PDO – PHP Database Extension

https://phpgurukul.com

Index

Chapter-1

PDO – PHP database extension

PDO (PHP Data Objects) is a PHP extension through which we can access and work with databases. Though PDO is similar in many aspects to mySQLi, it is better to work with for the following reasons:

- It is better protected against hackers.
- It is consistent across databases, so it can work with MySQL as well as other types of databases (SQLite, Oracle, PostgreSQL, etc.)
- It is object oriented at its core.

In this PDO tutorial you will find recipes for 4 basic functions that we perform with the database: insertion, selection, update, and deletion. The recipes are intended to work with MySQL, but we can easily switch it with another database.

How to connect with the database through PDO?
It is considered good practice to wrap the database connection within a try-catch block so that, if anything goes wrong, an exception will be thrown. We can customize the error message but, in order to keep things simple, we'll settle with the error message that PDO provides.

In order to connect to the database, we'll need the database name, username, and password.

```
1. // DB credentials.
2. define('DB_HOST','localhost');
3. define('DB_USER','your user name');
4. define('DB_PASS','your user password');
5. define('DB_NAME','your database name');
6. // Establish database connection.
7. try
8. {
9. $dbh = new PDO("mysql:host=".DB_HOST.";dbname=".DB_NAME,DB_USER, DB_PASS,ar
   ray(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES 'utf8'"));
10. }
11. catch (PDOException $e)
12. {
13. exit("Error: " . $e->getMessage());
14. }
```

How to close the database connection?

PHP automatically closes the database connection but, if the need arises, we can deliberately close the connection with the following line of code:

```
1. $dbh = null;
```

Chapter 2

How to use PDO to insert data into the database?

The SQL code for the users table:

```
1. CREATE TABLE IF NOT EXISTS users (id int(11) NOT NULL AUTO_INCREMENT,
2. name varchar(60) DEFAULT NULL,
3. phone varchar(12) DEFAULT NULL,
4. city varchar(60) DEFAULT NULL,
5. date_added date DEFAULT NULL,
6. PRIMARY KEY (id)
7. )
```

1.Write a regular SQL query but, instead of values, put named placeholders. For example:

```
1. $sql ="INSERT INTO `users`(`name`, `phone`, `city`, `date_added`)
2. VALUES(:name,:phone,:city,:date)";
```

The use of placeholders is known as prepared statements. We use prepared statements as templates that we can fill later on with actual values.

2.Prepare the query:

```
1. $query = $dbh -> prepare($sql);
```

3.Bind the placeholders to the variables:

```
1. $query->bindParam(':name',$name);
```

You can add a third parameter which filters the data before it reaches the database:

```
1. $query->bindParam(':name',$name,PDO::PARAM_STR);
2. $query->bindParam(':phone',$phone,PDO::PARAM_INT);
3. $query->bindParam(':city',$city,PDO::PARAM_STR);
4. $query->bindParam(':date',$date,PDO::PARAM_STR);
```

- PDO::PARAM_STR is used for strings.
- PDO::PARAM_INT is used for integers.
- PDO::PARAM_BOOL allows only boolean (true/false) values.
- PDO::PARAM_NULL allows only NULL datatype.

4. Assign the values to the variables.

```
1. $name = "Anuj kumar";
```

```
$phone = "1234567890";
$city = "New Delhi";
$date = date('Y-m-d');
```

5. Execute the query:

```
$query -> execute();
```

6. Check that the insertion really worked:

```
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId>0)
{
echo "Data inserted";
}
else
{
echo "Data not inserted";
}
```

If the last inserted id is greater than zero, the insertion worked.

All code together now:

```
$sql = "INSERT INTO `users`(`name`, `phone`, `city`, `date_added`)
VALUES
(:name,:phone,:city,:date)";
$query = $dbh -> prepare($sql);
$query->bindParam(':name',$name,PDO::PARAM_STR);
$query->bindParam(':phone',$phone,PDO::PARAM_INT);
$query->bindParam(':city',$city,PDO::PARAM_STR);
$query->bindParam(':date',$date);
// Insert the first row
$name = "Anuj";
$phone = "1234567890";
$city = "New Delhi";
$date = date('Y-m-d');
$query -> execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId>0)
{
echo "Data inserted";
 }
else {
echo "Data not inserted"; }

// Insert the second row
$name = "John Deo";
$phone = "9874563210";
$city = "New York";
$date = date('Y-m-d');
$query -> execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId>0){echo "Data inserted";}
else {echo "Data not inserted";}
```

The same task is better performed within a loop:

```
$sql = "INSERT INTO `users`(`name`, `phone`, `city`, `date_added`)
VALUES
(:name,:phone,:city,:date)";
$query = $dbh -> prepare($sql);
$query->bindParam(':name',$name,PDO::PARAM_STR);
$query->bindParam(':phone',$phone,PDO::PARAM_INT);
$query->bindParam(':city',$city,PDO::PARAM_STR);
$query->bindParam(':date',$date,PDO::PARAM_STR);
// Provide the data to the loop within an array
$date = date('Y-m-d');
$userData = array(
array("Anuj kumar","1234567890","New Delhi",$date),
array("John Doe","9876543210","New York",$date)
);

// Perform the query within a loop
foreach($userData as $key => $value)
{
$name = $value[0];
$phone = $value[1];
$city = $value[2];
$date = $value[3];
$query -> execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId>0){
echo "Data inserted";
} else {
echo "Data not inserted";}
}
```

Chapter 3

How to use PDO to read data from the database?

Reading data from the database is not so different than inserting data, with steps 1 to 5 being almost identical while the sixth step is different.

1. Write the regular select statement and again, instead of values, put named placeholders. For example:

```
$sql = "SELECT * FROM users";
```

2. Prepare the query:

```
$query = $dbh -> prepare($sql);
```

3.Execute the query:

```
$query -> execute();
```

4. Assign the data which you pulled from the database (in the preceding step) to a variable.

```
$results = $query -> fetchAll(PDO::FETCH_OBJ);
```

Here I used the parameter PDO::FETCH_OBJ that returns the fetched data as an object. If you'd like to fetch the data in the form of an array, use: PDO::FETCH_ASSOC.
5. Make sure that you were able to retrieve the data from the database, by counting the number of records.

```
if($query -> rowCount() > 0){}
```

6. In case that the query returned at least one record, we can echo the records within a foreach loop:

```
$sql = "SELECT * FROM users WHERE city = :city";
$query = $dbh -> prepare($sql);
$query -> bindParam(':city', $city, PDO::PARAM_STR);
$city = "New York";
$query -> execute();
$results = $query -> fetchAll(PDO::FETCH_OBJ);
if($query -> rowCount() > 0)  {
foreach($results as $result)
{
echo $result -> name . ", ";
echo $result -> city . ", ";
echo $result -> date_added;
}
}
```

Chapter 4

How to use PDO to update the database?

1.Write the regular update statement and again, instead of values, assign the named placeholders. For example:

```
$sql = "UPDATE `users` SET `city`= :city, `phone` = :tel
WHERE `id` = :id";
```

2. Prepare the query:

```
$query = $dbh->prepare($sql);
```

3. Bind the parameters:

```
$query -> bindParam(':city', $city, PDO::PARAM_STR);
$query -> bindParam(':tel' , $tel , PDO::PARAM_INT);
$query -> bindParam(':id' , $id , PDO::PARAM_INT);
```

4. Define the bound values:

```
$tel = '06901234567';
$city = 'New Delhi';
$id = 1;
```

5. Execute the query:

```
$query -> execute();
```

6. Check that the query has been performed and that the database has been successfully updated.

```
if($query -> rowCount() > 0)
{
$count = $query -> rowCount();
echo $count . "Record updated successfully.";
}
else
{
echo "Record not updated.";
}
```

All together now:

```
$sql = "UPDATE users
SET `city`= :city, `phone` = :tel WHERE `id` = :id";
$query = $dbh->prepare($sql);
$query -> bindParam(':city', $city, PDO::PARAM_STR);
$query -> bindParam(':tel' , $tel , PDO::PARAM_INT);
$query -> bindParam(':id' , $id , PDO::PARAM_INT);
$tel = '02012345678';
$city = 'London';
$id = 1;
$query -> execute();
if($query -> rowCount() > 0)
{
$count = $query -> rowCount();
echo $count . "Record updated successfully.";
}
else
{
echo "Record not updated.";
}
```

Chapter 5

How to delete records?

1. Write the delete statement :

```
1. $sql = "DELETE FROM `users` WHERE `id`=:id";
```

2. Prepare the query :

```
1. $query = $dbh -> prepare($sql);
```

3. Bind the parameters :

```
1. $query -> bindParam(':id', $id, PDO::PARAM_INT);
```

4. Define the bound values :

```
1. $id = 1;
```

5. Execute the query :

```
1. $query -> execute();
```

6. Check that the query has been performed and that the records have been successfully deleted from the database :

```
1. if($query -> rowCount() > 0)
2. {
3. $count = $query -> rowCount();
4. echo $count . " Data deleted.";
5. }
6. else
7. {
8. echo "No affected rows.";
9. }
```

All code together now:

```
$sql = "DELETE FROM `users` WHERE `id`=:id";
$query = $dbh -> prepare($sql);
$query -> bindParam(':id', $id, PDO::PARAM_INT);
$id = 1;
$query -> execute();
if($query -> rowCount() > 0)
{
$count = $query -> rowCount();
echo $count . " Data Deleted.";
}
else
{
echo "No affected rows.";
}
```

Chapter 6

PHP CRUD Operation using PDO Extension

CRUD Stands for create, read, update and delete record in the database.

File Structure for CRUD Operation

- **dbconfig.php :** Used for database connection
- **tblusers.sql :** Contain the structure of the database table
- **insert.php :** Used for add a record in the database
- **index.php :** Used for read the record from database .
- **update.php :** Used for updating a record.

Step 1– Create a database

Open browser type http://localhost/phpmyadmin, create a database named 'phpcrudpdo' . After creating database run the sql script.

```
-- Table structure for table `tblusers`
--
CREATE TABLE IF NOT EXISTS `tblusers` (
  `id` int(11) NOT NULL,
  `FirstName` varchar(150) NOT NULL,
  `LastName` varchar(150) NOT NULL,
  `EmailId` varchar(120) NOT NULL,
  `ContactNumber` char(11) NOT NULL,
  `Address` varchar(255) NOT NULL,
  `PostingDate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Indexes for table `tblusers`
--
ALTER TABLE `tblusers`
  ADD PRIMARY KEY (`id`);
-- AUTO_INCREMENT for dumped tables
-- AUTO_INCREMENT for table `tblusers`
--
ALTER TABLE `tblusers`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

Step 2– Create a database connection file (dbconfig.php)

create a dbconfig.php file and place the code given below :

```php
<?php
// DB credentials.
define('DB_HOST','localhost');
define('DB_USER','root');
define('DB_PASS','');
define('DB_NAME','phpcrudpdo');
// Establish database connection.
try
{
$dbh = new PDO("mysql:host=".DB_HOST.";dbname=".DB_NAME,DB_USER, DB_PASS);

}
catch (PDOException $e)
{
exit("Error: " . $e->getMessage());
}
?>
```

Step 3 : Insert a record in database

Create a insert.php file for insert record in the database . This page include a HTML form with input field where we can fill the data.

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>PHP CURD Operation using PDO Extension  </title>
        <meta name="viewport" content="width=device-width, initial-scale=1">
 <link href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.min.css" rel="stylesheet">
    <script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
    <script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.0/js/bootstrap.min.js"></script>
</head>
<body>

<div class="container">

<div class="row">
<div class="col-md-12">
<h3>Insert Record | PHP CRUD Operations using PDO Extension</h3>
<hr />
</div>
</div>


<form name="insertrecord" method="post">
```

```
<div class="row">
<div class="col-md-4"><b>First Name</b>
<input type="text" name="firstname" class="form-control" required>
</div>
<div class="col-md-4"><b>Last Name</b>
<input type="text" name="lastname" class="form-control" required>
</div>
</div>

<div class="row">
<div class="col-md-4"><b>Email id</b>
<input type="email" name="emailid" class="form-control" required>
</div>
<div class="col-md-4"><b>Contactno</b>
<input type="text" name="contactno" class="form-
control" maxlength="10" required>
</div>
</div>

<div class="row">
<div class="col-md-8"><b>Address</b>
<textarea class="form-control" name="address" required>
</div>
</div>
<div class="row" style="margin-top:1%">
<div class="col-md-8">
<input type="submit" name="insert" value="Submit">
</div>
</div>
</form>
</div>
</div>
</body>
</html>
```

Code for insert a record in database. Once the user filled all the data and click on the submit button then data will be saved in the database using the below code:

```
<?php
// include database connection file
require_once'dbconfig.php';
if(isset($_POST['insert']))
{
// Posted Values
$fname=$_POST['firstname'];
$lname=$_POST['lastname'];
$emailid=$_POST['emailid'];
$contactno=$_POST['contactno'];
$address=$_POST['address'];
// Query for Insertion
$sql="INSERT INTO tblusers(FirstName,LastName,EmailId,ContactNumber,Address
) VALUES(:fn,:ln,:eml,:cno,:adrss)";
//Prepare Query for Execution
$query = $dbh->prepare($sql);
// Bind the parameters
$query->bindParam(':fn',$fname,PDO::PARAM_STR);
$query->bindParam(':ln',$lname,PDO::PARAM_STR);
$query->bindParam(':eml',$emailid,PDO::PARAM_STR);
```

```
$query->bindParam(':cno',$contactno,PDO::PARAM_STR);
$query->bindParam(':adrss',$address,PDO::PARAM_STR);
// Query Execution
$query->execute();
// Check that the insertion really worked. If the last inserted id is great
er than zero, the insertion worked.
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId)
{
// Message for successfull insertion
echo "<script>alert('Record inserted successfully');</script>";
echo "<script>window.location.href='index.php'</script>";
}
else
{
// Message for unsuccessfull insertion
echo "<script>alert('Something went wrong. Please try again');</script>";
echo "<script>window.location.href='index.php'</script>";
}
}
?>
```

Step 4 : Read record from the database

Create a index.php file for read all records from database.

```
<?php
// include database connection file
require_once'dbconfig.php';  ?>
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
 <title>PHP CRUD Operations using PDO Extension </title>
 <meta name="viewport" content="width=device-width, initial-scale=1">
 <link href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.m
in.css" rel="stylesheet">
    <style type="text/css">
  </style>
    <script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
    <script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.0/js/bootstra
p.min.js"></script>
</head>
<body>
<div class="container">
<div class="row">
<div class="col-md-12">
<h3>PHP CRUD Operations using PDO Extension</h3> <hr />
<a href="insert.php"><button class="btn btn-
primary"> Insert Record</button></a>
<div class="table-responsive">
<table id="mytable" class="table table-bordred table-striped">
<thead>
<th>#</th>
<th>First Name</th>
<th>Last Name</th>
<th>Email</th>
```

```
<th>Contact</th>
<th>Address</th>
<th>Posting Date</th>
<th>Edit</th>
<th>Delete</th>
</thead>
<tbody>

<?php
$sql = "SELECT FirstName,LastName,EmailId,ContactNumber,Address,PostingDate
,id from tblusers";
//Prepare the query:
$query = $dbh->prepare($sql);
//Execute the query:
$query->execute();
//Assign the data which you pulled from the database (in the preceding step
) to a variable.
$results=$query->fetchAll(PDO::FETCH_OBJ);
// For serial number initialization
$cnt=1;
if($query->rowCount() > 0)
{
//In case that the query returned at least one record, we can echo the reco
rds within a foreach loop:
foreach($results as $result)
{
?>
<!-- Display Records -->
    <tr>
    <td><?php echo htmlentities($cnt);?></td>
    <td><?php echo htmlentities($result->FirstName);?></td>
    <td><?php echo htmlentities($result->LastName);?></td>
    <td><?php echo htmlentities($result->EmailId);?></td>
    <td><?php echo htmlentities($result->ContactNumber);?></td>
    <td><?php echo htmlentities($result->Address);?></td>
    <td><?php echo htmlentities($result->PostingDate);?></td>

<td><a href="update.php?id=<?php echo htmlentities($result-
>id);?>"><button class="btn btn-primary btn-
xs"><span class="glyphicon glyphicon-pencil"></span></button></a></td>

<td><a href="index.php?del=<?php echo htmlentities($result-
>id);?>"><button class="btn btn-danger btn-
xs" onClick="return confirm('Do you really want to delete');"><span class="
glyphicon glyphicon-trash"></span></button></a></td>
    </tr>

<?php
// for serial number increment
$cnt++;
}} ?>
</tbody>
</table>
</div>
</div>
</div>
</div>
</body>
 </html>
```

Step 5 : Update record in the database

Step 5.1 : Get data in HTML From

Create update.php file. For updating a record we have to get the row id of that record and store in $id. We access the $_GET[‘id’] variable to do it.

Code for get a record based on the given id. Through this way we can get data autofill-data in HTML Form.

```
<?php
// Get the userid
$userid=intval($_GET['id']);
$sql = "SELECT FirstName,LastName,EmailId,ContactNumber,Address,PostingDate
,id from tblusers where id=:uid";
//Prepare the query:
$query = $dbh->prepare($sql);
//Bind the parameters
$query->bindParam(':uid',$userid,PDO::PARAM_STR);
//Execute the query:
$query->execute();
//Assign the data which you pulled from the database (in the preceding step
) to a variable.
$results=$query->fetchAll(PDO::FETCH_OBJ);
// For serial number initialization
$cnt=1;
if($query->rowCount() > 0)
{
//In case that the query returned at least one record, we can echo the reco
rds within a foreach loop:
foreach($results as $result)
{
?>
<form name="insertrecord" method="post">
<div class="row">
<div class="col-md-4"><b>First Name</b>
<input type="text" name="firstname" value="<?php echo htmlentities($result-
>FirstName);?>" class="form-control" required>
</div>
<div class="col-md-4"><b>Last Name</b>
<input type="text" name="lastname" value="<?php echo htmlentities($result-
>LastName);?>" class="form-control" required>
</div>
</div>
<div class="row">
<div class="col-md-4"><b>Email id</b>
<input type="email" name="emailid" value="<?php echo htmlentities($result-
>EmailId);?>" class="form-control" required>
</div>
<div class="col-md-4"><b>Contactno</b>
<input type="text" name="contactno" value="<?php echo htmlentities($result-
>ContactNumber);?>" class="form-control" maxlength="10" required>
</div>
</div>
```

```
<div class="row">
<div class="col-md-8"><b>Address</b>
<textarea class="form-
control" name="address" required><?php echo htmlentities($result-
>Address);?>
</div>
</div>
<?php }} ?>

<div class="row" style="margin-top:1%">
<div class="col-md-8">
<input type="submit" name="update" value="Update">
</div>
</div>
```

Step 5.2 : Code for update the record

```
<?php
// include database connection file
require_once'dbconfig.php';
if(isset($_POST['update']))
{
// Get the userid
$userid=intval($_GET['id']);
// Posted Values
$fname=$_POST['firstname'];
$lname=$_POST['lastname'];
$emailid=$_POST['emailid'];
$contactno=$_POST['contactno'];
$address=$_POST['address'];
// Query for Updation
$sql="update tblusers set FirstName=:fn,LastName=:ln,EmailId=:eml,ContactNu
mber=:cno,Address=:adrss where id=:uid";
//Prepare Query for Execution
$query = $dbh->prepare($sql);
// Bind the parameters
$query->bindParam(':fn',$fname,PDO::PARAM_STR);
$query->bindParam(':ln',$lname,PDO::PARAM_STR);
$query->bindParam(':eml',$emailid,PDO::PARAM_STR);
$query->bindParam(':cno',$contactno,PDO::PARAM_STR);
$query->bindParam(':adrss',$address,PDO::PARAM_STR);
$query->bindParam(':uid',$userid,PDO::PARAM_STR);
// Query Execution
$query->execute();
// Mesage after updation
echo "<script>alert('Record Updated successfully');</script>";
// Code for redirection
echo "<script>window.location.href='index.php'</script>";
}
?>
```

Step 6 : Delete a record from the database

Place this code in the index.php file.

```php
<?php
// include database connection file
require_once'dbconfig.php';
// Code for record deletion
if(isset($_REQUEST['del']))
{
//Get row id
$uid=intval($_GET['del']);
//Qyery for deletion
$sql = "delete from tblusers WHERE  id=:id";
// Prepare query for execution
$query = $dbh->prepare($sql);
// bind the parameters
$query-> bindParam(':id',$uid, PDO::PARAM_STR);
// Query Execution
$query -> execute();
// Mesage after updation
echo "<script>alert('Record Updated successfully');</script>";
// Code for redirection
echo "<script>window.location.href='index.php'</script>";
}

?>
```

Chapter 7

Signup and Login operation using PDO

File structure for Sign up and Login operation

- **signup.php–** For user signup or user registration
- **check_availability.php–** For username and email-id availabilty
- **index.php–** For Userlogin
- **welcome.php–** After login user redirect to this page
- **logout.php–** Logout page
- **config.php–** Database connection file
- **pdosignup.sql–** Sql table structure for signup

Step 1– Create a database

Open browser type http://localhost/phpmyadmin, create a database named ‘pdosignup’ . After creating database run the sql script or import the sql file given inside the package.

Structure for sql table userdata

```
CREATE TABLE IF NOT EXISTS `userdata` (
  `id` int(11) NOT NULL,
  `FullName` varchar(200) DEFAULT NULL,
  `UserName` varchar(12) DEFAULT NULL,
  `UserEmail` varchar(200) DEFAULT NULL,
  `UserMobileNumber` varchar(10) DEFAULT NULL,
  `LoginPassword` varchar(255) DEFAULT NULL,
  `RegDate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1;

-- Indexes for table `userdata`
--
ALTER TABLE `userdata`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `userdata`
--
ALTER TABLE `userdata`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=8;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Step 2– Create a database configuration file config.php

```php
<?php
// DB credentials.
define('DB_HOST','localhost'); // Host name
define('DB_USER','root'); // db user name
define('DB_PASS',''); // db user password name
define('DB_NAME','pdosignup'); // db name
// Establish database connection.
try
{
$dbh = new PDO("mysql:host=".DB_HOST.";dbname=".DB_NAME,DB_USER, DB_PASS);

}
catch (PDOException $e)
{
exit("Error: " . $e->getMessage());
}
?>
```

Step 3– Create a Signup or Registration form signup.php

Step3.1 *HTML form with html5 validation pattern. In this form every input filed valid with*

HTML5 pattern validation

Fullname can contain any letters only with spaces

pattern=”[a-zA-Z\s]+”

Username can contain any letters or numbers, without spaces 6 to 12 chars

pattern=”^[a-zA-Z][a-zA-Z0-9-_.]{5,12}$”

email accepts valid email address

type=“email”

Mobile Number Contain only 10 digit numeric values

pattern=”[0-9]{10}” maxlength=”10″

Password should be at least 4 characters

pattern=”^\S{4,}$” onchange=”this.setCustomValidity(this.validity.patternMismatch ? ‘Must have at least 4 characters’ : ”); if(this.checkValidity()) form.password_two.pattern = this.value;”

Confirm Password should be same as password

pattern=”^\S{4,}$” onchange=”this.setCustomValidity(this.validity.patternMismatch ? ‘Please enter the same Password as above’ : ”)”

```
<form class="form-horizontal" action='' method="post">
<div class="control-group">
<!-- Full name -->
<label class="control-label"  for="fullname">Full Name</label>
<div class="controls">
<input type="text" id="fname" name="fname"  pattern="[a-zA-Z\s]+" title="Full name must contain letters only" class="input-xlarge" required>
<p class="help-block">Username can contain any letters or numbers, without spaces</p>
</div>
</div>


    <div class="control-group">
      <!-- Username -->
      <label class="control-label"  for="username">Username</label>
      <div class="controls">
        <input type="text" id="username" name="username" onBlur="checkUsernameAvailability()"  pattern="^[a-zA-Z][a-zA-Z0-9-_.]{5,12}$" title="User must be alphanumeric without spaces 6 to 12 chars" class="input-xlarge" required>
 <!-- Message for username availability-->
<span id="username-availability-status" style="font-size:12px;"></span>
        <p class="help-block">Username can contain any letters or numbers, without spaces</p>
      </div>
    </div>

    <div class="control-group">
      <!-- E-mail -->
      <label class="control-label" for="email">E-mail</label>
      <div class="controls">
        <input type="email" id="email" name="email" placeholder="" onBlur="checkEmailAvailability()" class="input-xlarge" required>
             <!-- Message for Email availability-->
<span id="email-availability-status" style="font-size:12px;"></span>
        <p class="help-block">Please provide your E-mail</p>
      </div>
    </div>

    <div class="control-group">
      <!-- Mobile Number -->
      <label class="control-label" for="mobilenumber">Mobile Number </label>
      <div class="controls">
        <input type="text" id="mobilenumber" name="mobilenumber" pattern="[0-9]{10}" maxlength="10"  title="10 numeric digits only"   class="input-xlarge" required>
        <p class="help-block">Mobile Number Contain only numeric values,net</p>
      </div>
```

```
    </div>
    <div class="control-group">
      <!-- Password-->
      <label class="control-label" for="password">Password</label>
      <div class="controls">
        <input type="password" id="password" name="password" pattern="^\S{4,}$" onchange="this.setCustomValidity(this.validity.patternMismatch ? 'Must have at least 4 characters' : ''); if(this.checkValidity()) form.password_two.pattern = this.value;"  required class="input-xlarge">
        <p class="help-block">Password should be at least 4 characters</p>
      </div>
    </div>

    <div class="control-group">
      <!-- Confirm Password -->
      <label class="control-label"  for="password_confirm">Password (Confirm)</label>
      <div class="controls">
        <input type="password" id="password_confirm" name="password_confirm" pattern="^\S{4,}$" onchange="this.setCustomValidity(this.validity.patternMismatch ? 'Please enter the same Password as above' : '')""  class="input-xlarge">
        <p class="help-block">Please confirm password</p>
      </div>
    </div>
<div class="control-group">
      <!-- Button -->
      <div class="controls">
        <button class="btn btn-success" type="submit" name="signup">Signup </button>

      </div>
</form>
```

Step3.2 *Check the username and email availability in the data using j-query*

j-query code

```
<!--Javascript for check username availability-->
<script>
function checkUsernameAvailability() {
$("#loaderIcon").show();
jQuery.ajax({
url: "check_availability.php",
data:'username='+$("#username").val(),
type: "POST",
success:function(data){
$("#username-availability-status").html(data);
$("#loaderIcon").hide();
},
error:function (){
}
});
}
</script>

<!--Javascript for check email availability-->
```

```
<script>
function checkEmailAvailability() {
$("#loaderIcon").show();
jQuery.ajax({
url: "check_availability.php",
data:'email='+$("#email").val(),
type: "POST",
success:function(data){

$("#email-availability-status").html(data);
$("#loaderIcon").hide();
},
error:function (){
 event.preventDefault();
}
});
}
</script>
```

PHP Code for checking availability in the database (check_availability.php)

```
<?php
require_once("config.php");
// Code for checking username availabilty
if(!emptyempty($_POST["username"])) {
$uname= $_POST["username"];
$sql ="SELECT UserName FROM  userdata WHERE UserName=:uname";
$query= $dbh -> prepare($sql);
$query-> bindParam(':uname', $uname, PDO::PARAM_STR);
$query-> execute();
$results = $query -> fetchAll(PDO::FETCH_OBJ);
if($query -> rowCount() > 0)
{
echo "<span style='color:red'> Username already exists.</span>";
} else{
echo "<span style='color:green'> Username available for Registration.</span
>";
}
}

// Code for checking email availabilty
if(!emptyempty($_POST["email"])) {
$email= $_POST["email"];
$sql ="SELECT UserEmail FROM  userdata WHERE UserEmail=:email";
$query= $dbh -> prepare($sql);
$query-> bindParam(':email', $email, PDO::PARAM_STR);
$query-> execute();
$results = $query -> fetchAll(PDO::FETCH_OBJ);
if($query -> rowCount() > 0)
{
echo "<span style='color:red'>Email-id already exists.</span>";
} else{
echo "<span style='color:green'>Email-
id available for Registration.</span>";
}
}

```

```
35. ?>
```

Step3.3 *code for inserting sign up values in the database*

```
1. <?php
2. //Database Configuration File
3. include('config.php');
4. error_reporting(0);
5. if(isset($_POST['signup']))
6. {
7. //Getting Post Values
8. $fullname=$_POST['fname'];
9. $username=$_POST['username'];
10.$email=$_POST['email'];
11.$mobile=$_POST['mobilenumber'];
12.$password=md5($_POST['password']);
13.// Query for validation of username and email-id
14.$ret="SELECT * FROM userdata where (UserName=:uname ||  UserEmail=:uemail)"
   ;
15.$queryt = $dbh -> prepare($ret);
16.$queryt->bindParam(':uemail',$email,PDO::PARAM_STR);
17.$queryt->bindParam(':uname',$username,PDO::PARAM_STR);
18.$queryt -> execute();
19.$results = $queryt -> fetchAll(PDO::FETCH_OBJ);
20.if($queryt -> rowCount() == 0)
21.{
22.// Query for Insertion
23.$sql="INSERT INTO userdata(FullName,UserName,UserEmail,UserMobileNumber,Log
   inPassword) VALUES(:fname,:uname,:uemail,:umobile,:upassword)";
24.$query = $dbh->prepare($sql);
25.// Binding Post Values
26.$query->bindParam(':fname',$fullname,PDO::PARAM_STR);
27.$query->bindParam(':uname',$username,PDO::PARAM_STR);
28.$query->bindParam(':uemail',$email,PDO::PARAM_STR);
29.$query->bindParam(':umobile',$mobile,PDO::PARAM_INT);
30.$query->bindParam(':upassword',$password,PDO::PARAM_STR);
31.$query->execute();
32.$lastInsertId = $dbh->lastInsertId();
33.if($lastInsertId)
34.{
35.$msg="You have signup  Scuccessfully";
36.}
37.else
38.{
39.$error="Something went wrong.Please try again";
40.}
41.}
42. else
43.{
44.$error="Username or Email-id already exist. Please try again";
45.}
46.}
47.?>
```

Here is the full code that we have written for signup(signup.php) :

```php
<?php
//Database Configuration File
include('config.php');
error_reporting(0);
if(isset($_POST['signup']))
{
//Getting Post Values
$fullname=$_POST['fname'];
$username=$_POST['username'];
$email=$_POST['email'];
$mobile=$_POST['mobilenumber'];
$password=md5($_POST['password']);
// Query for validation of username and email-id
$ret="SELECT * FROM userdata where (UserName=:uname ||  UserEmail=:uemail)"
;
$queryt = $dbh -> prepare($ret);
$queryt->bindParam(':uemail',$email,PDO::PARAM_STR);
$queryt->bindParam(':uname',$username,PDO::PARAM_STR);
$queryt -> execute();
$results = $queryt -> fetchAll(PDO::FETCH_OBJ);
if($queryt -> rowCount() == 0)
{
// Query for Insertion
$sql="INSERT INTO userdata(FullName,UserName,UserEmail,UserMobileNumber,Log
inPassword) VALUES(:fname,:uname,:uemail,:umobile,:upassword)";
$query = $dbh->prepare($sql);
// Binding Post Values
$query->bindParam(':fname',$fullname,PDO::PARAM_STR);
$query->bindParam(':uname',$username,PDO::PARAM_STR);
$query->bindParam(':uemail',$email,PDO::PARAM_STR);
$query->bindParam(':umobile',$mobile,PDO::PARAM_INT);
$query->bindParam(':upassword',$password,PDO::PARAM_STR);
$query->execute();
$lastInsertId = $dbh->lastInsertId();
if($lastInsertId)
{
$msg="You have signup  Scuccessfully";
}
else
{
$error="Something went wrong.Please try again";
}
}
 else
{
$error="Username or Email-id already exist. Please try again";
}
}
?>

<!DOCTYPE html>
```

```
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>PDO | Registration Form</title>
        <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://netdna.bootstrapcdn.com/twitter-bootstrap/2.3.2/css/bootstrap-combined.min.css" rel="stylesheet">
    <script src="https://code.jquery.com/jquery-1.11.1.min.js"></script>
    <script src="https://netdna.bootstrapcdn.com/twitter-bootstrap/2.3.2/js/bootstrap.min.js"></script>
  <style>
        .errorWrap {
    padding: 10px;
    margin: 0 0 20px 0;
    background: #fff;
    border-left: 4px solid #dd3d36;
    -webkit-box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
    box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
}
.succWrap{
    padding: 10px;
    margin: 0 0 20px 0;
    background: #fff;
    border-left: 4px solid #5cb85c;
    -webkit-box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
    box-shadow: 0 1px 1px 0 rgba(0,0,0,.1);
}
        </style>
<!--Javascript for check username availability-->
<script>
function checkUsernameAvailability() {
$("#loaderIcon").show();
jQuery.ajax({
url: "check_availability.php",
data:'username='+$("#username").val(),
type: "POST",
success:function(data){
$("#username-availability-status").html(data);
$("#loaderIcon").hide();
},
error:function (){
}
});
}
</script>

<!--Javascript for check email availability-->
<script>
function checkEmailAvailability() {
$("#loaderIcon").show();
jQuery.ajax({
url: "check_availability.php",
      data:'email='+$("#email").val(),
      type: "POST",
      success:function(data){

      $("#email-availability-status").html(data);
      $("#loaderIcon").hide();
```

```
106.       },
107.       error:function (){
108.        event.preventDefault();
109.       }
110.       });
111.       }
112.       </script>
113.
114.
115.       </head>
116.       <body>
117.       <form class="form-horizontal" action='' method="post">
118.         <fieldset>
119.           <div id="legend" style="padding-left:4%">
120.             <legend class="">Register | <a href="index.php">Sign in</a></l
    egend>
121.           </div>
122.       <!--Error Message-->
123.         <?php if($error){ ?><div class="errorWrap">
124.                       <strong>Error </strong> : <?php echo htmlentities($e
    rror);?></div>
125.                       <?php } ?>
126.       <!--Success Message-->
127.       <?php if($msg){ ?><div class="succWrap">
128.                       <strong>Well Done </strong> : <?php echo htmlentitie
    s($msg);?></div>
129.                       <?php } ?>
130.
131.        <div class="control-group">
132.             <!-- Full name -->
133.             <label class="control-
    label"  for="fullname">Full Name</label>
134.             <div class="controls">
135.               <input type="text" id="fname" name="fname"  pattern="[a-zA-
    Z\s]+" title="Full name must contain letters only" class="input-
    xlarge" required>
136.               <p class="help-
    block">Full can contain any letters only</p>
137.             </div>
138.           </div>
139.           <div class="control-group">
140.             <!-- Username -->
141.             <label class="control-
    label"  for="username">Username</label>
142.             <div class="controls">
143.               <input type="text" id="username" name="username" onBlur="che
    ckUsernameAvailability()"  pattern="^[a-zA-Z][a-zA-Z0-9-
    _.]{5,12}$" title="User must be alphanumeric without spaces 6 to 12 chars"
    class="input-xlarge" required>
144.                     <span id="username-availability-status" style="font-
    size:12px;"></span>
145.               <p class="help-
    block">Username can contain any letters or numbers, without spaces 6 to 12
    chars </p>
146.             </div>
147.           </div>
148.
149.           <div class="control-group">
150.             <!-- E-mail -->
```

```
151.            <label class="control-label" for="email">E-mail</label>
152.            <div class="controls">
153.              <input type="email" id="email" name="email" placeholder="" o
   nBlur="checkEmailAvailability()" class="input-xlarge" required>
154.                    <span id="email-availability-status" style="font-
   size:12px;"></span>
155.              <p class="help-block">Please provide your E-mail</p>
156.            </div>
157.          </div>
158.
159.          <div class="control-group">
160.            <!-- Mobile Number -->
161.            <label class="control-
   label" for="mobilenumber">Mobile Number </label>
162.            <div class="controls">
163.              <input type="text" id="mobilenumber" name="mobilenumber" pat
   tern="[0-
   9]{10}" maxlength="10"  title="10 numeric digits only"   class="input-
   xlarge" required>
164.              <p class="help-
   block">Mobile Number Contain only 10 digit numeric values</p>
165.            </div>
166.          </div>
167.          <div class="control-group">
168.            <!-- Password-->
169.            <label class="control-label" for="password">Password</label>
170.            <div class="controls">
171.              <input type="password" id="password" name="password" pattern
   ="^\S{4,}$" onchange="this.setCustomValidity(this.validity.patternMismatch
   ? 'Must have at least 4 characters' : ''); if(this.checkValidity()) form.pa
   ssword_two.pattern = this.value;"  required class="input-xlarge">
172.              <p class="help-
   block">Password should be at least 4 characters</p>
173.            </div>
174.          </div>
175.
176.          <div class="control-group">
177.            <!-- Confirm Password -->
178.            <label class="control-
   label"  for="password_confirm">Password (Confirm)</label>
179.            <div class="controls">
180.              <input type="password" id="password_confirm" name="password_
   confirm" pattern="^\S{4,}$" onchange="this.setCustomValidity(this.validity.
   patternMismatch ? 'Please enter the same Password as above' : '')""  class=
   "input-xlarge">
181.              <p class="help-block">Please confirm password</p>
182.            </div>
183.          </div>
184.
185.          <div class="control-group">
186.            <!-- Button -->
187.            <div class="controls">
188.              <button class="btn btn-
   success" type="submit" name="signup">Signup </button>
189.            </div>
190.          </div>
191.        </fieldset>
192.      </form>
193.      <script type="text/javascript">
```

```
194.
195.       </script>
196.       </body>
197.       </html>
```

Step 4– Signin or Login

Create a HTML sign form

```
1.  <form id="loginForm" method="post">
2.  div class="form-group">
3.  <label for="username" class="control-label">Username / Email id</label>
4.  <input type="text" class="form-
    control" id="username" name="username"  required="" title="Please enter you
     username or Email-id" placeholder="email or username" >
5.   <span class="help-block"></span>
6.   </div>
7.  <div class="form-group">
8.  <label for="password" class="control-label">Password</label>
9.  <input type="password" class="form-
    control" id="password" name="password" placeholder="Password" value="" requ
    ired="" title="Please enter your password">
10.  <span class="help-block"></span>
11. </div>
12.
13. <button type="submit" class="btn btn-success btn-
    block" name="login">Login</button>
14. </form>
```

User can login through valid username/email and password.

PHP Code for Signin or Login

```
1.  <?php
2.    session_start();
3.  //Database Configuration File
4.  include('config.php');
5.  error_reporting(0);
6.    if(isset($_POST['login']))
7.    {
8.      // Getting username/ email and password
9.      $uname=$_POST['username'];
10.     $password=md5($_POST['password']);
11.     // Fetch data from database on the basis of username/email and password
12.     $sql ="SELECT UserName,UserEmail,LoginPassword FROM userdata WHERE (Use
    rName=:usname || UserEmail=:usname) and (LoginPassword=:usrpassword)";
13.     $query= $dbh -> prepare($sql);
14.     $query-> bindParam(':usname', $uname, PDO::PARAM_STR);
15.     $query-> bindParam(':usrpassword', $password, PDO::PARAM_STR);
16.     $query-> execute();
17.     $results=$query->fetchAll(PDO::FETCH_OBJ);
18.   if($query->rowCount() > 0)
19.   {
20.     $_SESSION['userlogin']=$_POST['username'];
```

```
    echo "<script type='text/javascript'> document.location = 'welcome.php' ; </script>";
  } else{
    echo "<script>alert('Invalid Details');</script>";
  }
}
?>
```

Step 5– Create a welcome page.

After login user will redirect to this page. Validate this page using user session if anyone want to access this page without login will redirect to index.php automatically.

Code for welcome.php

```
<?php
session_start();
include('config.php');
// Validating Session
if(strlen($_SESSION['userlogin'])==0)
{
header('location:index.php');
}
else{
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>PDO | Welcome Page</title>
        <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://netdna.bootstrapcdn.com/twitter-bootstrap/2.3.2/css/bootstrap-combined.min.css" rel="stylesheet">
    <style type="text/css">
          .center {text-align: center; margin-left: auto; margin-right: auto; margin-bottom: auto; margin-top: auto;}

    </style>
    <script src="https://code.jquery.com/jquery-1.11.1.min.js"></script>
    <script src="https://netdna.bootstrapcdn.com/twitter-bootstrap/2.3.2/js/bootstrap.min.js"></script>
</head>
<body>
<div class="container">
  <div class="row">
    <div class="span12">
      <div class="hero-unit center">
<?php
// Code for fecthing user full name on the bassis of username or email.
$username=$_SESSION['userlogin'];
$query=$dbh->prepare("SELECT  FullName FROM userdata WHERE (UserName=:username || UserEmail=:username)");
      $query->execute(array(':username'=> $username));
```

```
        while($row=$query->fetch(PDO::FETCH_ASSOC)){
    $username=$row['FullName'];
        }
        ?>

          <h1>Welcome Back <font face="Tahoma" color="red"><?php echo $user
name;?> ! </font></h1>
          <br />
          <p>Lorem ipsum dolor sit amet, sit veniam senserit mediocritatem
et, melius aperiam complectitur an qui. Ut numquam vocibus accumsan mel. Pe
r ei etiam vituperatoribus, ne quot mandamus conceptam has, pri molestiae c
onstituam quaerendum an. In molestiae torquatos eam.
          </p>
          <a href="logout.php" class="btn btn-large btn-
info"><i class="icon-home icon-white"></i> Log me out</a>
        </div>
        <br />

      </div>
        <br />
        <!-- By ConnerT HTML & CSS Enthusiast -->
    </div>
  </div>
</div>

</body>
</html>
<?php } ?>
```

Step 6 – logout Page.

Logout page used for destroy the session or unset the session variable.

code for logout.php page

```
<?php
session_start();
unset($_SESSION['userlogin']); // unset session variable
session_destroy(); // destroy session
header("location:index.php");
?>
```

Download full script - https://phpgurukul.com/sign-up-and-login-operation-using-pdo/