Web Technologies and Programming

Lecture 25

Validating User Input

Summary of Previous Lecture

- **Passing Form Data**
  - **action**
  - **method (POST or GET)**
    - **When to Use GET?**
    - **When to Use POST?**
    - **Compare GET vs. POST**
- **Super Global Variables**

Summary of Previous Lecture

- **Passing data with forms**
  - Passing Text Field Data
  - Passing Hidden Field Data
  - Getting Value From Checkbox
  - Getting Value From Radio Button
  - Getting Value From Select List
- **Using session Variables**

Today's Lecture Outline

- **Regular expressions in PHP**
- **Validating user input at server**
- **String functions**

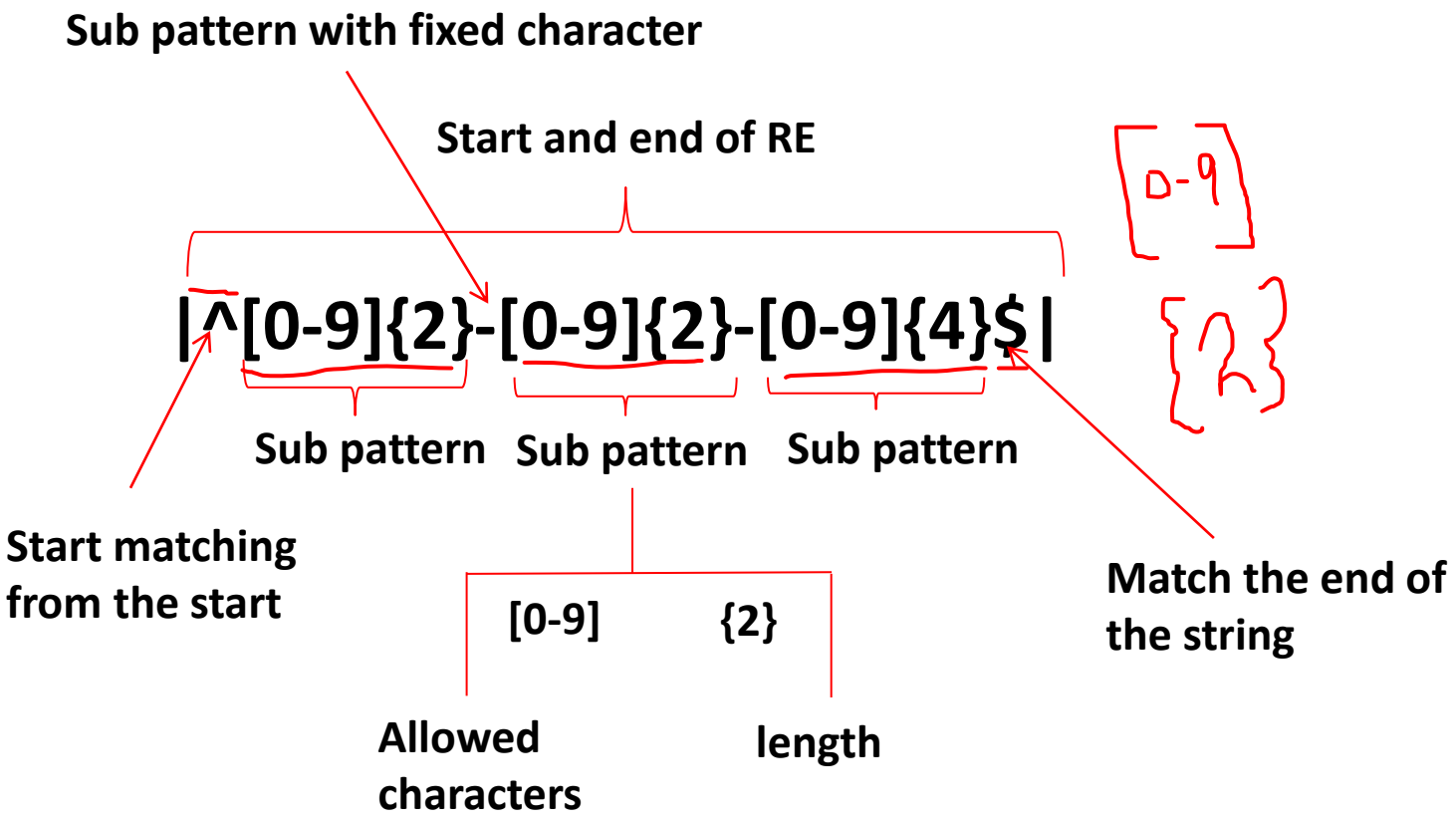
1. Regular expressions in PHP

- **Regular expressions are sequence or pattern of characters itself. They provide the foundation for pattern-matching functionality**
- **A regular expression is a concise notation to describe patterns in strings**
- **Regular expressions provide the foundation for describing or matching data according to defined syntax rules**
  - **Example: |^[0-9]{2}-[0-9]{2}-[0-9]{4}$|**

1. Regular expressions in PHP

- **Using regular expression you can search a particular string inside a another string, you can replace one string by another string and you can split a string into many chunks.**
- **PHP offers functions specific to two sets of regular expression functions, each corresponding to a certain type of regular expression. You can use any of them based on your comfort.**
- POSIX Regular Expressions
- PERL Style Regular Expressions

1. Regular expressions in PHP…



1. Regular expressions in PHP…

- **Brackets**
- **Brackets ([]) have a special meaning when used in the context of regular expressions. They are used to find a range of characters.**
- **[0-9]** It matches any decimal digit from 0 through 9.
- **Brackets: [a-Z], [A-Z], [a-z], [0-9]**

1. Regular expressions in PHP…

- **Quantifiers**
- **The frequency or position of bracketed character sequences and single characters can be denoted by a special character.**
- **Each special character having a specific connotation. The +, *, ?, {int. range}, and $ flags all follow a character sequence**

1. Regular expressions in PHP…

- **p+ :** It matches any string containing at least one p.
- **p* :** It matches any string containing zero or more p's.
- **p? :** It matches any string containing zero or more p's. This is just an alternative way to use p*.
- **p{N}:** It matches any string containing a sequence of **N** p's
- **p{2,3}:** It matches any string containing a sequence of two or three p's.
- **p{2, }:** It matches any string containing a sequence of at least two p's.
- **p$:** It matches any string with p at the end of it.
- **^p :** It matches any string with p at the beginning of it.

1. Regular expressions in PHP…

- **Start and end of the RE:**
  - **optional, ||**
- **Sub-patterns:**
  - **range of allowed characters**
    - **[0-9]**
  - **Allowed length**
    - **{2}**
- **Sub-patterns with fixed character**

1. Regular expressions in PHP…

- **Matching from the start: ^:**
  - 1212-12-2014

Pattern exists if do not match from start

- **Matching till end: $:**
  - 12-12-2014123

Pattern exists if do not match till end

- **For exact match we should use both ^ and $**

1.1 Notations for RE

- ^:
  - Match strings that start with the given pattern
- $:
  - Match strings that end with the given pattern
- -:
  - Range of characters
- [ ]:
  - Makes a class of characters
- [^]:
  - Negates the class of character

1.1 Notation for RE…

- **Quantifiers:**
  - **{n}:**
    - **matches a character, class or sub-pattern for n times**
  - **{n, m}:**
    - **matches a character, class or sub-pattern for minimum n times and maximum m times**

1.1 Notation for RE…

- **?:**
  - **matches the character, class or sub-pattern 0 or 1 time**
    - **equal to {0,1}**
- **+:**
  - **matches the character, class or sub-pattern 1 or more times**
    - **equals to {1, }**
- ***:**
  - **matches the character, class or sub-pattern 0 or any number of time**
    - **equals {0, }**

1.1 Notation for RE…

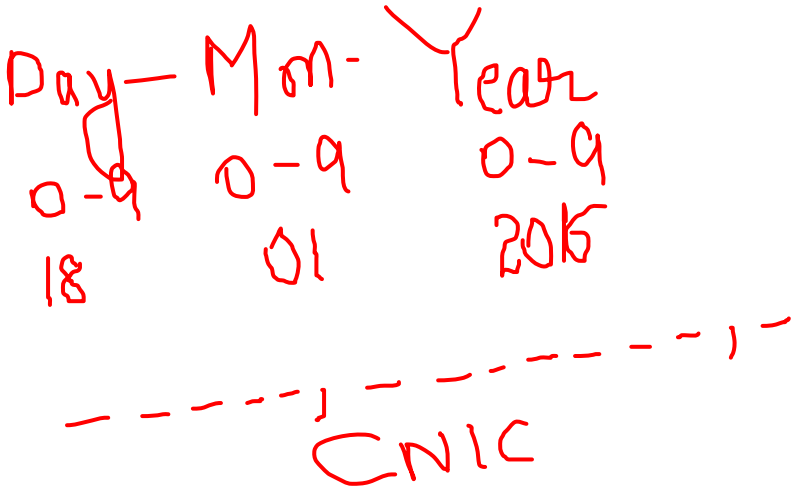
Predefined character ranges:

- **\d:**
  - **Exactly as [0-9]**
- **\D:**
  - **Exactly as [^0-9]**
- **\w:**
  - **Exactly as [a-zA-Z0-9]**

1.1 Notation for RE…

RE examples:

- **Validating date:**
  - **|^\d{2}-\d{2}-\d{4}$|**
- **Validating CNIC:**
  - **|^\d{5}-\d{7}-\d{1}$|**
- **Validating Email:**
  - **|^[a-zA-Z0-9_.]+@[a-z]{3,5}.[a-z]{2,3}$|**



1.1 Notation for RE…

- **Validating name:**
  - **|^[a-zA-Z]{5,25}$|**
- **Validating Password:**
  - **must contain ‘@’**
    - **|@|**

2. Validating User's Input

- **preg_match():**
  - searches a string for a specific pattern
  - returns TRUE if it exists
  - retruns FALSE otherwise
  - preg_match("pattern",$string);

2. Validating User's Input

- **preg_match_all():**
- **The preg_match_all() function matches all occurrences of pattern in string.**

- **preg_grep():**
- **The preg_grep() function searches all elements of input_array, returning all elements matching the regexp pattern.**

2. Validating User's Input…

```
<html>
<head>
    <title> User Validation </title>
</head>

<body>
    <h1> User Registration </h1>
    <form action="UserValidation.php" method="post">
        <table>
            <tr>
                <td> Name </td>
                <td> <input type="text" name="name"> </td>
            </tr>
            <tr>
                <td> E-mail </td>
                <td> <input type="text" name="email"> </td>
            </tr>
            <tr>
                <td> CNIC </td>
                <td> <input type="text" name="cnic"> </td>
            </tr>
            <tr>
                <td> DOB </td>
                <td> <input type="text" name="dob"> </td>
            </tr>
            <tr>
                <td> <input type="submit"> </td>
            </tr>
    </form>
</body>
</html>
```

2. Validating User's Input…

2. Validating User's Input

```
<html>
<head>
    <title> User Validation </title>
</head>

<body>
    <h1> Validation Status </h1>
    <?php
        $name = $_POST['name'];
        $email = $_POST['email'];
        $cnic = $_POST['cnic'];
        $dob = $_POST['dob'];
```

Receiving Values

2. Validating User's Input

```php
if(!preg_match("|^[a-zA-Z ]{3,25}$|", $name))
    echo "Invalid Name Input";
else
    echo "Your Name is ".$name;

echo "<br>";

if(!preg_match("|^[a-zA-Z0-9_.]+@[a-z]{3,5}.[a-z]{3}$|", $email))
    echo "Invalid Email Address";
else
    echo "Your Email Address is ".$email;

echo "<br>";
```

Validating Name

Validating Email Address

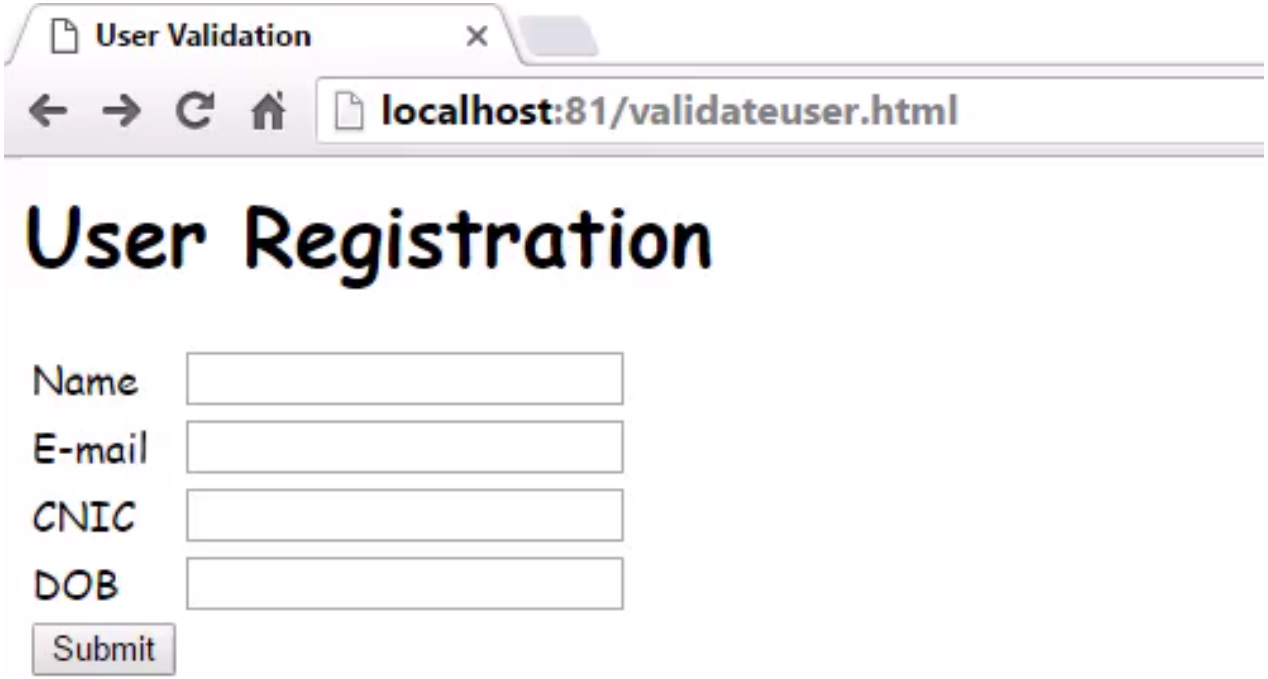
2. Validating User's Input

```php
    if(!preg_match("|^\d{5}-\d{7}-\d{1}$|", $cnic))
        echo "Invalid CNIC";
    else
        echo "Your CNIC is ".$cnic;

    echo "<br>";

    if(!preg_match("|^\d{2}-\d{2}-\d{4}$|", $dob))
        echo "Invalid Date of Birth";
    else
        echo "Your Date of Birth is ".$dob;
?>
```

Validating CNIC

Validating DoB

2. Validating User's Input



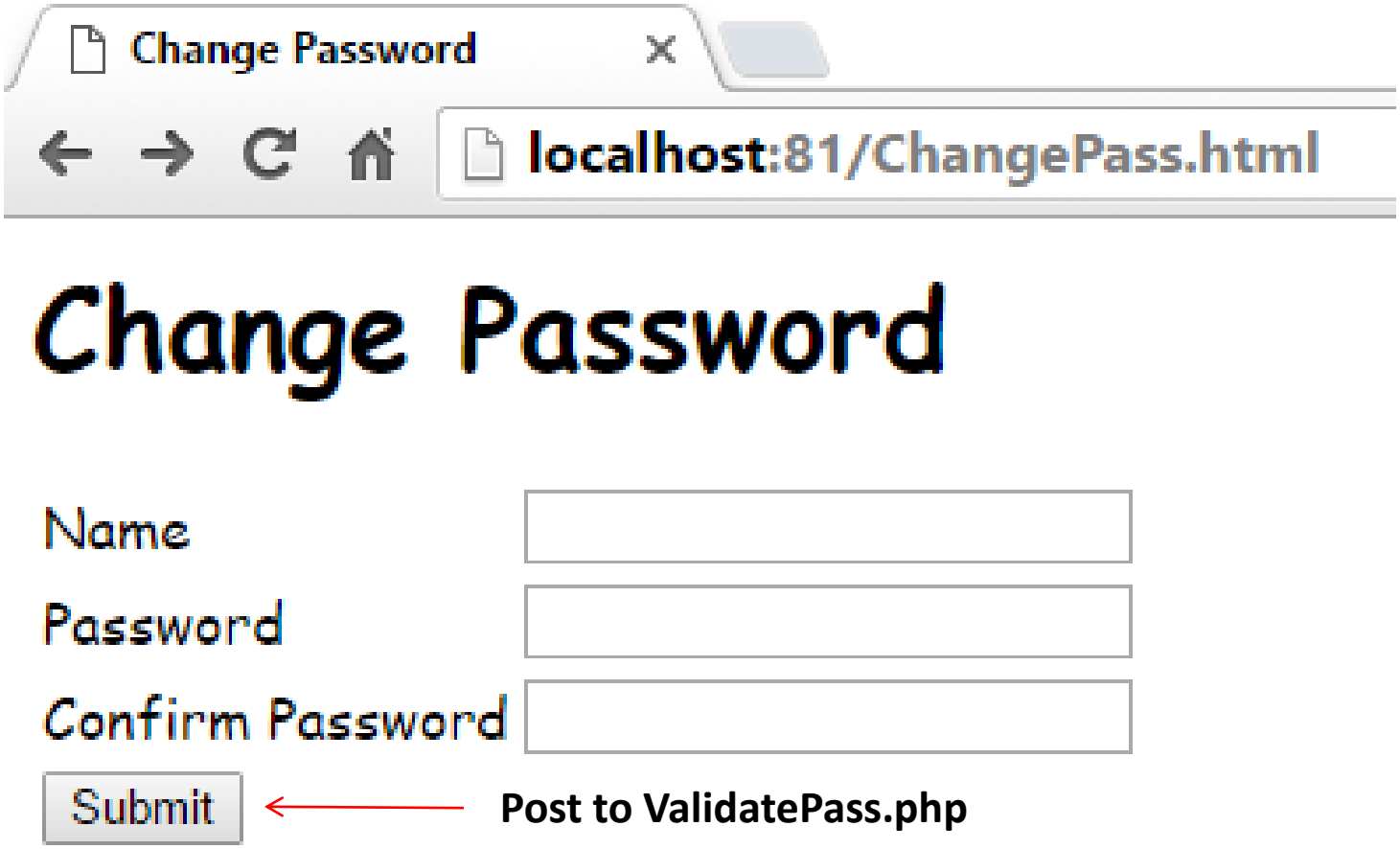
3. String Functions in PHP

- **strlen():**
  - **Returns the length of the string**
  - **strlen($string);**
- **strcmp():**
  - **Compares two strings**
  - **Returns 0 if strings are equal**
  - **1 if first string is greater than second string**
  - **-1 if second string is greater than first string**
  - **strcmp($string1, $string2);**
- **strcasecmp():**
  - **Compares two strings in case insensitive manner**
  - **strcasecmp($string1, $string2);**

3. String Functions in PHP…

```
<h1> Change Password </h1>
<form action="ValidatePass.php" method="post">
    <table>
        <tr>
            <td> Name </td>
            <td> <input type="text" name="name"> </td>
        </tr>
        <tr>
            <td> Password </td>
            <td> <input type="text" name="pass"> </td>
        </tr>
        <tr>
            <td> Confirm Password </td>
            <td> <input type="text" name="cpass"> </td>
        </tr>
        <tr>
            <td> <input type="submit"> </td>
        </tr>
</form>
```

3. String Functions in PHP…



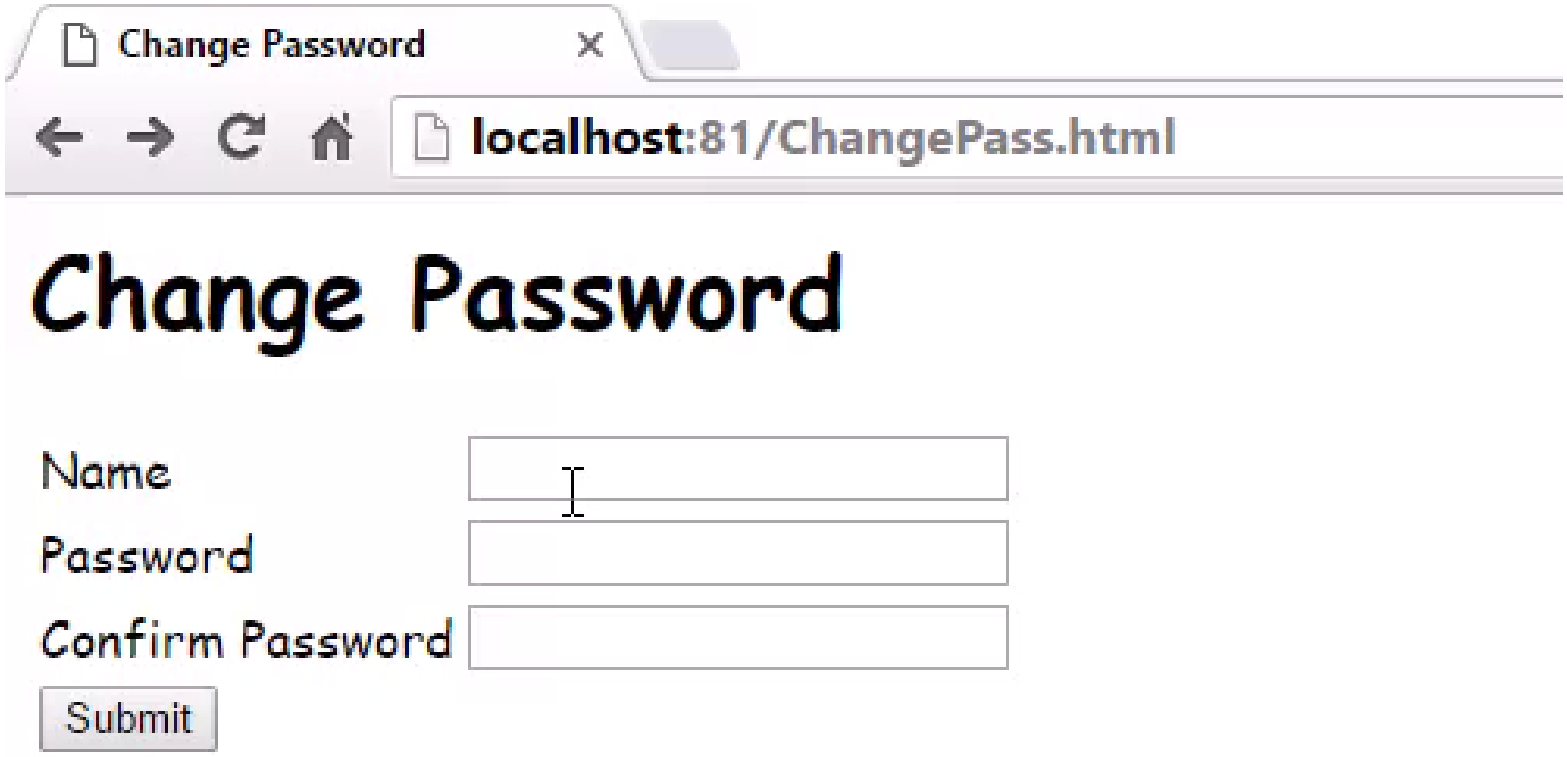
3. String Functions in PHP…

```
<h1> Password Validation </h1>
<?php
    $name = $_POST['name'];
    $pass = $_POST['pass'];
    $cpass = $_POST['cpass'];

    if(strlen($pass) < 8)
        echo $name."! Your Password Is Too Short!";
    else
        echo "Your Password Length Is ".strlen($pass);
    ?>
```

Getting Variables

Using strlen()

3. String Functions in PHP…

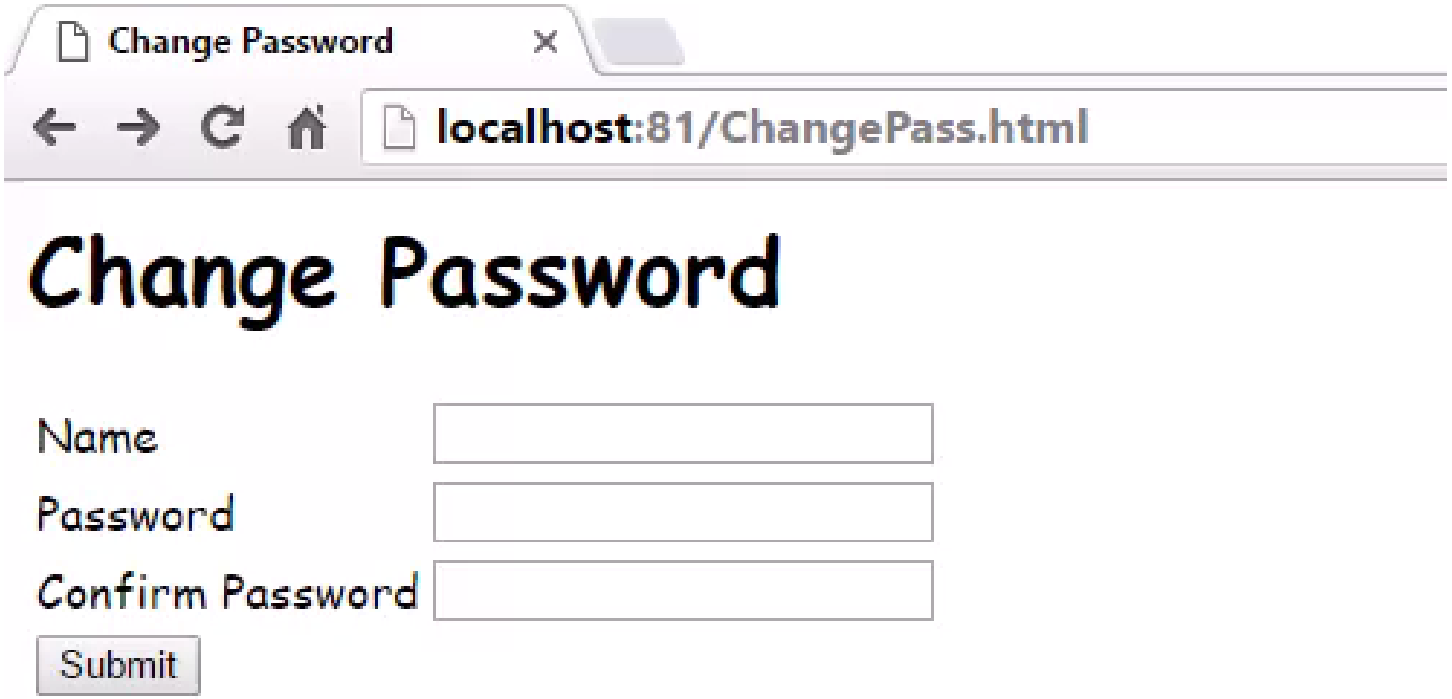


3. String Functions in PHP…

```
<h1> Password Validation </h1>
<?php
    $name = $_POST['name'];
    $pass = $_POST['pass'];
    $cpass = $_POST['cpass'];

    if(strlen($pass) < 8)
        echo $name."! Your Password Is Too Short!";
    else
        echo "Your Password Length Is ".strlen($pass);

    echo "<br>";

    if(strcmp($pass, $cpass)<>0)
        echo "Your Passwords Doesn't Match! Please Try Again!";
    else
        echo "Passwords Matched!";
?>
```

Compares Pass and cPass

3. String Functions in PHP…

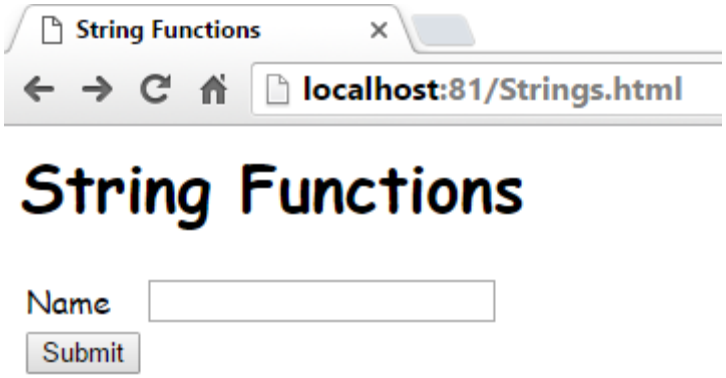


3. String Functions in PHP…

- **strtolower():**
  - **Convert a string in lower case**
  - **strtolower($string);**
- **strtoupper():**
  - **Convert a string in upper case**
  - **strtoupper($string);**
- **ucfirst():**
  - **Convert the first character of a string to upper case**
  - **ucfirst($string);**
- **ucwords():**
  - **Convert the first character of each word in a string to upper case**
  - **ucfirst($string);**

3. String functions in PHP…

```
<h1> String Functions </h1>
<form action="StringFuncs.php" method="post">
    <table>
        <tr>
            <td> Name </td>
            <td> <input type="text" name="name"> </td>
        </tr>
        <tr>
            <td> <input type="submit"> </td>
        </tr>
</form>
```



3. String functions in PHP…

```
<h1> String Functions </h1>
<?php
    $name = $_POST['name'];

    echo strtolower($name);
    echo "<br>";
    echo strtoupper($name);
    echo "<br>";
    echo ucfirst($name);
    echo "<br>";
    echo ucwords($name);
    echo "<br>";
?>
```

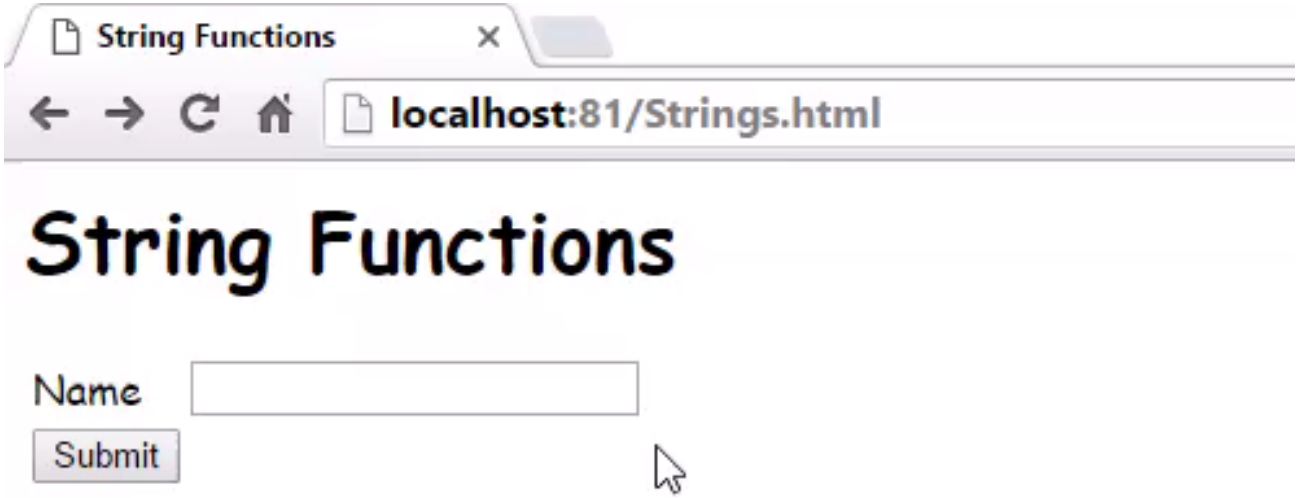
Converts to Lowercase

Converts to Uppercase

Using ucfirst()

Using ucwords()

3. String functions in PHP…



3. String Functions in PHP…

- **strpos():**
  - finds the position of the first case-sensitive occurrence of a substring in a string
  - strpos($string,sub-string);
- **strrpos():**
  - finds the position of the last case-sensitive occurrence of a substring in a string
  - strrpos($string,sub-string);
- **substr_count():**
  - returns the number of times one string occurs within another
  - substr_count($string,sub-string);

3. String functions in PHP…

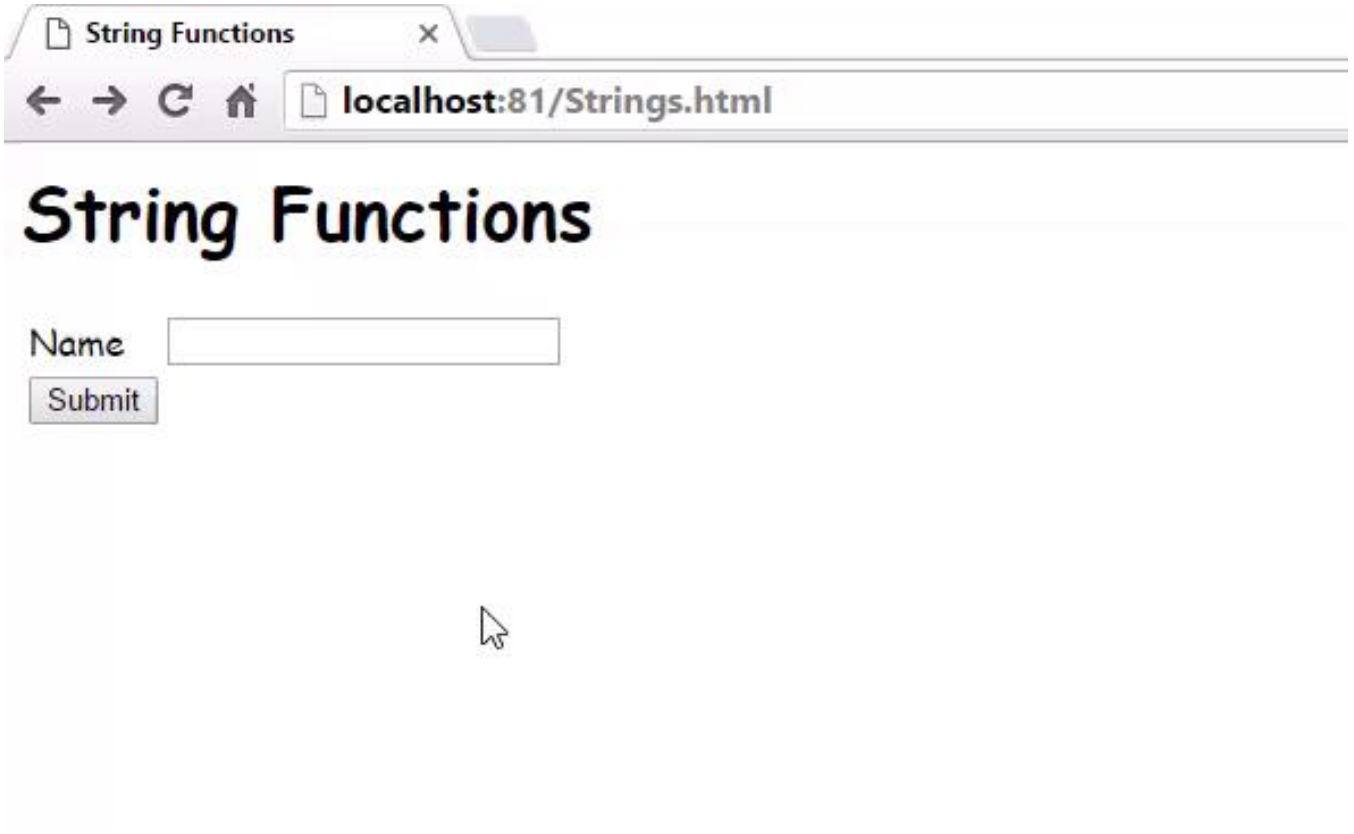
First Occurrence of ‘E’

```
echo "First Occurrence of E: ".strpos($name, 'e');
echo "<br>";
echo "Last Occurrence of E: ".strrpos($name, 'e');
echo "<br>";
echo "Total Occurrences of E: ".substr_count($name, 'e');
echo "<br>";
```

Last Occurrence of ‘E’

All Occurrences of ‘E’

3. String functions in PHP…



Summary of PHP Lectures

- Setting the environment
- PHP overview
  - What is a PHP File
  - Open-source
  - Platform independent
  - What Can PHP Do?
  - Why PHP?
  - Basic PHP Syntax
  - Writing and Executing PHP Code
- PHP constants
  - Constants are Global
- PHP variables
  - Local
  - Global
  - Static
  - Type Determination
- PHP Strings
- PHP is a Loosely Typed Language

Summary of PHP Lectures

- **Operators in PHP**
  - **Arithmetic Operators: +, - ,*, /, %, **
  - **Assignment Operators: =**
  - **String Operators: . , .=**
  - **Increment/decrement Operators: ++ , --**
  - **Logical Operators: AND, OR, NOT, XOR, &&, ||, !**
  - **Comparison Operators: >, <, <=, >=**
  - **Equality Operators: ==, !=, ===**

- **Conditional statements**
- **if statement - executes some code if one condition is true**
- **if...else statement - executes some code if a condition is true and another code if that condition is false**
- **if...elseif....else statement - executes different codes for more than two conditions**
- **switch statement - selects one of many blocks of code to be executed**

Summary of PHP Lectures

- **Looping statements**
  - **For Loop**
  - **While Loop**
  - **Do-While Loop**
  - **ForEach Loop**
- **Arrays in PHP**
  - **Associative arrays**
  - **Sorting arrays**

Summary of PHP Lectures

- **Passing Form Data**
  - **action**
  - **method (POST or GET)**
    - **When to Use GET?**
    - **When to Use POST?**
    - **Compare GET vs. POST**
- **Super Global Variables**

Summary of PHP Lectures

- **Passing data with forms**
  - Passing Text Field Data
  - Passing Hidden Field Data
  - Getting Value From Checkbox
  - Getting Value From Radio Button
  - Getting Value From Select List
- **Using session Variables**

Summary of Today's Lecture

- **Writing regular expression in PHP**
  - **Brackets []**
  - **Quantifiers +, *, ?, {int. range}, and $**
  - **Sub-patterns**
  - **Predefined character ranges\d:\D:\w:**

- **Validating User's Input**
  - **Validating name:**
  - **Validating Password:**
  - **Validating date:**
  - **Validating CNIC:**
  - **Validating Email:**
  - **Validating user's input**

Summary of Today's Lecture

- **Defined functions.**
  - **preg_match():**
  - **preg_match_all():**
  - **preg_grep():**

- **String Functions in PHP**
  - **strlen():**
  - **strcmp():**
  - **strcasecmp():**
  - **strtolower():**
  - **strtoupper():**
  - **ucfirst():**
  - **ucwords():**
  - **strpos():**
  - **strrpos():**
  - **substr_count():**

THANK YOU