friendly
   4. Fresh Content or Outdated Content
   5. Using understandable language on the web pages
   6. Visual Design/Quality of Images
   7. Look & Feel (using uniform look & feel)
   8. Engaging Content (Making the site interesting)
   9. Navigation (Making the site easy to use)
   10. Contents Representation (Font size/Style)
   11. Download/ Upload speed

7. **Design/Execution Steps**

   Before coding of the website, planning is important, students should visit different websites (Min. 5) for the different client projects and note down the evaluation results for these websites, either good website or bad website in following format:

Sr. No.	Website URL	Purpose of Website	Things Liked in the website	Things disliked in the website	Overall evaluation of the website (Good/Bad)

8. **Conclusion:**

   Hence, Students has learned what are different design issues & how to design good Website.

9. **Viva Questions:**

   a. What are different parameters to find good quality of website?

   b. What are different web design issues?

   c. What are the advantages of websites?

   d. What are steps for website Designing?

   e. If you are told to design website which website design template you refer from this assignment? Why?

Assignment No. 2

1. **Title: -**

   Design a static website using HTML, CSS & Bootstrap.

2. **Objectives: -**

   Understand basic concepts of HTML, CSS & Bootstrap.

3. **Problem Statement: -**

   Implement a web page index.htm for any client website (e.g., a restaurant website project) using following:

   a. HTML syntax: heading tags, basic tags and attributes, frames, tables, images, lists, links for text and images, forms etc.

   b. Use of Internal CSS, Inline CSS, External CSS.

4. **Problem Outcome: -**

   Students will be able to understand

   1. Design static webpages using HTML.
   2. Apply CSS on web sites.
   3. Apply CSS Bootstrap on web sites.

5. **Hardware & Software Requirement: -**

Hardware Requirements	Software Requirements
At least 64 MB RAM, 350 MB Free fixed disk space	Windows/ Linux OS, Notepad or any web code editor, web browser

6. **Theory Concepts: -**

   a. What is HTML?

   - HTML: HTML is the standard markup language for creating Web pages.
   - HTML stands for Hyper Text Markup Language
   - HTML describes the structure of Web pages using markup
   - HTML elements are the building blocks of HTML pages
   - HTML elements are represented by tags
   - HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
   - Browsers do not display the HTML tags, but use them to render the content of the page

   b. What is CSS?

   - CSS stands for Cascading Style Sheet. It is nothing, but design language intended to simplify the process of making web pages presentable.CSS handles the feel and look part of a web page.By using CSS, one can control the color of text, style of fonts, spacing between paragraphs, layout designs.

- CSS is easy to learn, easy to understandand it provides powerful control on presentation of an HTML document.
- **Advantages of CSS**: It saves the time, Pages load faster, Easy maintenance, Superior styles to HTML, Multiple Device Compatibility, Global web standards, Offline Browsing, Platform Independence.

c. What TECHNOLOGY / TOOL has used?

- 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
- HTML tags are element names surrounded by angle brackets:
  <tagname>content goes here...</tagname>
- CSS can be added to HTML elements in 3 ways:
- **Inline -** by using the style attribute in HTML elements. An inline CSS is used to apply a unique style to a single HTML element.
- Example.
  <h1 style="color:blue;">This is a Blue Heading</h1>
- **Internal -** by using a <style> element in the <head> section. An internal CSS is used to define a style for a single HTML page. An internal CSS is defined in the <head> section of an HTML page, within a <style> element.
- Example:

```
<style>
        body {background-color: powderblue;}
        h1   {color: blue;} p    {color: red;}
</style>
```

- **External -** by using an external CSS file. An external style sheet is used to define the style for many HTML pages. With an external style sheet, you can change the look of an entire web site, by changing one file!To use an external style sheet, add a link to it in the <head> section of the HTML page.
- **Example:** <link rel="stylesheet" href="styles.css">
- Use the HTML <head> element to store <style> and <link> elements
- Use the CSS color property for text colors
- Use the CSS font-family property for text fonts
- Use the CSS font-size property for text sizes
- Use the CSS border property for borders
- Use the CSS padding property for space inside the border
- Use the CSS margin property for space outside the border

7. DESIGN / EXECUTION STEPS

Following steps are used to Create and Execute web applications,

1) Write the HTML code in notepad/VS Code editor and save with .html extension.
2) Write the CSS code in notepad and save with .css extension.
3) Import CSS file in HTML page. Also import Bootstrap file from either way.
4) Add styles to HTML page elements through CSS & Bootstrap styles.
5) Open HTML page in the browser.

8. Implementation: -

Program Code

Login.html

```
<html>
   <head>
      <link rel="stylesheet" type="text/css" href="Stylesheet1.css">
      <title>Login Form</title>
   </head>
   <body onload="document.Login.UN.focus();">
      <form name="Login" onsubmit="valLogin();">
         <table border="2" width="90%" align="center" bgcolor="cyan">
            <tr align="center">
               <td><img src="logo.jpg" width="150" height="40"></td>
               <td><h1>Login Form</h1></td>
            </tr>
            <tr>
               <td colspan="3"><marquee><a href="Registration.html">If Not
               Registered Click Here</a></marquee></td>
            </tr>
         </table>
         <table width="90%" border="2" align="center" bgcolor="pink">
            <tr>
               <td><b>   &nbspUser Name</b></td>
               <td>    <input type="text" placeholder="Enter
               User Name" name="uid" size="25"required/></td>
            </tr>
            <tr>
               <td><b>   &nbspPassword</b></td>
               <td>    <input type="password"
               placeholder="Enter Password" name="pwd" size="25" required/></td>
            </tr>
            <tr align="center">
               <td><input type="submit" value="Login" class="ab" name="submit"
               onclick="Loginval()"></td>
               <td><input type="Reset" value="Cancel" class="ab"
               name="reset"></td>
            </tr>
         </table>
      </form>
   </body>
</html>
```

Register.html

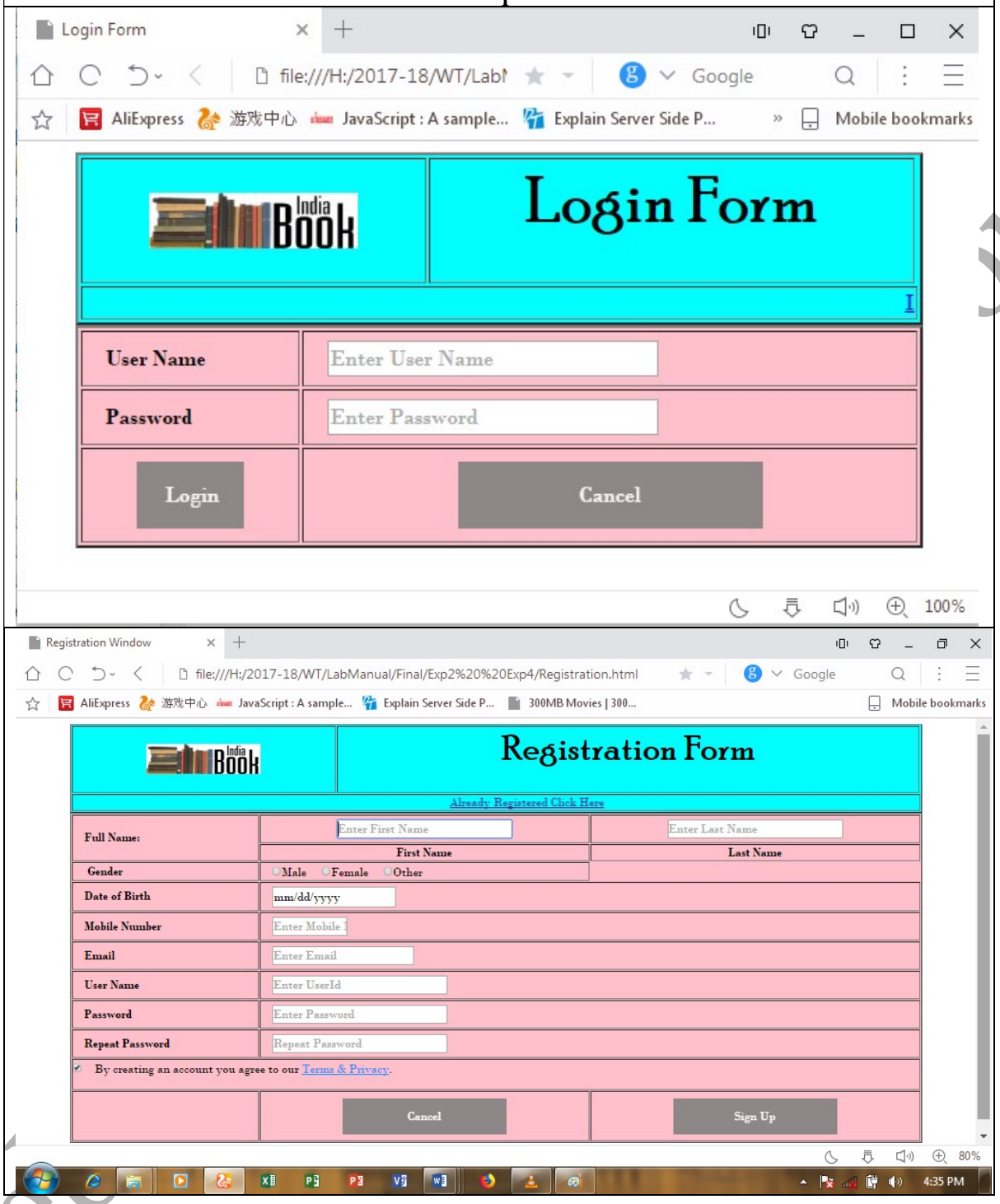
```
<html>
   <head>
      <link rel="stylesheet" type="text/css" href="Stylesheet1.css">
      <title>Registration Window</title>
   </head>
   <body>
      <form name='Registration'>
         <table border="2" width="90%" align="center" bgcolor="cyan">
            <tr align="center">
               <td><img src="logo.jpg" width="150" height="40"></td>
               <td><h1>Registration Form</h1></td>
            </tr>
            <tr>
               <td colspan="3"><marquee><a href="Login.html">Already Registered Click
               Here</a></marquee></td>
            </tr>
         </table>
         <table width="90%" border="2" align="center" bgcolor="pink">
            <tr>
               <td rowspan="2"><b>   &nbspFull Name: </b></td>
               <td align="center"><input type="text" name="fname" maxlength="50"
               size="25" placeholder="Enter First Name"required></td>
               <td align="center"><input type="text" name="lname" maxlength="50"
               size="25" placeholder="Enter Last Name"required></td>
            </tr>
            <tr align="center">
               <td><b>First Name</b></td>
               <td><b>Last Name</b></td>
            </tr>
            <tr>
               <td ><b>     Gender</b></td>
               <td>    <input type="radio" name="sex"
               value="Male"/>Male     <input type="radio" name="sex"
               value="Female"/>Female     <input type="radio"
               name="sex" value="Other"/>Other</td>
            </tr>
            <tr>
               <td><b>   &nbspDate of Birth</b></td>
               <td colspan="2">    <input type="Date"
               name="DOB"/></td>
            </tr>
            <tr>
               <td><b>   &nbspMobile Number</b></td>
               <td colspan="2">    <input type="text"
               placeholder="Enter Mobile Number" name="mono" size="10"
               maxlength="10"/></td>
            </tr>
            <tr>
               <td><b>   &nbspEmail</b></td>
               <td colspan="2">    <input type="text"
               placeholder="Enter Email Id" name="email"/></td>
            </tr>
            <tr>
               <td ><b>   &nbspUser Name</b></td>
```

```
                <td colspan="2">    <input type="text"
                placeholder="Enter UserId" name="uid" size="25"/></td>
            </tr>
            <tr>
                <td><b>   &nbspPassword</b></td>
                <td colspan="2">    <input type="password"
                placeholder="Enter Password" name="pwd" size="25"/></td>
            </tr>
            <tr>
                <td><b>   &nbspRepeat Password</b></td>
                <td colspan="2">    <input type="password"
                placeholder="Repeat Password" name="rpwd" size="25"/></td>
            </tr>
            <tr>
                <td colspan="3"><input type="checkbox" checked="checked"
                required/>   &nbspBy creating an account you agree to our
                <a href="#" style="color:dodgerblue">Terms & Privacy</a>.</p></td>
            </tr>
            <tr align="center">
                <td></td>
                <td><input type="Reset" value="Cancel" class="ab" name="reset"></td>
                <td><input type="submit" value="Sign Up" name="submit"class="ab"
                onclick="valid()"/></td>
            </tr>
        </table>
    </form>
</body>
</html>
```

Stylesheet.css

```
h1{
    font-family:Poor Richard;
    font-size:300%;
}
td{
    font-family:Bodoni MT;
    font-size:100%;
}
input
{
    font-family:Bodoni MT;
    font-size:100%;
    margin:5px 0;
}
.ab{
    background-color: Gray;
    color: white;
    padding: 14px 20px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: 50%;
    opacity: 0.9;
}
```

Output



9. Conclusion: -

From above assignment I have conclude that: _____

_____

_____

10. Viva Questions:

a. What is the difference between HTML and HTML5?
b. What is the difference between html elements and tags?
c. What is marquee?
d. What is the use of span tag? Give an example?
e. What is the use of 'required 'attribute in HTML5?
f. What is External stylesheet? What are the advantages and disadvantages?
g. What is CSS selector?
h. What are the components of CSS style?
i. What are browser safe color?

Experiment No. 4

1. **Title: -**

   Design a static website using HTML & JavaScript.

2. **Objectives: -**

   a. Understand basic concepts of HTML & JavaScript.

   b. Understand how to validate the various fields with JavaScript.

3. **Problem Statement: -**

   Implement an application in Java Script using following:

   a. Design UI of application using HTML, CSS etc.

   b. Include Java script validation

   c. Use of prompt and alert window using Java Script

4. **Problem Outcome: -**

   Students will be able to understand

   a. Design static webpage using HTML.

   b. Apply JavaScript to HTML pages for validation of data.

5. **Hardware & Software Requirement: -**

Hardware Requirements	Software Requirements
At least 64 MB RAM, 350 MB Free fixed disk space	Windows/ Linux OS, Notepad or any web code editor, web browser

6. **Theory Concepts: -**

   a. What is JavaScript?

- JavaScript is most commonly used as a client-side scripting language. This means that JavaScript code is written into an HTML page.
- When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it's up to the browser to do something with it.
- JavaScript was designed to add interactivity to HTML pages. JavaScript is a scripting language.
- A scripting language is a lightweight programming language. A JavaScript consists of lines of executable computer code
- A JavaScript is usually embedded directly into HTML pages. JavaScript is an interpreted language (means that scripts execute without preliminary compilation)
- Everyone can use JavaScript without purchasing a license.
- As JavaScript is integrated with HTML it is very easy to implement. It is open as well as cross-platform.
- The advantages of using JavaScript are –
  - It requires less server interaction

- Immediate feedback to the visitors
- Increased interactivity
- Richer interfaces

b. JavaScript Regular Expressions

- A regular expression is a sequence of characters that forms a search pattern.
- The search pattern can be used for text search and text replace operations.
- When you search for data in a text, you can use this search pattern to describe what you are searching for. A regular expression can be a single character, or a more complicated pattern. Regular expressions can be used to perform all types of text search and text replace operations.
- **Syntax**
  /pattern/modifiers;
- **Example**
  /ait27/i;
- **Example explained:** /w3schools/i is a regular expression. Ait27 is a pattern (to be used in a search). i is a modifier (modifies the search to be case-insensitive).

c. Validation:

- When client enters the all necessary data and press the submit button form validation is done at server side If data entered by a client is incorrect or missing, the server needs to send all data back to the client and request for resubmission of form with correct information. This is really a lengthy process which puts a lot of load(burden) on the server.
- So, JavaScript provides a way to validate form's data on the client's side itself before sending it to the web server. Form validation performs two functions-
  - Basic Validation –First of all the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for the data.
  - Data Format Validation – Secondly, the data that is entered must be checked for correct format and its value. The code must include appropriate logic to test correctness of data.

d. What can a JavaScript Do?

- JavaScript gives HTML designers a programming tool - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages
- JavaScript can put dynamic text into an HTML page - A JavaScript statement like this:

  document.write("<h1>" + name + "</h1>")

  can write a variable text into an HTML page
- JavaScript can react to events - A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element
- JavaScript can read and write HTML elements - A JavaScript can read and change the content of an HTML element

- JavaScript can be used to validate data - A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing
- JavaScript can be used to detect the visitor's browser - A JavaScript can be used to detect the visitor's browser, and - depending on the browser - load another page specifically designed for that browser

e. What Technology/ Tools are used?

- JavaScript can be implemented using JavaScript statements that are placed within the <script>.
- You can place the <script> tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the <head> tags.
- The script tag takes two important attributes:
  - Language − This attribute specifies what scripting language you are using. Typically, its value will be JavaScript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.
  - Type − This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".
- Another way to use JavaScript is by importing external JavaScript file. It is most preferred approach to keep the JavaScript in an external file with file extension .js, and reference it via the src(source) attribute as follows:

  <script src=" JavaScriptfilename.js"></script>

  Here, even though no content is present still the closing tag </script> is needed.

7. Design/Execution Steps: -

Following steps are used to Create and Execute web applications,

1. Write an HTML code in notepad and save with .html extension.
2. Write a JavaScript code in notepad & save with .js extension.
3. Import the JavaScript file in HTML file.
4. Write the function for validation of email id and phone no and enclosed this function in script file.
5. Call this function on 'onClick' event of submit button.
6. Open HTML page in the browser.

8. Implementations: -

Register.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
 <head>
        <link rel="stylesheet" type="text/css" href="Stylesheet1.css">
  <title>Registration Window</title>
        <script src="RegistrationValidate.js" language="JavaScript"></script>
 </head>
 <body onload="document.Registration.fname.focus();">
```

```
<form name='Registration'>
<table border="2" width="90%" align="center" bgcolor="cyan">
<tr align="center">
<td><img src="logo.jpg" width="150" height="40"></td>
<td><h1>Registration Form</h1></td>
</tr>
<tr><td colspan="3"><marquee><a href="Login.html">Already
Registered Click Here</a></marquee></td></tr></table>
<table width="90%" border="2" align="center" bgcolor="pink">
<tr>
<td rowspan="2"><b>   &nbspFull
Name: </b></td>
<td align="center"><input type="text" name="fname"
maxlength="50" size="25" placeholder="Enter First Name"required></td>
<td align="center"><input type="text" name="lname"
maxlength="50" size="25" placeholder="Enter Last Name"required></td>
</tr>
<tr align="center">
<td><b>First Name</b></td>
<td><b>Last Name</b></td>
</tr>
<tr>
<td ><b>    
Gender</b></td>
<td>    <input type="radio"
name="sex" value="Male"/><span>Male</span>
    <input type="radio"
name="sex" value="Female"/><span>Female</span>
    <input type="radio"
name="sex" value="Other"/><span>Other</span></td>
</tr>
<tr>
<td><b>   &nbspDate of
Birth</b></td>
<td colspan="2">    <input
type="Date" name="DOB" required/></td>
</tr>
<tr>
<td><b>   &nbspMobile Number</b></td>
<td colspan="2">    <input type="text"
placeholder="Enter Mobile Number" name="mono" size="10"
maxlength="10"required/></td>
</tr>
<tr>
<td><b>   &nbspEmail</b></td>
```

```
                <td colspan="2">    <input type="text"
placeholder="Enter Email" name="email" required/></td>
                        </tr>
                        <tr>
                        <td ><b>   &nbspUser Name</b></td>
                <td colspan="2">    <input type="text"
placeholder="Enter UserId" name="uid" size="25" required/></td>
                        </tr>
                        <tr>
                                <td><b>   &nbspPassword</b></td>
                                <td colspan="2">    <input
type="password" placeholder="Enter Password" name="pwd" size="25"
required/></td>
                        </tr>
                <tr>
                        <td><b>   &nbspRepeat Password</b></td>
                        <td colspan="2">    <input
type="password" placeholder="Repeat Password" name="rpwd" size="25"
required/></td>
                </tr>
                <tr>
                        <td colspan="3"><input type="checkbox" checked="checked"
required/>   &nbspBy creating an account you agree to our <a
href="#" style="color:dodgerblue">Terms & Privacy</a>.</p></td>
                </tr>
                <tr align="center">
                        <td></td>
                        <td><input type="Reset" value="Cancel" class="ab"
name="reset"></td>
                        <td><input type="submit" value="Sign Up"
name="submit"class="ab" onclick="valid()"></td>
                </tr>
        </form>
  </body>
</html>
```

CalStyle.css

```
h1{
font-family:Poor Richard;
font-size:300%;
}
td{
font-family:Bodoni MT;
font-size:100%;
}
input
```

```
{
font-family:Bodoni MT;
font-size:100%;
margin:5px 0;
}
.ab{
background-color: Gray;
  color: white;
  padding: 14px 20px;
  margin: 8px 0;
  border: none;
  cursor: pointer;
  width: 50%;
  opacity: 0.9;
}
```

RegisterValidation.js

```
function valid()
{
    var uid=document.Registration.uid.value;
    var email=document.Registration.email.value;
    var sex=document.Registration.sex.value;
    var mobo=document.Registration.mono.value;
    var pwd=document.Registration.pwd.value;
    var rpwd=document.Registration.rpwd.value;
    var dat=document.Registration.DOB.value;
    var fname=document.Registration.fname.value;
    var lname=document.Registration.lname.value;
    if(valname(fname))
    {
        if(valname(lname))
        {
            if(valsex(sex))
            {
                if(valdate(dat))
                {
                    if(valmobo(mobo))
                    {
                        if(valemail(email))
                        {
                            if(valuid(uid,7,12))
                            {
                                if(valpass(pwd))
                                {
                                    if(valrpass(pwd,rpwd))
                                    {
                                        alert("User Registered
Successfully.");
```

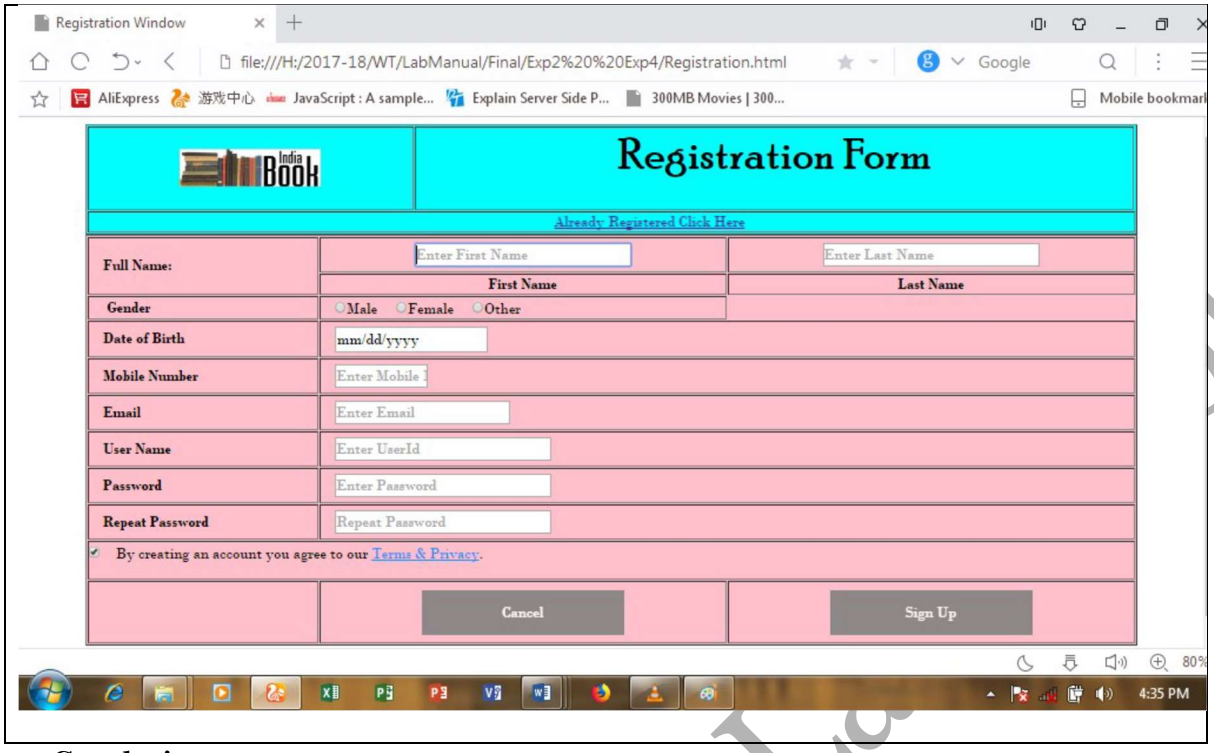
```
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}
function valname(name)
{
    var letters=/^[A-Za-z]+$/;
    if(name.match(letters))
        return true;
    else
    {
        alert("Name must have alphabet characters only.");
        return false;
    }
}
function valdate(dat)
{
    if(dat.length==0)
    {
        alert("Select Date of Birth");
        return false;
    }
    else
        return true;
}
function valrpass(pwd,rpwd)
{
    if(rpwd.length==0)
    {
        alert("Confirm Password field doesnot empty.");
        return;
    }
    if(rpwd==pwd)
        return true;
    else
    {
        alert("Confirm Password doesnot Matched.");
        return false;
    }

}
function valpass(pwd)
```

```
{
     if(pwd.length < 8)
     {
        alert("Error: Password must contain at least 8 characters!");
        return false;
     }
     re = /[0-9]/;
     if(!re.test(pwd))
     {
        alert("Error: password must contain at least one number (0-9)!");
        return false;
     }
     re = /[a-z]/;
     if(!re.test(pwd))
     {
        alert("Error: password must contain at least one lowercase letter
(a-z)!");
        return false;
     }
     re = /[A-Z]/;
     if(!re.test(pwd))
     {
        alert("Error: password must contain at least one uppercase letter
(A-Z)!");
        return false;
     }
     return true;
}
function valmobo(mobo)
{
    var numbers=/^[0-9]+$/;
    if(mobo.length==10)
    {
        if(mobo.match(numbers))
            return true;
        else
        {
            alert("Mobile Number should be in Number format only.");
            return false;
        }
    }
    else
    {
        alert("Invalid Mobile number");
        return false;
    }
}
function valsex(sex)
```

```
{
    var sexlen=sex.length;
    if(sexlen==0)
    {
        alert("Select sex.");
        return false;
    }
    return true;
}
function valemail(email)
{
    var mformat=/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;
    if(email.match(mformat))
    {
        return true;
    }
    else
    {
        alert("Invalid E-mail ID");
        return false;
    }
}
function valuid(uid,mn,mx)
{
    var uidlen=uid.length;
    if(uidlen==0)
    {
        alert("User Id should not be empty.");
        return false;
    }
    else if((uidlen>=mx)||(uidlen<mn))
    {
        alert("User Id Length should be inbetween "+mn+" to "+mx+" only.");
        return false;
    }
    return true;
}
```

Output



9. Conclusion: -

From above assignment I have conclude that:_____

_____

10. Review Questions:

A) Programming Questions

a. Write a program to design registration form for students by using HTML, CSS& Java Script and perform following validations: all fields mandatory, phone number and email address validation.

b. Write a program to design the EbookShop (book_id, book_title, book_author, book_price, quantity, Customer_name, Customer_email_address, Customer_mobile_no) Bill Entry form by using HTML, CSS, Bootstrap & JavaScript and perform the following validations: all fields mandatory, book_id, book_price, book_id, Customer_email_address, Customer_mobile_no validation.

B) Viva-Questions for Oral Preparation:

a. What are the features of JavaScript? Why JavaScript is so famous?
b. What is a name function in JavaScript & how to define it?
c. What is the difference between Attributes and Property?
d. What are the ways to define a variable in JavaScript?
e. Name some of the JavaScript Frameworks
f. What is the difference between window & document in JavaScript?
g. How can you convert the string of any base to integer in JavaScript?
h. What is a prompt box in JavaScript?
i. How to empty an Array in JavaScript?
j. How to use external JavaScript file?
k. What is the use of window object?

l. Difference between Client-side JavaScript and Server-side JavaScript?
m. How to change the background color of HTML document using JavaScript?

Assignment No. 4

1. **Title: -**

   Design a static website using XML & CSS/XSLT.

2. **Objectives: -**

   Understand basic concepts of XML & XSLT.

3. **Problem Statement: -**

   Design the XML document to store the information of the Employees of any business organization and demonstrate the use of

   a. DTD

   b. XML Schema

   And display the content in tabular format by using CSS/XSL.

4. **Problem Outcome: -**

   Students will be able to understand

   a. Understand the Document type Definition concept.

   b. Design static webpage using XML.

   c. Apply XSL to XML pages.

5. **Hardware & Software Requirement: -**

Hardware Requirements	Software Requirements
At least 64 MB RAM, 350 MB Free fixed disk space	Windows/ Linux OS, Notepad or any web code editor, web browser

6. **Theory Concepts: -**

   a. **What is XML?**

- XML stands for Extensible Markup Language. It is nothing but the text-based markup language which is derived from Standard Generalized Markup Language (SGML).
- XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML soon, but it introduces new possibilities by adopting many successful features of HTML.
- There are three important characteristics of XML that make it useful in a variety of systems and solutions –
  - XML is extensible – XML allows you to create your own self-descriptive tags, or language, that suits your application.
  - XML carries the data, does not present it – XML allows you to store the data irrespective of how it will be presented.

- XML is a public standard – XML was developed by an organization called the Worldwide
- Web Consortium (W3C) and is available as an open standard.
- XML documents can have a reference to a DTD or to an XML Schema.
- There are some advantages of XML as:
  - Uses human, not computer language. XML is readable & understandable even by novice, & no more difficult to code than HTML.
  - XML is completely compatible with Java & 100% portable. Any application that can process XML can use your information, regardless of platform.

b. What Technology/Tools can be used?

- The XML document have an XML declaration, but it is optional, and it is written as–

```
<? xml version = "1.0" encoding = "UTF-8"?>
```

- Where version is nothing but the version of an XML document and UTF specifies the character- encoding used in the document.
- Each XML-element needs to be closed either with start or with end elements as shown below –
- **<element>……</element>**: An XML document can have only one root element.

```
<root>
    <x>...</x>
    <y>...</y>
</root>
```

- **XML Attributes:**
  - Using a name/value pair, an attribute specifies a single property for an element. An XML element can have one or more attributes.
  - For example – XMLTutorial
  - Here href is the attribute name and http://www.google.com/ is attribute value.

c. XSLT:

- XSLT stands for XSL Transformations
- XSLT is the most important part of XSL
- XSLT transforms an XML document into another XML document
- XSLT uses XPath to navigate in XML documents
- XSLT is a W3C Recommendation.
- This assignment will teach you how to use XSLT to transform XML documents into other formats (like transforming XML into HTML).
- XSLT is used to transform an XML document into another XML document, or another type of document that is recognized by a browser, like HTML and XHTML. Normally XSLT does this by transforming each XML element into an (X)HTML element.
- With XSLT you can add/remove elements and attributes to or from the output file. You can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more.
- A common way to describe the transformation process is to say that XSLT transforms an XML source-tree into an XML result-tree.

- **XSL = Style Sheets for XML**
  - XSL stands for EXtensible Stylesheet Language.
  - The World Wide Web Consortium (W3C) started to develop XSL because there was a need for an XML-based Stylesheet Language.
  - XML does not use predefined tags, and therefore the meaning of each tag is not well understood.
  - A <table> element could indicate an HTML table, a piece of furniture, or something else - and browsers do not know how to display it!
- **XSLT Uses XPath**
  - XSLT uses XPath to find information in an XML document. XPath is used to navigate through elements and attributes in XML documents.
- **How Does it Work?**
  - In the transformation process, XSLT uses XPath to define parts of the source document that should match one or more predefined templates.
  - When a match is found, XSLT will transform the matching part of the source document into the result document.
  - The root element that declares the document to be an XSL style sheet is <xsl:stylesheet> or <xsl:transform>.
  - Note: <xsl:stylesheet> and <xsl:transform> are completely synonymous and either can be used!

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

  - To get access to the XSLT elements, attributes and features we must declare the XSLT namespace at the top of the document.
  - The xmlns:xsl="http://www.w3.org/1999/XSL/Transform" points to the official W3C XSLT namespace. If you use this namespace, you must also include the attribute version="1.0".
- **The <xsl:template> Element**
  - The <xsl:template> element is used to build templates.
  - The match attribute is used to associate a template with an XML element. The match attribute can also be used to define a template for the entire XML document. The value of the match attribute is an XPath expression (i.e. match="/" defines the whole document).
- **The <xsl:value-of> Element**
  - The <xsl:value-of> element can be used to extract the value of an XML element and add it to the output stream of the transformation
- **The <xsl:for-each> Element**
  - The XSL <xsl:for-each> element can be used to select every XML element of a specified node-set

d. DTD

- A DTD is a Document Type Definition.
- A DTD defines the structure and the legal elements and attributes of an XML document.

- With a DTD, independent groups of people can agree on a standard DTD for interchanging data.
- An application can use a DTD to verify that XML data is valid.
  XML document with an internal DTD
  <?xml version="1.0"?>
  <!DOCTYPE note [
  <!ELEMENT note (to,from,heading,body)>
  <!ELEMENT to (#PCDATA)>
  <!ELEMENT from (#PCDATA)>
  <!ELEMENT heading (#PCDATA)>
  <!ELEMENT body (#PCDATA)>
  ]>
  <note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend</body>
  </note>
- The DTD above is interpreted like this:
  - **!DOCTYPE note** defines that the root element of this document is note
  - **!ELEMENT note** defines that the note element must contain four elements: "to,from,heading,body"
  - **!ELEMENT to** defines the to element to be of type "#PCDATA"
  - **!ELEMENT from** defines the from element to be of type "#PCDATA"
  - **!ELEMENT heading** defines the heading element to be of type "#PCDATA"
  - **!ELEMENT body** defines the body element to be of type "#PCDATA"
  - PCDATA means parsed character data. Think of character data as the text found between the start tag and the end tag of an XML element.
- PCDATA is text that WILL be parsed by a parser. The text will be examined by the parser for entities and markup.
- Tags inside the text will be treated as markup and entities will be expanded.
- However, parsed character data should not contain any &, <, or > characters; these need to be represented by the & < and > entities, respectively.

e. XML Schema

- An XML Schema describes the structure of an XML document.
- The XML Schema language is also referred to as XML Schema Definition (XSD).
- XML Schema is an XML-based (and more powerful) alternative to DTD.
- Many of these XML standards are defined by XML Schemas.
- The purpose of an XML Schema is to define the legal building blocks of an XML document: the elements and attributes that can appear in a document, the number of (and order of) child elements, data types for elements and attributes, default and fixed values for elements and attributes

- A well-formed XML document is a document that conforms to the XML syntax rules, like:
  - it must begin with the XML declaration
  - it must have one unique root element
  - start-tags must have matching end-tags
  - elements are case sensitive
  - all elements must be closed
  - all elements must be properly nested
  - all attribute values must be quoted
  - entities must be used for special characters
- Even if documents are well-formed, they can still contain errors, and those errors can have serious consequences.
- **XSD - The <schema> Element**
  - The <schema> element is the root element of every XML Schema.
  - The <schema> element is the root element of every XML Schema:

```
<?xml version="1.0"?>

<xs:schema>
...
...
</xs:schema>
```

  - The <schema> element may contain some attributes. A schema declaration often looks something like this:

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="https://www.w3schools.com"
xmlns="https://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>
```

  - xmlns:xs="http://www.w3.org/2001/XMLSchema" indicates that the elements and data types used in the schema come from the "http://www.w3.org/2001/XMLSchema" namespace. It also specifies that the elements and data types that come from the "http://www.w3.org/2001/XMLSchema" namespace should be prefixed with xs:
  - The fragment: targetNamespace="https://www.w3schools.com" indicates that the elements defined by this schema (note, to, from, heading, body.) come from the "https://www.w3schools.com" namespace.
  - The fragment: xmlns="https://www.w3schools.com" indicates that the default namespace is "https://www.w3schools.com".
  - The fragment: elementFormDefault="qualified" indicates that any elements used by the XML instance document which were declared in this schema must be namespace qualified.
- **XSD Simple Elements**
  - XML Schemas define the elements of your XML files.

- A simple element is an XML element that contains only text. It cannot contain any other elements or attributes.
- The syntax for defining a simple element is:

<xs:element name="xxx" type="yyy"/>

- where xxx is the name of the element and yyy is the data type of the element.
- XML Schema has a lot of built-in data types. The most common types are:
  - ✓ xs:string
  - ✓ xs:decimal
  - ✓ xs:integer
  - ✓ xs:boolean
  - ✓ xs:date
  - ✓ xs:time

7. Design/Execution Steps

- Following steps are used to Create and Execute web applications,
  1. Write the XML code in editor and save with .xml extension.
  2. Write the XSD/DTD code in editor and save with .xsd/.dtd resp. extension.
  3. Write the XSLT code in editor and save with .xsl extension.
  4. Import files (xsd/dtd, xsl) in XML page.
  5. Open XML page in the browser.

8. Implementations: -

Employee.dtd

```
<!DOCTYPE Person[
<!ELEMENT Person(Employee+)>
<!ELEMENT Employee(EmpId,EmpName,Department,Salary)>
<!ELEMENT EmpId(#PCDATA)>
<!ELEMENT EmpName(#PCDATA)>
<!ELEMENT Department(#PCDATA)>
<!ELEMENT Salary(#PCDATA)>
]>
```

Employee.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="EmployeeSchema.xsl"?>
<Person xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="Employee.xsd">
<Person>
  <Employee>
    <EmpId>1715</EmpId>
    <EmpName>Kuldeep Hule</EmpName>
    <Department>Computer Engineering</Department>
    <Salary>70000</Salary>
  </Employee>
  <Employee>
    <EmpId>1717</EmpId>
    <EmpName>Priyanka Hule</EmpName>
    <Department>Computer Engineering</Department>
    <Salary>20000</Salary>
  </Employee>
```

```
  <Employee>
    <EmpId>1720</EmpId>
    <EmpName>Seeta Yadav</EmpName>
    <Department>Computer Engineering</Department>
    <Salary>70000</Salary>
  </Employee>
  <Employee>
    <EmpId>1725</EmpId>
    <EmpName>Varsha Kulkarni</EmpName>
    <Department>First Engineering</Department>
    <Salary>16000</Salary>
  </Employee>
</Person>
```

Employee.xsl

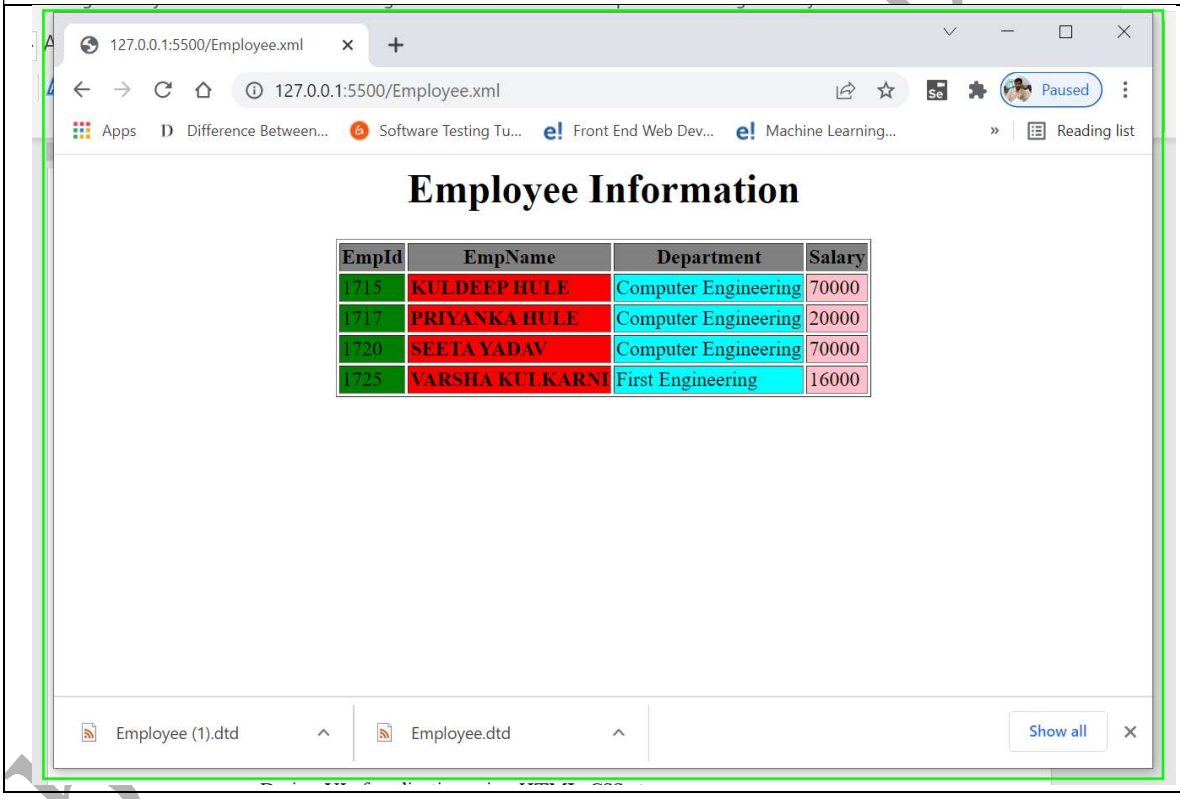
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
  <body>
    <center>
      <h1>Employee Information</h1>
      <table border="1">
        <tr bgcolor="gray">
          <th>EmpId</th>
          <th>EmpName</th>
          <th>Department</th>
          <th>Salary</th>
        </tr>
        <xsl:for-each select="Person/Employee">
          <tr>
            <td bgcolor="green"><xsl:value-of select="EmpId"/></td>
            <td bgcolor="red"><b><xsl:value-of select=
              "translate(EmpName,'abcdefghijklmnopqrstuvwxyz',
        'ABCDEFGHIJKLMNOPQRSTUVWXYZ')"/></b></td>
            <td bgcolor="cyan"><xsl:value-of select="Department"/></td>
            <td bgcolor="pink"><xsl:value-of select="Salary"/></td>
          </tr>
        </xsl:for-each>
      </table>
    </center>
  </body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Employee.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Person">
    <xs:complexType>
```

```
        <xs:sequence>
          <xs:element name="Employee" maxOccurs="unbounded">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="EmpId" type="xs:string" />
                <xs:element name="EmpName" type="xs:string" />
                <xs:element name="Department" type="xs:string" />
                <xs:element name="Salary" type="xs:string" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
</xs:schema>
```

Output



9. Conclusion: -

From above assignment I have conclude that:_____

10. Review Questions

a. **Programming Assignment:**
   i. Write a program to design book catalog by using XML and XSLT to Sort and display title, author, price and year of the book.

ii. Write a program to design T&P Placement Record by using XML and XSLT to display Company_name, technology_work, avg_salary and year_of_Startup of the book.

b. Viva Questions:

i. Why Is Xml Such an Important Development?
ii. Who Is Responsible for Xml?
iii. Give Some Examples of Xml Dtds Or Schemas that You Have Worked With?
iv. What Is Xslt Data Model?
v. Does Xml Replace Html?
vi. How to Rename A Particular Element and Attribute from Xml Using Xsl?
vii. Compare Xslt And Xpath?
viii. How to Use Filtering in Xslt?
ix. How You Define Complex and Simple Type of Element?
x. Why Xml Schemas Are Estensible?
xi. How You Set Default and Fixed Values for Attributes?
xii. How We Can Say Xml Schemas Are Successors of Dtds?
xiii. How Do I Create My Own Document Type?