# Password Tutorial

## Password Hashing

We are going to look at how to encrypt the password in our **registration.php** file and to store this encrypted version to our **users** table in our database.

We will then amend the **login.php** file to compare the password entered and verify it with the hashed version stored on the database.

Finally we need to ensure that the username is unique on the users table i.e. we can't have two users with the same username.

To start download the files from moodle under Password Files. This will create a password folder which contain the files we need.

Make sure you have Apache web server and MySql running

### Registration File

We'll start with the **register.php** file. Here we will make the change to hash the password. To do this we'll user the ***password_hash()*** function

```
        $phash = password_hash($password, PASSWORD_BCRYPT);
```

$phash is what we now will store to the database. So the file should look like (new code in red)

```
<?php
//
```

```
session_start();
//
$usernameErr = "";
$username = "";
$emailErr = "";
$email = "";
$passwordErr = "";
$password = "";
if($_POST){
        $username = $_POST['username'];
        $password = $_POST['password'];
          $email = $_POST['email'];

          if (empty($username) || strlen($username) < 4 )
              $usernameErr = "Username is required, at least 4 chars";
          if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
                $emailErr = 'Sorry you did not enter a correct email! ';
          }
          if (strlen($password) < 8 ) {
                $passwordErr = 'password 8 or characters';
          }

          if (empty($usernameErr) && empty($emailErr) && empty($passwordErr)) {

        try {
        $host = '127.0.0.1';
        $dbname = 'wdtest';
        $user = 'root';
        $pass = '';
        # MySQL with PDO_MYSQL
        $DBH = new PDO("mysql:host=$host;dbname=$dbname", $user, $pass);

          $phash = password_hash($password, PASSWORD_BCRYPT);

          $sql = "INSERT INTO Users (username, password, email) VALUES (?, ?, ?);";
          $sth = $DBH->prepare($sql);

          $sth->bindParam(1, $username);
          $sth->bindParam(2, $phash);
          $sth->bindParam(3, $email);

          $sth->execute();
          $_SESSION["username"] = $username;
          $_SESSION["email"] = $email;
          header('Location:  confirm.php');
          exit();
           } catch(PDOException $e) {echo $e->getMessage();}
          }
}
?>
<!DOCTYPE>
<html>
<head>
  <link rel="stylesheet" href="style.css">
    <style>
          .error {display: block;color: #FF0000; }
          </style>
</head>
<body>
<h2> Registration Form</h2>
<form class='form-style' action="register.php" method="post">
Username <input type="text" name="username" value="<?php echo $username; ?>"/>
        <span class = "error"><?php echo $usernameErr;?></span>
email <input type="text" name="email" value="<?php echo $email; ?>"/>
                    <span class = "error"><?php echo $emailErr;?></span>
Password <input type="password" name="password" value="<?php echo $password;
?>"/>
           <span class = "error"><?php echo $passwordErr;?></span>
<input type="submit" class='button' name='submit' value= 'Register'/>
</form>
</body>
</html>
```

Run the **register.php** file in a browser under localhost and register a new user.

Go to heidi and run a *select * from users;* in the query pane. You should see a new user with an encrypted password.

Now we need to make a change to make the user unique. To do this we will do a select on the users table with the username input on the form. If we return a row then we will show an error other wish we continue as before .

```
                $checkUserStmt = $DBH->prepare("select * from users where username
= ?" );
                $checkUserStmt->bindParam(1, $username);
                $checkUserStmt->execute();
```

To check if a row was returned we can use the ***rowCount()*** function

```
                if ($checkUserStmt->rowCount() == 0) {
                        .
                        .
                        .
                }
                else
                        echo 'Username already taken!';
```

So our final file looks like

```
<?php
//
session_start();
//
$usernameErr = "";
$username = "";
$emailErr = "";
```

```
$email = "";
$passwordErr = "";
$password = "";
if($_POST){
        $username = $_POST['username'];
        $password = $_POST['password'];
        $email = $_POST['email'];

        if (empty($username) || strlen($username) < 4 )
           $usernameErr = "Username is required, at least 4 chars";
        if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
                        $emailErr = 'Sorry you did not enter a correct email! ';
        }
        if (strlen($password) < 8 ) {
                $passwordErr = 'password 8 or characters';
        }

        if (empty($usernameErr) && empty($emailErr) && empty($passwordErr)) {

          try {
              $host = '127.0.0.1';
              $dbname = 'wdtest';
              $user = 'root';
              $pass = '';
              # MySQL with PDO_MYSQL
              $DBH = new PDO("mysql:host=$host;dbname=$dbname", $user, $pass);

                $checkUserStmt = $DBH->prepare("select * from users where username
= ?" );
                $checkUserStmt->bindParam(1, $username);
                $checkUserStmt->execute();

                if ($checkUserStmt->rowCount() == 0) { //no user with this name exists

                        $phash = password_hash($password, PASSWORD_BCRYPT);

                        $sql = "INSERT INTO Users (username, password, email) VALUES
(?, ?, ?);";
```

```
                        $sth = $DBH->prepare($sql);

                        $sth->bindParam(1, $username);
                        $sth->bindParam(2, $phash);
                        $sth->bindParam(3, $email);

                        $sth->execute();
                        $_SESSION["username"] = $username;
                        $_SESSION["email"] = $email;
                        header('Location:  confirm.php');
                        exit();
                }
                else
                        echo 'Username already taken!';
                } catch(PDOException $e) {echo $e->getMessage();}
        }
}
?>
<!DOCTYPE>
<html>
<head>
  <link rel="stylesheet" href="style.css">
    <style>
        .error {display: block;color: #FF0000; }
        </style>
</head>
<body>
<h2> Registration Form</h2>
<form class='form-style' action="register.php" method="post">
Username <input type="text" name="username" value="<?php echo $username; ?>"/>
       <span class = "error"><?php echo $usernameErr;?></span>
email <input type="text" name="email" value="<?php echo $email; ?>"/>
                <span class = "error"><?php echo $emailErr;?></span>
Password <input type="password" name="password" value="<?php echo $password;
?>"/>
         <span class = "error"><?php echo $passwordErr;?></span>
<input type="submit" class='button' name='submit' value= 'Register'/>
</form>
</body>
```

```
</html>
```

Before testing this you might want to delete the entries in the users table

```
delete from users;
```

Ignore the error message.

## Login File

Now we need to amend the login.php file to verify the encrypted password on database table to that entered by the user. To do this we will make use of the

*password_verify() function.*

We need to change our select statement from

```
$q = $DBH->prepare("select * from users where username = :username and
password = :password LIMIT 1");
```

To, as we no longer can use the password in the search terms

```
$q = $DBH->prepare("select * from users where username = :username LIMIT
1");
```

If the select statement returns 0 then we're good to go. We then proceed to verify the password. We can use the **rowCount()** function along with the **fetch()** function to return the password field from the select statement and we use the password verify function to verify the user entered password with that on the database

```
            $q = $DBH->prepare("select * from users where username = :username
LIMIT 1");
            $q->bindValue(':username', $username);
            $q->execute();

            $row = $q->fetch(PDO::FETCH_ASSOC);

            if ($q->rowCount() > 0) {
                    $phash = $row['password'];
                    if (password_verify($password,$phash)) {
                             $email = $row['email'];
                            $_SESSION["username"] = $username;
                            $_SESSION["email"] = $email;
                            header('Location:  confirm.php');
                            exit();
                    }
                    else
                            $message= 'Invalid password.';

            } else {
               $message= 'Sorry your log in details are not correct';
            }
```

## Our final file in its entirety

```
<?php
//
session_start();
//
$usernameErr = "";
$username = "";
$passwordErr = "";
$password = "";
if($_POST){
   $username = $_POST['username'];
   $password = $_POST['password'];
    if (empty($username) || strlen($username) < 4 )
```

```
                $usernameErr = "Username is required, at least 4 chars";
          if (strlen($password) < 8 ) {
                $passwordErr = 'password 8 or more chars';
          }

        if (empty($usernameErr) && empty($passwordErr)) {

           try {
             $host = '127.0.0.1';
             $dbname = 'wdtest';
             $user = 'root';
             $pass = '';

             # MySQL with PDO_MYSQL
             $DBH = new PDO("mysql:host=$host;dbname=$dbname", $user, $pass);

                         $q = $DBH->prepare("select * from users where username = :username
LIMIT 1");
                         $q->bindValue(':username', $username);
                         $q->execute();

                         $row = $q->fetch(PDO::FETCH_ASSOC);

                         if ($q->rowCount() > 0) {
                                 $phash = $row['password'];
                                 if (password_verify($password,$phash)) {
                                          $email = $row['email'];
                                         $_SESSION["username"] = $username;
                                         $_SESSION["email"] = $email;
                                         header('Location:  confirm.php');
                                         exit();
                                 }
                                 else
                                         $message= 'Invalid password.';

                         } else {
                            $message= 'Sorry your log in details are not correct';
                         }
              } catch(PDOException $e) {echo $e->getMessage();}
              }
        }
        ?>
        <!DOCTYPE>
```

```
<html>
<head>
  <link rel="stylesheet" href="style.css">
    <style>
            .error {display: block;color: #FF0000; }
            </style>
</head>
<body>
<h2>Login</h2><br></br>
<form class='form-style' action="login.php" method="post">
Username <input type="text" name="username" value="<?php echo $username; ?>"/>
                <span class = "error"><?php echo $usernameErr;?></span>
Password <input type="password" name="password" />
                <span class = "error"><?php echo $passwordErr;?></span>
<input type="submit" name="submit" value="Login" class='button'/>
<?php
if(!empty($message)){  echo '<br>';
echo $message;
}
?>
</form>
</body>
</html>
```

## Finally

We want to be able to pass the id of the row on the users table into a session variable. To do this in the **login.php** file simply add

```
$_SESSION["user_id"] = $row['id'];
```

After setting of the other session variables.

In **register.php** we can use a builtin function to get the id of the row that has been inserted to the users table. We use the **lastInsertId()** function. Add the following where the other session variables are set.

```
$_SESSION["user_id"] = $DBH->lastInsertId();
```

Now let's print out this id in the **confirm.php** file

Add the line

```
echo "<br/>id: ".$_SESSION["user_id"];
```

Test the application and make sure you see the id in the confirm file.