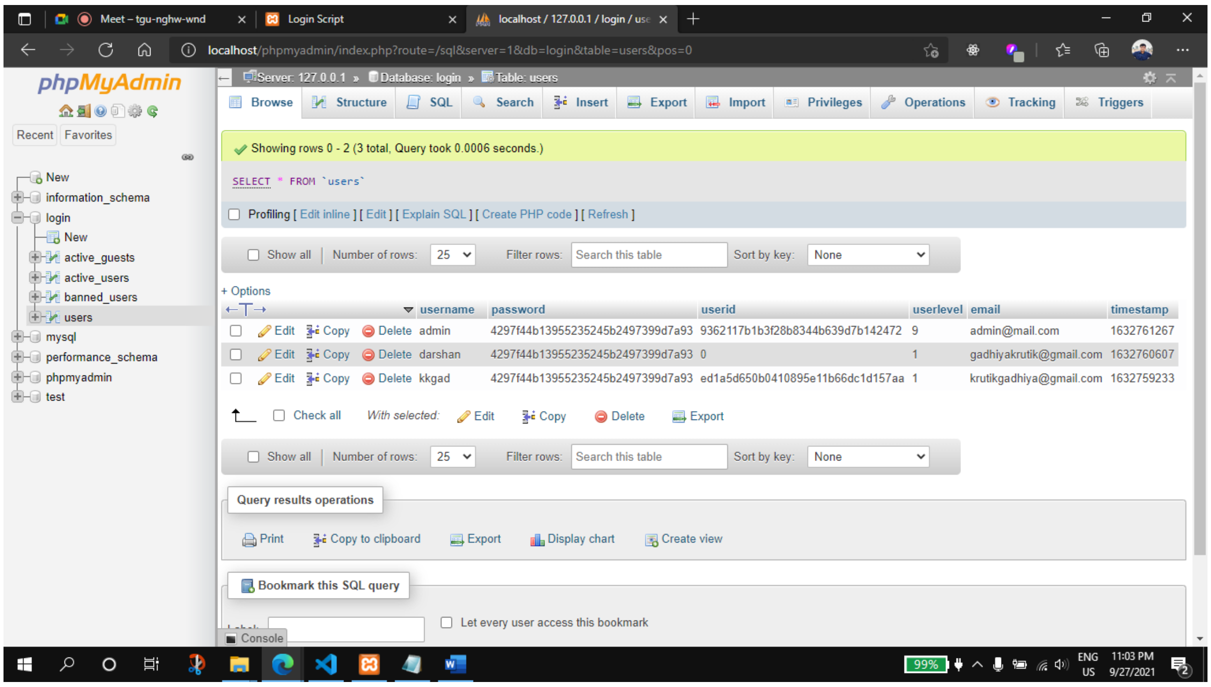
AWT Practical - 10 Task

Login Script Demo with PHP

This week we will demonstrate Login Script with the objective of learning the following concepts in the real world. Students will be required to create a report with implementation of all following concepts in any script of their choice with clear explanation-understanding of concepts.

1. MySql as a Database and its syntax
2. Database connection with my-sql
  a. OOP way of connecting database
  b. Procedural Way of Connecting Database
  c. PDO(PHP Data Objects)
3. Session management
4. Use of PHP Constants in practice
5. MVC and Code Structuring
6. Form error Handling
7. Includes & Redirections

1. In this practical we will be performing the login script in PHP, here we have used the MySql database, the syntax is pretty common for creating the table, insert data to table and much more.



localhost/phpmyadmin/index.php?route=/sql&server=1&db=login&table=users&pos=0

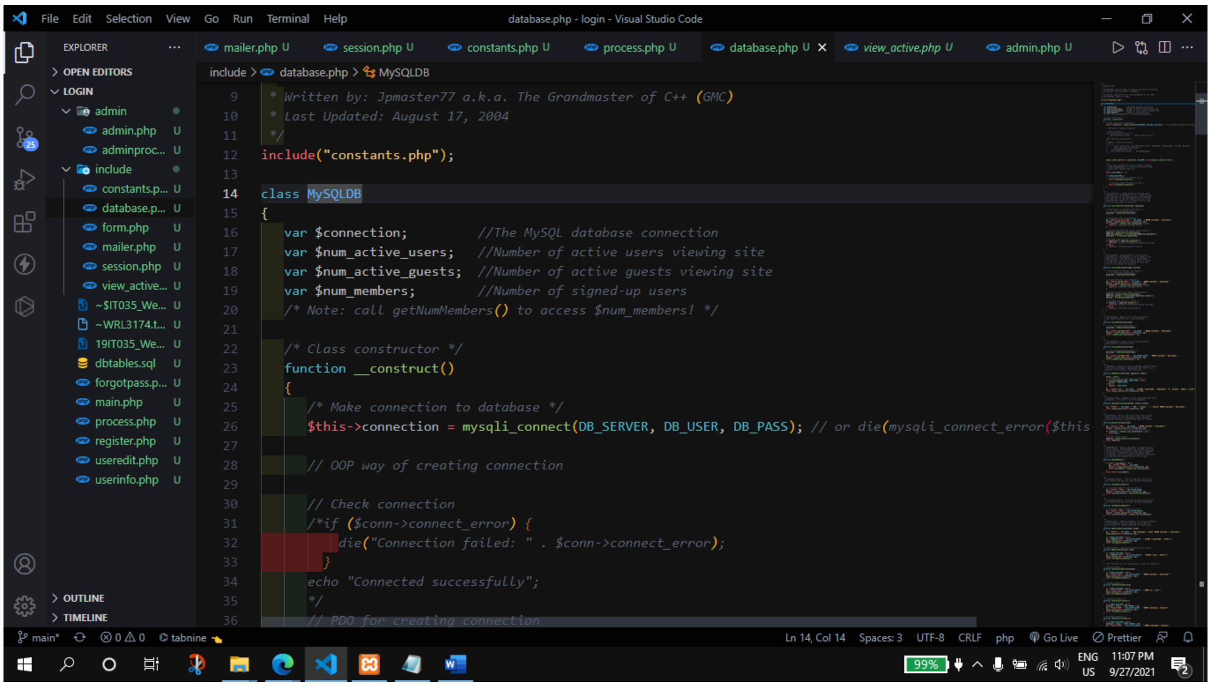
Server: 127.0.0.1 » Database: login » Table: users

Showing rows 0 - 2 (3 total, Query took 0.0006 seconds.)

```
SELECT * FROM `users`
```

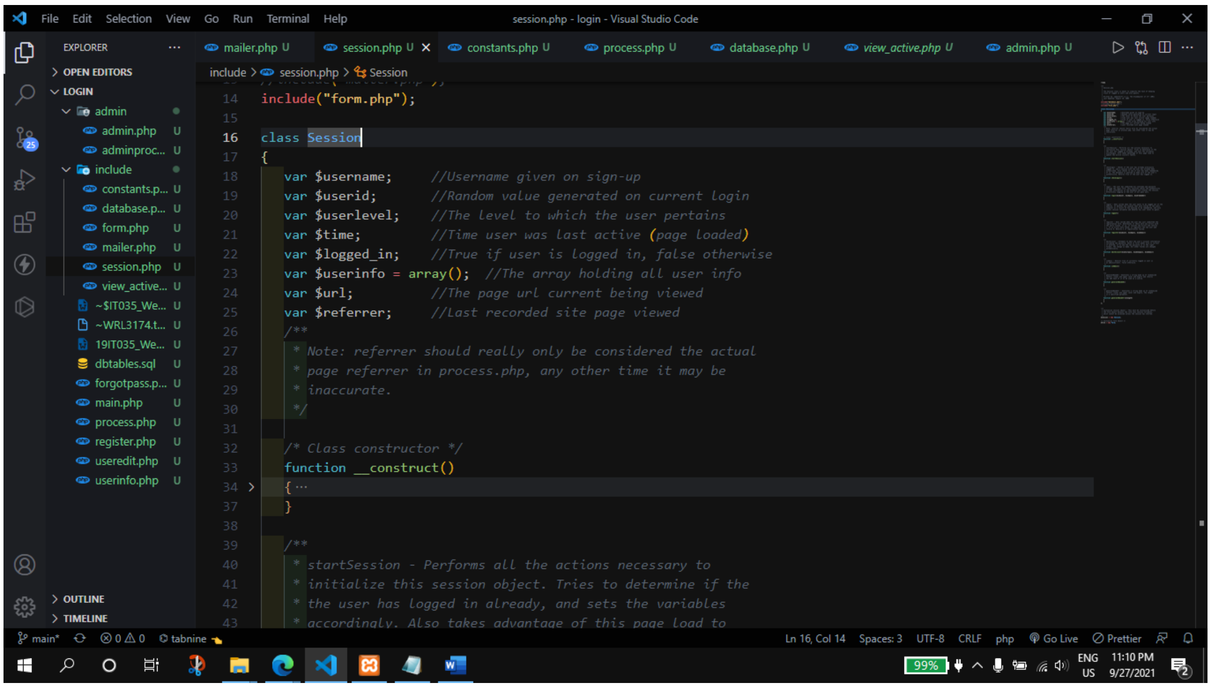
username	password	userid	userlevel	email	timestamp
admin	4297f44b13955235245b2497399d7a93	9362117b1b3f28b8344b639d7b142472	9	admin@mail.com	1632761267
darshan	4297f44b13955235245b2497399d7a93	0	1	gadhiyakrutik@gmail.com	1632760607
kkgad	4297f44b13955235245b2497399d7a93	ed1a5d650b0410895e11b66dc1d157aa	1	krutikgadhiya@gmail.com	1632759233

2. There are multiple ways we can connect database with the PHP as specially the MySql DB, 1st method is to use the mysqli_connect() method which is the procedural way to connecting, 2nd method is OOP way using new mysqli() and the final way is using PDO which stands for PHP data Objects in this particular practical I have used the procedural way.



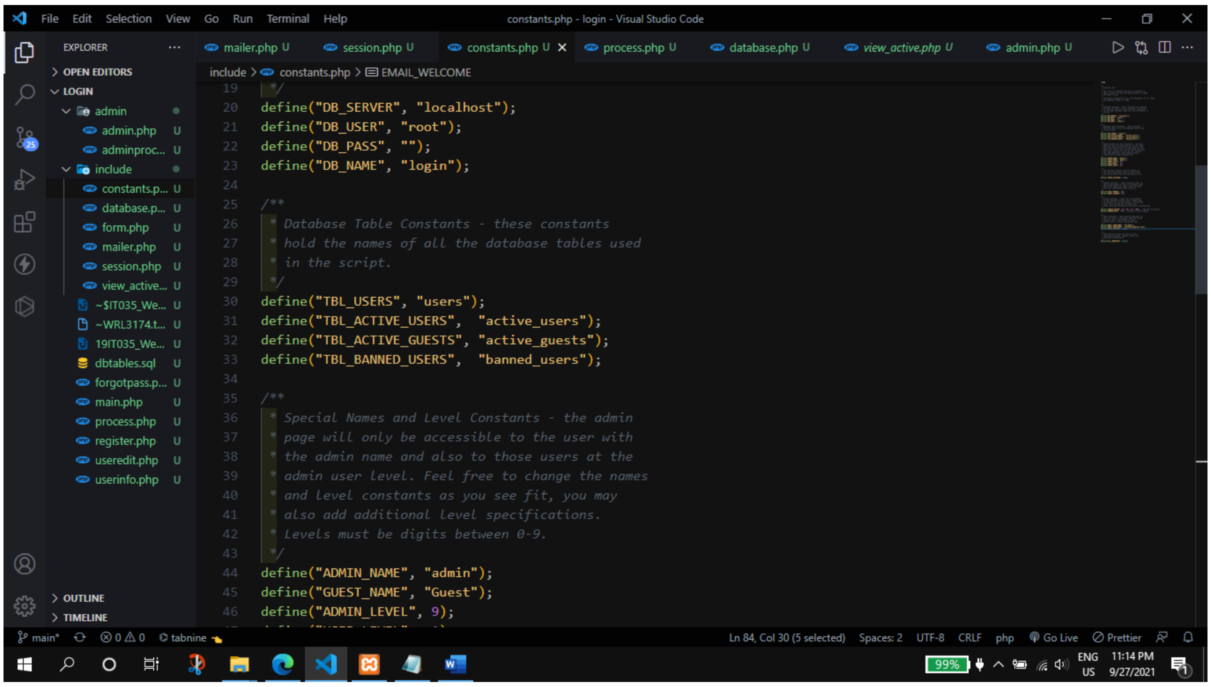
```
 * Written by: Jpmaster77 a.k.a. The Grandmaster of C++ (GMC)
 * Last Updated: August 17, 2004
 */
include("constants.php");

class MySQLDB
{
   var $connection;         //The MySQL database connection
   var $num_active_users;   //Number of active users viewing site
   var $num_active_guests;  //Number of active guests viewing site
   var $num_members;        //Number of signed-up users
   /* Note: call getNumMembers() to access $num_members! */

   /* Class constructor */
   function __construct()
   {
      /* Make connection to database */
      $this->connection = mysqli_connect(DB_SERVER, DB_USER, DB_PASS); // or die(mysqli_connect_error($this

      // OOP way of creating connection

      // Check connection
      /*if ($conn->connect_error) {
         die("Connection failed: " . $conn->connect_error);
      }
      echo "Connected successfully";
      */
      // PDO for creating connection
```

3. For the session management we have made a separate file with Session class, which contains various properties and methods which handles the different operations on the session



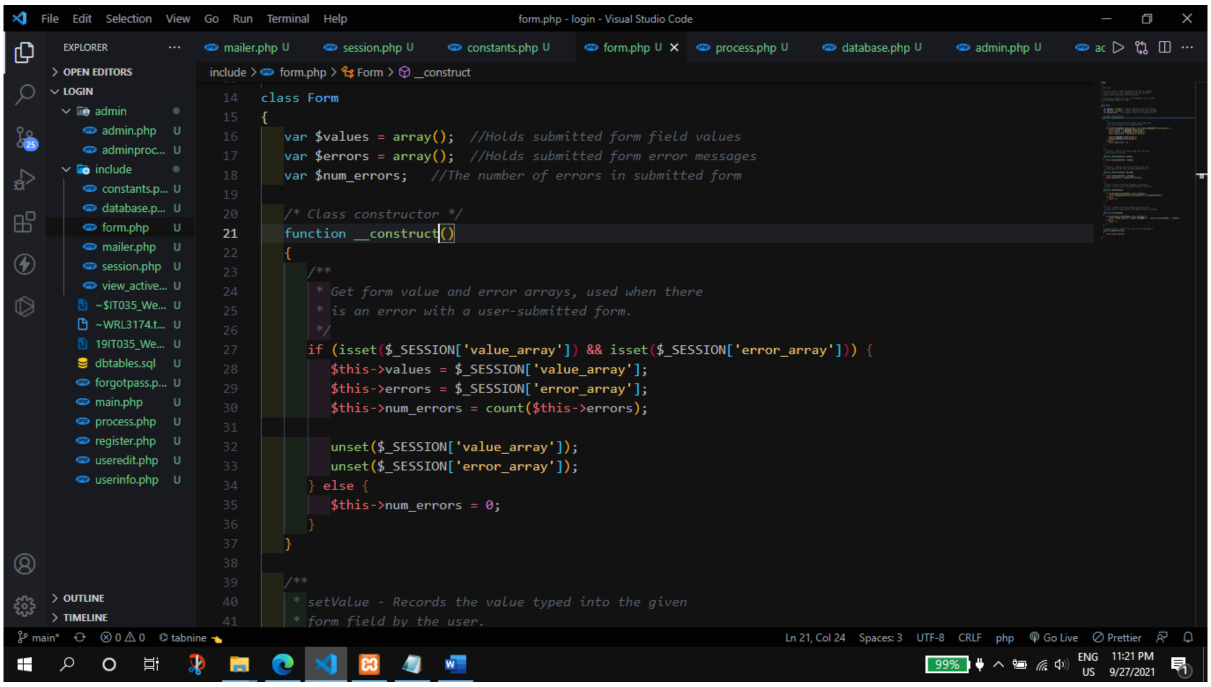
```
include("form.php");

class Session
{
   var $username;     //Username given on sign-up
   var $userid;       //Random value generated on current login
   var $userlevel;    //The level to which the user pertains
   var $time;         //Time user was last active (page loaded)
   var $logged_in;    //True if user is logged in, false otherwise
   var $userinfo = array();  //The array holding all user info
   var $url;          //The page url current being viewed
   var $referrer;     //Last recorded site page viewed
   /**
    * Note: referrer should really only be considered the actual
    * page referrer in process.php, any other time it may be
    * inaccurate.
    */

   /* Class constructor */
   function __construct()
   {…
   }

   /**
    * startSession - Performs all the actions necessary to
    * initialize this session object. Tries to determine if the
    * the user has logged in already, and sets the variables
    * accordingly. Also takes advantage of this page load to
```

4. the there is some date which is redundant and we need to use them again and again ad various places such as the table name, user types etc. so instead of hardcoding them we can define them in other file as constants as their values are not going to change and then we can use them in other files. The same we have done here the name of the file is given as Constant.php.



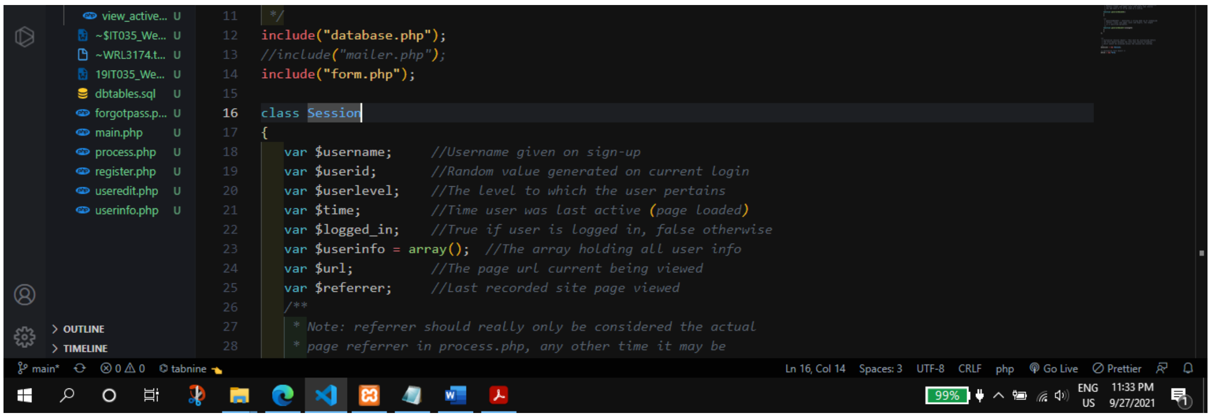
```php
 */
define("DB_SERVER", "localhost");
define("DB_USER", "root");
define("DB_PASS", "");
define("DB_NAME", "login");

/**
 * Database Table Constants - these constants
 * hold the names of all the database tables used
 * in the script.
 */
define("TBL_USERS", "users");
define("TBL_ACTIVE_USERS",  "active_users");
define("TBL_ACTIVE_GUESTS", "active_guests");
define("TBL_BANNED_USERS",  "banned_users");

/**
 * Special Names and Level Constants - the admin
 * page will only be accessible to the user with
 * the admin name and also to those users at the
 * admin user level. Feel free to change the names
 * and level constants as you see fit, you may
 * also add additional level specifications.
 * Levels must be digits between 0-9.
 */
define("ADMIN_NAME", "admin");
define("GUEST_NAME", "Guest");
define("ADMIN_LEVEL", 9);
```

5. MVC stands for model view and controller this is the model is code that actually perform the tasks such as database update, sending email etc. the view is that the user interacts with and the controller which sits between model and view and depending on the users action, tells the model to perform the task.
6. Form handling is also an important thing to do as we have to store the correct data so the form handling ensures the correct data is being entered to the database by performing check on the data received from the user and depending upon the type of the date we can instruct the user to what to do next.
   For the form we have created the separate file which contains the From class similar to the Session class which contains few properties and the method to handle the data.



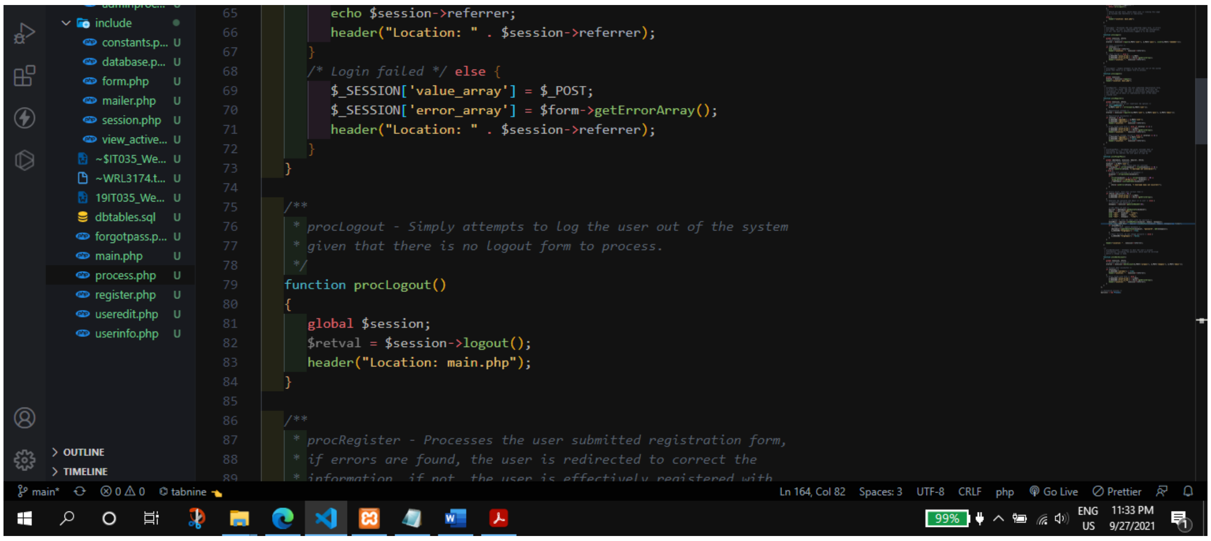
```php
class Form
{
   var $values = array();  //Holds submitted form field values
   var $errors = array();  //Holds submitted form error messages
   var $num_errors;   //The number of errors in submitted form

   /* Class constructor */
   function __construct()
   {
      /**
       * Get form value and error arrays, used when there
       * is an error with a user-submitted form.
       */
      if (isset($_SESSION['value_array']) && isset($_SESSION['error_array'])) {
         $this->values = $_SESSION['value_array'];
         $this->errors = $_SESSION['error_array'];
         $this->num_errors = count($this->errors);

         unset($_SESSION['value_array']);
         unset($_SESSION['error_array']);
      } else {
         $this->num_errors = 0;
      }
   }

   /**
    * setValue - Records the value typed into the given
    * form field by the user.
```

7. The separate files that we created we need to include them in to different files so that we can use sessions in that file, we can connect to database etc. that can be done using include() which takes one argument the name of the file we want to include



```php
 */
include("database.php");
//include("mailer.php");
include("form.php");

class Session
{
   var $username;     //Username given on sign-up
   var $userid;       //Random value generated on current login
   var $userlevel;    //The level to which the user pertains
   var $time;         //Time user was last active (page loaded)
   var $logged_in;    //True if user is logged in, false otherwise
   var $userinfo = array();  //The array holding all user info
   var $url;          //The page url current being viewed
   var $referrer;     //Last recorded site page viewed
   /**
    * Note: referrer should really only be considered the actual
    * page referrer in process.php, any other time it may be
```

For redirection the best way is query string as we can redirect user from the backend.



```php
         echo $session->referrer;
         header("Location: " . $session->referrer);
      }
      /* Login failed */ else {
         $_SESSION['value_array'] = $_POST;
         $_SESSION['error_array'] = $form->getErrorArray();
         header("Location: " . $session->referrer);
      }
   }

   /**
    * procLogout - Simply attempts to log the user out of the system
    * given that there is no logout form to process.
    */
   function procLogout()
   {
      global $session;
      $retval = $session->logout();
      header("Location: main.php");
   }

   /**
    * procRegister - Processes the user submitted registration form,
    * if errors are found, the user is redirected to correct the
    * information, if not, the user is effectively registered with
```

Problems/Issues found in the projects:

- missing php from <?php ?> at many places In **almost** all the files

in almost all the files the php from the php tag were missing and due to those incomplete tags the files are not detected as php so, most part of the system does not work

- wrong constructor name

in the classes the name of the constructors were wrong, the constructor were given the name of the class, but in the new PHP syntax the constructor is declared as __constructor()

- get_magic_quotes_gpc()

this method is deprecated and no longer supported and there is no meaning of writing/using it.

- regex expression not written correctly the regex expression is not written correctly it was missing the starting and ending /.

- mysql_result() method is depricitated

this method was used to get the contents of multiple cells in one function call, as it is deprecated, I found some alternate to it made another function which act like the same.

- mysql_numrows wrong method name

the correct name is mysqli_num_rows()

- <?php ?> was enclosed in double inverted coma

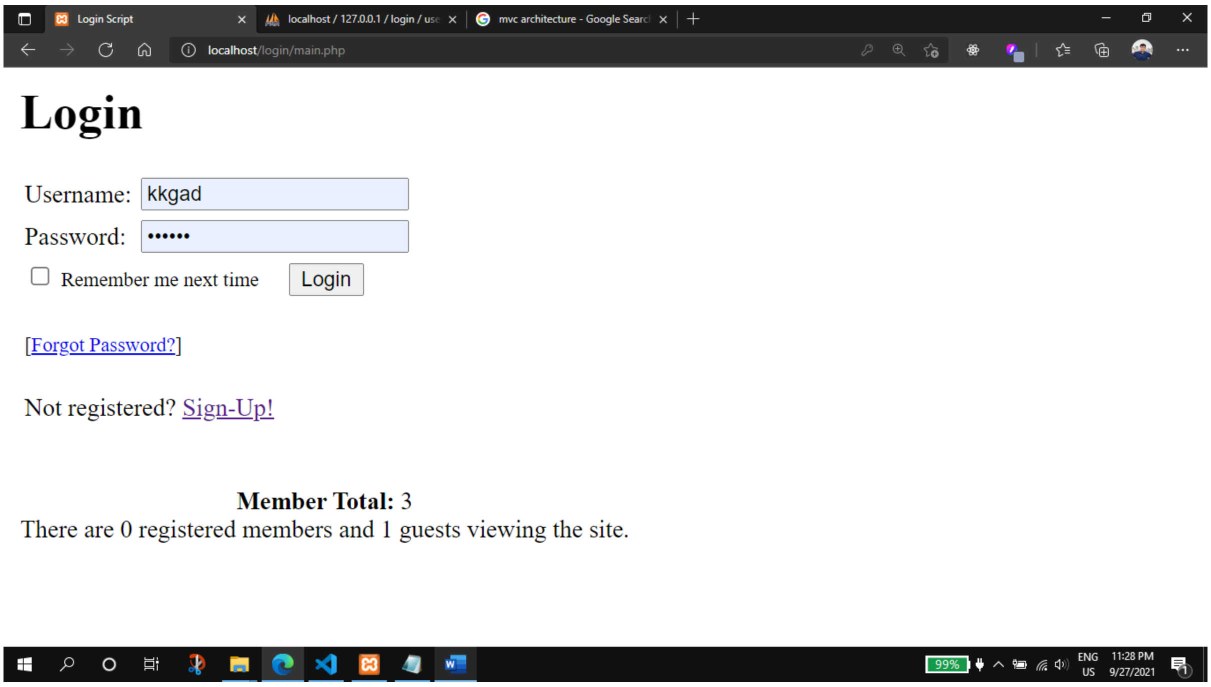
In many places in the form the <?php ?> tag was enclosed within double inverted commas and which is not correct.

- mysqli_query() paramater were swapped

the position of the parameter were swapped, so changing it the site start working.

- when user is deleted then the entry in active_user table is still their so added implementation for the same

Output:


Login Script
localhost / 127.0.0.1 / login / use
mvc architecture - Google Search
localhost/login/main.php

Login

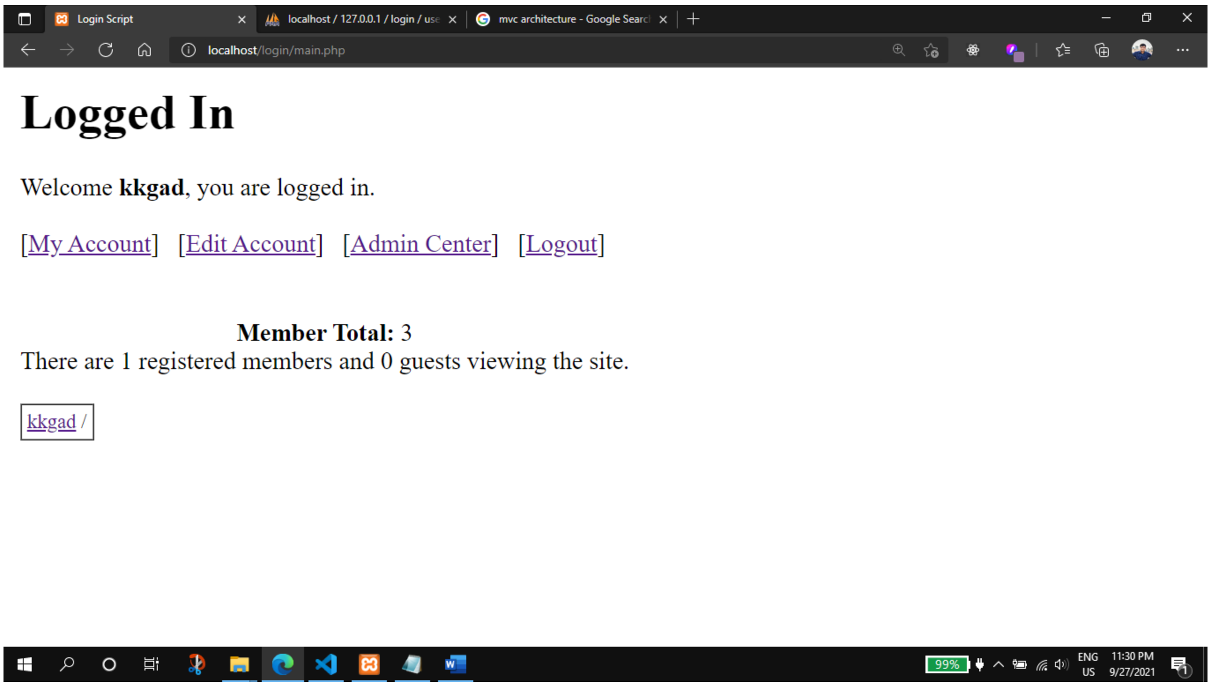
Username: kkgad
Password: ••••••
Remember me next time Login

[Forgot Password?]

Not registered? Sign-Up!

Member Total: 3
There are 0 registered members and 1 guests viewing the site.

99%
ENG US
11:28 PM
9/27/2021



localhost/login/main.php

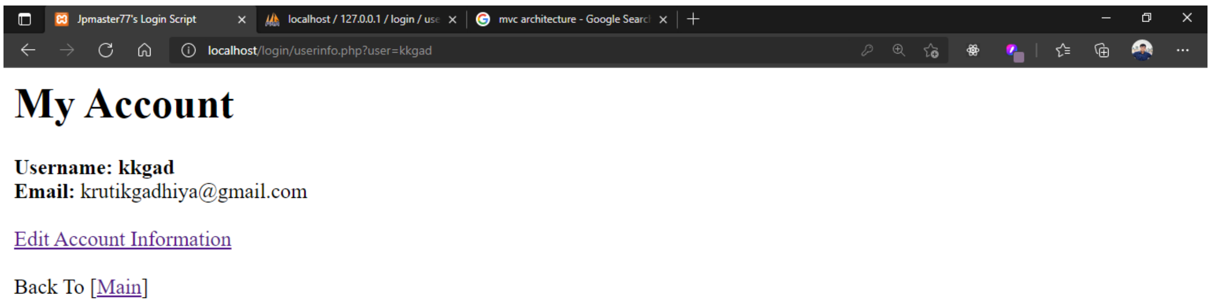
Logged In

Welcome **kkgad**, you are logged in.

[My Account] [Edit Account] [Admin Center] [Logout]

Member Total: 3
There are 1 registered members and 0 guests viewing the site.

kkgad /



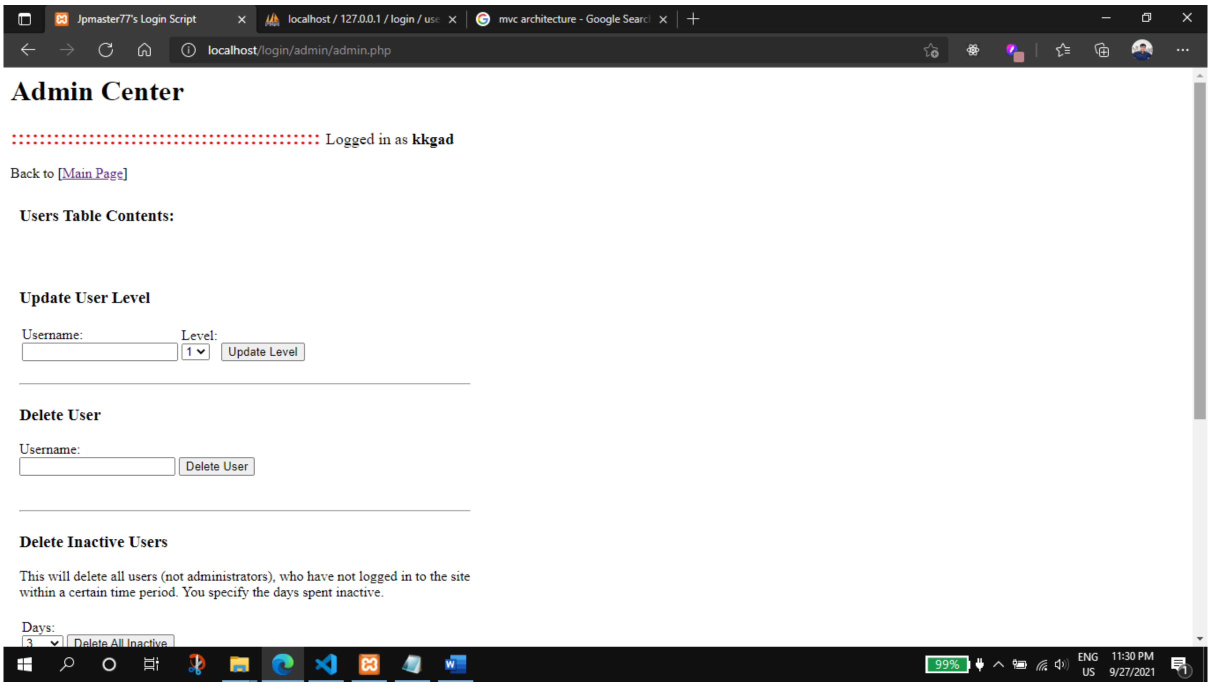
localhost/login/userinfo.php?user=kkgad

My Account

Username: kkgad
Email: krutikgadhiya@gmail.com

Edit Account Information

Back To [Main]



localhost/login/admin/admin.php

Admin Center

:: Logged in as **kkgad**

Back to [Main Page]

Users Table Contents:

Update User Level

Username: Level: 1 [Update Level]

Delete User

Username: [Delete User]

Delete Inactive Users

This will delete all users (not administrators), who have not logged in to the site within a certain time period. You specify the days spent inactive.

Days: 3 [Delete All Inactive]