# Registration Form

A Skill Oriented Course Web Application report

Submitted in the partial fulfillment of the requirements for

the award of the degree of

Bachelor of Technology
In

Computer Science and Engineering

by

Student_Name : kota sagar
Roll_Number : 20761A0595

Under the guidance of

Dr. K Devi Priya
Associate Professor, Dept. of CSE

**Department of Computer Science and Engineering**
**Lakireddy Bali Reddy College of Engineering (Autonomous)**
**Accredited by NAAC & NBA (Under Tier - I)**
**Affiliated to JNTUK, Kakinada; ISO 9001:2015 Certified**
**2021-22**

# LAKIREDDY BALI REDDY COLLEGE OF ENGINEERING

(AUTONOMOUS)

Accredited by NAAC & NBA (Under Tier - I) ISO 9001:2015 Certified Institution
Approved by AICTE, New Delhi and Affiliated to JNTUK, Kakinada
L.B. REDDY NAGAR, MYLAVARAM, KRISHNA DIST., A.P.-521 230.
http://cse.lbrce.ac.in, cselbreddy@gmail.com, Phone: 08659-222933, Fax: 08659-222931

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

## Certificate

This is to certify that the Skill Oriented Course Web Application / Project entitled "**registration form**" is being submitted by **kota sagar** in partial fulfillment for the award of B. Tech in Computer Science & Engineering to the Jawaharlal Nehru Technological University Kakinada is a record of bonafide work carried out by him/her under our guidance.

The results embodied in this Skill Oriented Course Web Application / Project report have not been submitted to any other University or Institute for the award of any degree or diploma.

| | |
|---|---|
| **Projectguide** | **Head of the department** |
| Dr. K Devi Priya, | Dr.D.Veeraiah, |
| Associate Professor. | Professor. |

**External Examiner**

## ACKNOWLEDGEMENT

I would like to thank **Dr. K Devi Priya,** Associate Professor**,** CSE department for the encouraging and support in carrying out this Skill Oriented Course Web Application.

I also take the privilege to record my thanks to **Dr. D. Veeraiah,** Professor,Head of the Department of CSE whose encouragement, cooperation and valuable support crown my success.

I express my thanks to the support given by management in completing my Web Application / Project. I also express my sincere gratitude & deep sense of respect to the Principal, **Dr.K. Appa Rao** for making us available all the required assistance and his support and inspiration to carry out this Skill Oriented Course Web Application / Project in the Institute.

I am thankful to the teaching and non-teaching staff of CSE department for their direct as well as indirect help in my Skill Oriented Course Web Application / Project.

I am elated to avail my selves to this opportunity to express my deep sense of gratitude to my parents.

# Registration form

## Abstract

I have designed a simple registration from about the job registration form ,
->Mainly I concentrated on the frontend and also covered the all the concepts in html and css and javascript
->in this job registration form I have classified web page  into 2 types

1.primary details
2.secoundary details

->In this primary side I have mentioned all the basic details such as address ,contact info and bio data …
->on the other side the main details are required for the job like previous job details and education qualifications  and company requirements etc…
->this info it will give brief idea about my project

## CONTENTS

## LIST OF IMAGES

## *INTRODUCTION :*

HTML stands for Hyper Text Markup Language, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

## TECHNOLOGIES USED:

### 1.HTML : (HYPER TEXT MARKUP LANGUAGE)

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

### BASIC TAGS :

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document
- The <html> element is the root element of an HTML page
- The <head> element contains meta information about the HTML page
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading
- The <p> element defines a paragraph

**SOME BASIC CONCEPTS OF HTML:**

**1.HTML Links - Hyperlinks**

- HTML links are hyperlinks.
- You can click on a link and jump to another document.
- When you move the mouse over a link, the mouse arrow will turn into a little hand.
- A link does not have to be text. A link can be an image or any other HTML element

Syntax:

The HTML <a> tag defines a hyperlink. It has the following syntax:

```
<a href="url">link text</a>
```

The most important attribute of the <a> element is the href attribute, which indicates the link's destination.

The *link text* is the part that will be visible to the reader.

Clicking on the link text, will send the reader to the specified URL address.

By default, links will appear as follows in all browsers:

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

By default, the linked page will be displayed in the current browser window. To change this, you must specify another target for the link.

The target attribute specifies where to open the linked document.

The target attribute can have one of the following values:

- _self - Default. Opens the document in the same window/tab as it was clicked
- _blank - Opens the document in a new window or tab
- _parent - Opens the document in the parent frame
- _top - Opens the document in the full body of the window

**2.INSERTING IMAGES:**

Images can improve the design and the appearance of a web page.

- The HTML <img> tag is used to embed an image in a web page.

- Images are not technically inserted into a web page; images are linked to web pages. The <img> tag creates a holding space for the referenced image.

- The <img> tag is empty, it contains attributes only, and does not have a closing tag.

The <img> tag has two required attributes:

- src - Specifies the path to the image
- alt - Specifies an alternate text for the image

**Syntax:**

```
<img src="url" alt="alternatetext">
```

## 3.HTML TABLES:

- HTML tables allow web developers to arrange data into rows and columns.
- A table in HTML consists of table cells inside rows and column.
- Each table cell is defined by a <td> and a </td> tag.
- td stands for table data.
- Everything between <td> and </td> are the content of the table cell.
- Each table row starts with a <tr> and end with a </tr> tag.
- tr stands for table row.
- Sometimes you want your cells to be headers, in those cases use the <th> tag instead of the <td> tag

Syntax:

```
<table>
  <tr>
    <th>column 1</th>
    <th>column 2</th>
    <th>column 3</th>
  </tr>
  <tr>
    <td>data item 1</td>
    <td>data item 2</td>
    <td>data item 3</td>
  </tr>
</table>
```

**4.HTML LISTS:**

- HTML lists allow web developers to group a set of related items in lists.
- An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.
    - ➢ The list items will be marked with bullets (small black circles) by default.
- An ordered list starts with the <ol> tag. Each list item starts with the <li> tag.
    - ➢ The list items will be marked with numbers by default.

Syntax and example:

➢ Unordered list

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Output:

- Coffee
- Tea
- Milk

- Ordered list

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Output:

```
1. Coffee
2. Tea
3. Milk
```

## 5.HTML FORMS:

- An HTML form is used to collect user input. The user input is most often sent to a server for processing.

- The HTML <form> element is used to create an HTML form for user input.
- The <form> element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.

**Syntax:**

```
<form>
  .
  form elements
  .
</form>
```

Elements:

1.input: The HTML <input> element is the most used form element.

An <input> element can be displayed in many ways, depending on the type attribute.

- <input type="button">
- <input type="checkbox">
- <input type="email">
- <input type="password">
- <input type="submit">
- <input type="text">

2.label:

The <label> tag defines a label for many form elements.

The <label> element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focus on the input element.

The for attribute of the <label> tag should be equal to the id attribute of the <input> element to bind them together.

Syntax: <label for="name">labelname</label>

Example:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

Output:

```
First name:

Last name:

```

**2.CSS : (CASCADING STYLE SHEETS)**

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External stylesheets are stored in CSS files.

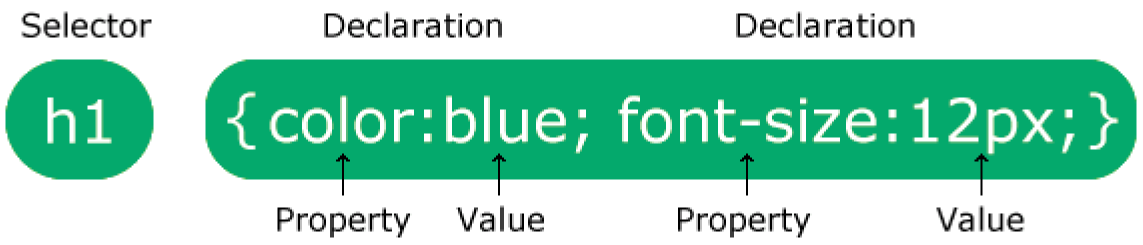
**WHY WE HAVE TO USE CSS:**

- HTML was NEVER intended to contain tags for formatting a web page!
- HTML was created to describe the content of a web page, like:
- <h1>This is a heading</h1> And <p>This is a paragraph.</p>
- When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.

- To solve this problem, the World Wide Web Consortium (W3C) created CSS.
- CSS removed the style formatting from the HTML page!

SYNTAX:

A CSS rule 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.
- Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

EXAMPLE:

```
p {
  color: red;
  text-align: center;
}
```

**CSS SELECTORS:**

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinator selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

*ELEMENT SELECTOR*:

```
p {
  text-align: center;
  color: red;
}
```

*ID SELECTOR*:

```
#para1 {
  text-align: center;
  color: red;
}
```

*CLASS SELECTOR*:

```
.center {
  text-align: center;
  color: red;
}
```

*UNIVERSAL SELECTOR*:

```
* {
  text-align: center;
  color: blue;
}
```

**ADDING CSS TO HTML DOCUMENT**

There are three ways of inserting a style sheet:

- External CSS
- Internal CSS
- Inline CSS

1.EXTERNAL CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

Example:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="mystyle.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

```
</body>
</html>
```

An external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

2.INTERNAL CSS:

An external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: linen;
}

h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
```

```
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

3.INLINE CSS:

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Example:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>
```

**CASCADING ORDER:**

What style will be used when there is more than one style specified for an HTML element?

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

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

3.**JS : (JAVA SCRIPT)**

- JavaScript is the world's most popular programming language.
- JavaScript is the programming language of the Web.
- JavaScript is easy to learn.
- JavaScript is one of the **3 languages** all web developers **must** learn:

  1. **HTML** to define the content of web pages
  2. **CSS** to specify the layout of web pages
  3. **JavaScript** to program the behavior of web pages

**HOW TO ADD JS FILE TO HTML:**

In HTML, JavaScript code is inserted between <script> and </script> tags.

Example:

```
<script>
document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
```

You can place any number of scripts in an HTML document.

Scripts can be placed in the <body>, or in the <head> section of an HTML page, or in both.

**FUNCTIONS AND EVENTS:**

A JavaScript function is a block of JavaScript code, that can be executed when "called" for.

For example, a function can be called when an **event** occurs, like when the user clicks a button.

**Example:**

```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph
changed.";
}
</script>
```

```
</head>
<body>

<h2>Demo JavaScript in Head</h2>

<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>

</body>
</html>
```

**JAVA SCRIPT OUTPUT:**

JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML.
- Writing into the HTML output using document.write().
- Writing into an alert box, using window.alert().
- Writing into the browser console, using console.log().

To access an HTML element, JavaScript can use the document.getElementById(id) method.

The id attribute defines the HTML element. The innerHTML property defines the HTML content.

Example:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My First Paragraph</p>
```

```
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 5 + 6;
</script>

</body>
</html>
```

Output:

**My First Web Page**

My First Paragraph.

11

---

**JAVA SCRIPT SYNTAXES**:

JavaScript syntax is the set of rules, how JavaScript programs are constructed:

```
// How to create variables:
var x;
let y;

// How to use variables:
x = 5;
y = 6;
let z = x + y;
```

**VARIABLES AND LITERALS:**

The JavaScript syntax defines two types of values:

- Fixed values
- Variable values

Fixed values are called **Literals**.

- **Numbers** are written with or without decimals:

  10.50

  1001

- **Strings** are text, written within double or single quotes:
  - "John Doe"

    'John Doe'

Variable values are called **Variables**.

- In a programming language, **variables** are used to **store** data values.
- JavaScript uses the keywords var, let and const to **declare** variables.
- An **equal sign** is used to **assign values** to variables.

- In this example, x is defined as a variable. Then, x is assigned (given) the value 6:

```
let x;
x=6;
```

**JAVA SCRIPT FUNCTIONS:**

A JavaScript function is defined with the function keyword, followed by a **name**, followed by parentheses **()**.

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).

The parentheses may include parameter names separated by commas: **(*parameter1, parameter2, ...*)**

The code to be executed, by the function, is placed inside curly brackets: **{}**

**SYNTAX:**

```
function name(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

- Function **parameters** are listed inside the parentheses () in the function definition.
- Function **arguments** are the **values** received by the function when it is invoked.
- Inside the function, the arguments (the parameters) behave as local variables.

FUNCTION INVOCATION:

The code inside the function will execute when "something" **invokes** (calls) the function:

- When an event occurs (when a user clicks a button)
- When it is invoked (called) from JavaScript code
- Automatically (self invoked)

FUNCTION RETURN:

When JavaScript reaches a return statement, the function will stop executing.

If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.

Functions often compute a **return value**. The return value is "returned" back to the "caller".

Example:

```
let x = myFunction(4, 3);

 // Function is called, return value will end up in x

function myFunction(a, b) {
  return a * b;                    // Function returns the product of a and b
}
```

OUTPUT:

The result in x will be:

```
12
```

**ADVANTAGES:**

You can reuse code: Define the code once, and use it many times.

You can use the same code many times with different arguments, to produce different results.

**REGULAR EXPRESSIONS (REGEX);**

A regular expression is a sequence of characters that forms a **search pattern**.

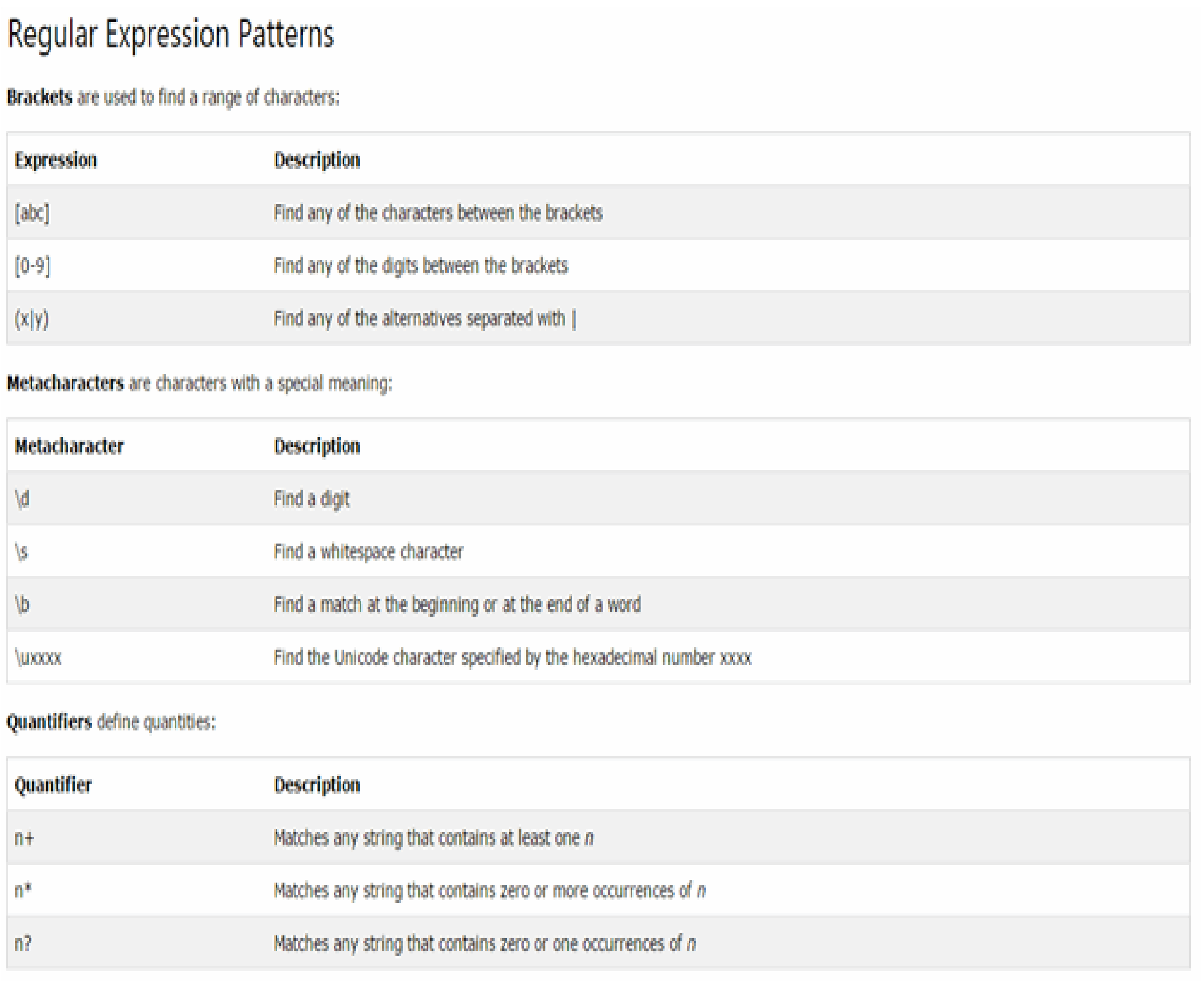
When you search for data in a text, you can use this search pattern to describe what you are searching for.

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*;



## Regular Expression Patterns

**Brackets** are used to find a range of characters:

| Expression | Description |
|---|---|
| [abc] | Find any of the characters between the brackets |
| [0-9] | Find any of the digits between the brackets |
| (x\|y) | Find any of the alternatives separated with \| |

**Metacharacters** are characters with a special meaning:

| Metacharacter | Description |
|---|---|
| \d | Find a digit |
| \s | Find a whitespace character |
| \b | Find a match at the beginning or at the end of a word |
| \uxxxx | Find the Unicode character specified by the hexadecimal number xxxx |

**Quantifiers** define quantities:

| Quantifier | Description |
|---|---|
| n+ | Matches any string that contains at least one *n* |
| n* | Matches any string that contains zero or more occurrences of *n* |
| n? | Matches any string that contains zero or one occurrences of *n* |