### **Custom User Registration with Role Assignment in WordPress**

#### **Overview**

This code adds custom user roles and a frontend user registration form to WordPress. It enables users to register as either a **Teacher** or a **Student** and assigns them the corresponding role upon successful registration. The code is divided into three major parts:

1. Registering custom roles.
2. Displaying the registration form via a shortcode.
3. Processing the form data and assigning roles.

### **1. Register Custom Roles**

The function turing\_register\_custom\_roles() is used to create custom roles in WordPress. These roles are:

* **Teacher**
* **Student**

#### **How It Works:**

* It checks whether the roles have already been set by looking for an option turing\_roles\_are\_set.
* If not set, it creates the roles with specific capabilities (e.g., read, edit\_posts, etc.).
* After the roles are created, the option turing\_roles\_are\_set is updated to prevent re-adding roles on every page load.

#### **Code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// register custom roles

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function turing\_register\_custom\_roles()

{

// Check if roles are already registered before adding

if (!get\_option('turing\_roles\_are\_set')) {

add\_role('teacher', 'Teacher', array(

'read' => true,

'edit\_posts' => true,

'delete\_posts' => false,

'upload\_files' => true,

));

add\_role('student', 'Student', array(

'read' => true,

'edit\_posts' => false,

'delete\_posts' => false,

'upload\_files' => false,

));

// Mark that roles are set

update\_option('turing\_roles\_are\_set', true);

}

}

add\_action('after\_setup\_theme', 'turing\_register\_custom\_roles');

#### **Role Capabilities:**

* **Teacher**: Can read posts, edit them, and upload files, but cannot delete posts.
* **Student**: Can only read posts, cannot edit, delete, or upload files.

### **2. Frontend User Registration Form (Shortcode)**

The function turing\_frontend\_user\_registration\_form() generates a user registration form, which is displayed by using the shortcode [user\_registration\_form].

#### **Form Fields:**

* **Username**
* **Email**
* **Password**
* **Role (Teacher/Student)**

This form submits data via POST and invokes the turing\_register\_user\_and\_assign\_role() function to process the data.

#### **Code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// User Register Form Shortcode

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function turing\_frontend\_user\_registration\_form()

{

ob\_start();

?>

<form action="" method="POST">

<label for="username">Username</label>

<input type="text" name="username" required />

<label for="email">Email</label>

<input type="email" name="email" required />

<label for="password">Password</label>

<input type="password" name="password" required />

<label for="role">Role</label>

<select name="role" id="role" required>

<option value="">Select Role</option>

<option value="teacher">Teacher</option>

<option value="student">Student</option>

</select>

<input type="submit" name="submit\_user\_registration" value="Register" />

</form>

<?php

// Check if the form was submitted and process it

if (isset($\_POST['submit\_user\_registration'])) {

turing\_register\_user\_and\_assign\_role();

}

return ob\_get\_clean();

}

add\_shortcode('user\_registration\_form', 'turing\_frontend\_user\_registration\_form');

#### **Shortcode Usage:**

* You can use the shortcode [user\_registration\_form] in any page or post to display the registration form.

### **3. Processing the User Registration and Role Assignment**

The function turing\_register\_user\_and\_assign\_role() handles the logic for:

* Validating user input.
* Checking if the username or email already exists.
* Registering the user and assigning the selected role.

#### **How It Works:**

* **Input Validation:** It checks for valid username, email, password, and role.
* **Existing User Check:** It prevents registration if the username or email already exists in the database.
* **Role Validation:** Only the **Teacher** or **Student** roles are allowed.
* **User Registration:** If all checks pass, a new user is created, and the selected role is assigned.

#### **Code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// register\_user\_and\_assign\_role

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function turing\_register\_user\_and\_assign\_role()

{

$username = sanitize\_text\_field($\_POST['username']);

$email = sanitize\_email($\_POST['email']);

$password = sanitize\_text\_field($\_POST['password']);

$role = sanitize\_text\_field($\_POST['role']);

// Check if the user already exists

if (username\_exists($username) || email\_exists($email)) {

echo 'Username or email already exists.';

return;

}

// Check if the role is valid and exists

$valid\_roles = ['teacher', 'student']; // Only allow specific roles

if (!in\_array($role, $valid\_roles)) {

echo 'Invalid role selection.';

return;

}

// Register the new user

$user\_id = wp\_create\_user($username, $password, $email);

// If user creation was successful

if (!is\_wp\_error($user\_id)) {

// Assign the custom role to the new user

$user = new WP\_User($user\_id);

$user->set\_role($role); // Assign the selected role

echo 'User registered successfully and role assigned!';

} else {

// Error in user registration, output the specific error message

echo 'Error in user registration: ' . $user\_id->get\_error\_message();

}

}

#### **Form Submission & Output:**

* If registration is successful, the user is shown a success message: "User registered successfully and role assigned!".
* If there's an error (e.g., username already exists), the user is shown an error message.

### **Summary:**

* **Custom Roles:** Teacher and Student are created with specific capabilities.
* **Frontend Form:** A simple registration form is created using the shortcode [user\_registration\_form].
* **User Registration:** After form submission, the user is created, and the selected role is assigned.
* **Error Handling:** It checks for existing usernames, emails, and invalid roles before registering the user.