**Step 12: Implement Validation in Our Blog**

In this section we’re going to add validation error messages when user fails to fill the creation and edit form correctly. We can do this using frontend languages like Javascript but lets use PHP for our case. Alright let’s start.

**Update the AdminController.php**

In the store method of our AdminController add if condtion to check whether the form fields *(****$title***, ***$content*** and ***$category\_id*** of the create form are not empty. Here is the method.

// Create: Store a new post

    public function store($postData) {

        // Retrieve form data

        $title = $postData['title'];

        $content = $postData['content'];

        $category\_id = $postData['category\_id'];

        $imageFilename = null;  // Default to null if no image is uploaded

        // Validate title, content, and category

        $errors = [];

        if (empty($title)) {

            $errors[] = 'Title is required.';

        }

        if (empty($content)) {

            $errors[] = 'Content is required.';

        }

        if (empty($category\_id)) {

            $errors[] = 'Category is required.';

        }

        // Only proceed if there are no errors

        if (empty($errors)) {

        // Check if an image file is uploaded and there are no upload errors

        if (isset($\_FILES['image']) && $\_FILES['image']['error'] === UPLOAD\_ERR\_OK) {

            // Define allowed file types and size

            $allowedTypes = ['image/jpeg', 'image/png', 'image/gif'];

            $maxSize = 5 \* 1024 \* 1024; // 5 MB

            // Extract file information from the uploaded image

            $fileTmpPath = $\_FILES['image']['tmp\_name']; // Temporary path where the uploaded file is stored

            $fileName = $\_FILES['image']['name'];       // Original name of the uploaded file

            $fileType = $\_FILES['image']['type'];       // MIME type of the uploaded file

            $fileSize = $\_FILES['image']['size'];       // Size of the uploaded file in bytes

            // Check file type and size

            if (in\_array($fileType, $allowedTypes) && $fileSize <= $maxSize) {

                // Create a unique file name to avoid overwriting existing files

                $newFileName = uniqid() . '-' . $fileName;

                // Set the upload path for the image

                $uploadPath = 'S:/xampp/htdocs/php8blogapp/public/images/'; // Define the directory where uploaded images will be stored

                $destination = $uploadPath . $newFileName; // Concatenate the upload path with the new file name to create the full destination path

                // Ensure the upload directory exists

                if (!file\_exists($uploadPath)) {

                    // Create the directory if it does not exist, with full read/write/execute permissions (0777) and enable recursion (true)

                    mkdir($uploadPath, 0777, true);

                }

                // Move the uploaded file

                if (move\_uploaded\_file($fileTmpPath, $destination)) {

                    // File uploaded successfully

                    $imageFilename = $newFileName; // Set this for database storage

                } else {

                    // Handle error (file move failed)

                    $errors[] = 'Failed to upload image.';

                }

            } else {

                // Handle invalid file type or size

                $errors[] = 'Invalid file type or size.';

            }

        } else {

            // Handle no file uploaded or upload error

            if ($\_FILES['image']['error'] != UPLOAD\_ERR\_NO\_FILE) {

                $errors[] = 'Error in file upload.';

            }

        }

    }

    // Check again for any errors including file upload errors

    if (empty($errors)) {

        // Create the post in the database

        try {

            $this->model->createPost($title, $content, $category\_id, $imageFilename);

            // Redirect to the posts page after successful creation

            header('Location: /admin/posts');

            $\_SESSION['message'] = 'Post created successfully!.';

            exit;

        } catch (Exception $e) {

            // If there's a database error, display it

            echo 'Error creating post: ' . $e->getMessage();

            exit;

        }

        } else {

            // Handle validation errors

            $\_SESSION['form\_errors'] = $errors;

            header('Location: /admin/posts/create');

            exit;

        }

    }

**Update The View File**: ***admin/posts/create.php***

What we are adding here is the code for rendering validation error messages.

<!-- admin/posts/create.php -->

<div>

    <!-- View all Post Button -->

    <div style="text-align: right; margin-bottom: 20px;">

        <a href="/admin/posts/">View Posts</a>

    </div>

    <h1>Create Post</h1>

        <!-- Validation error messages -->

        <?php if (isset($\_SESSION['form\_errors'])): ?>

            <div class="alert alert-danger">

                <?php foreach ($\_SESSION['form\_errors'] as $error): ?>

                    <p><?php echo $error; ?></p>

                <?php endforeach; ?>

                <?php unset($\_SESSION['form\_errors']); ?>

            </div>

        <?php endif; ?>

    <form method="POST" action="/admin/posts/store" enctype="multipart/form-data">

        <div>

            <label for="title">Title:</label>

            <input type="text" id="title" name="title">

        </div>

        <div>

            <label for="content">Content:</label>

            <textarea id="content" name="content"></textarea>

        </div>

        <div>

            <label for="image">Image:</label>

            <input type="file" id="image" name="image">

        </div>

        <!-- Populate Categories in this view -->

        <div>

            <label for="category\_id">Category:</label>

            <select id="category\_id" name="category\_id">

                <?php foreach ($categories as $category): ?>

                    <option value="<?php echo $category->id; ?>"><?php echo $category->name; ?></option>

                <?php endforeach; ?>

            </select>

        </div>

        <button type="submit">Create Post</button>

    </form>

</div>

**Add Validation to Update Method**

**Modify the update method in the admincontroller**

We can also add validation to our update method. This is when a user is editing a post. In case the form fields are empty we display error messages like what we did above. Let’s go ahead and do this.

// Update: Update an existing post

    public function update($postData, $id) {

        $title = $postData['title'];

        $content = $postData['content'];

        $category\_id = $postData['category\_id'];

        // Initialize the image filename from existing data

        $imageFilename = $postData['existing\_image\_filename'] ?? null;

         // Validation

         $errors = [];

         if (empty($title)) {

             $errors[] = 'Title is required.';

         }

         if (empty($content)) {

             $errors[] = 'Content is required.';

         }

         if (empty($category\_id)) {

             $errors[] = 'Category is required.';

         }

         // Only proceed if there are no errors

         if (empty($errors)) {

            // Check if an image file is uploaded and there are no upload errors

            if (isset($\_FILES['image']) && $\_FILES['image']['error'] === UPLOAD\_ERR\_OK) {

                // Define allowed file types and size

                $allowedTypes = ['image/jpeg', 'image/png', 'image/gif'];

                $maxSize = 5 \* 1024 \* 1024; // 5 MB

                // Extract file information from the uploaded image

                $fileTmpPath = $\_FILES['image']['tmp\_name']; // Temporary path where the uploaded file is stored

                $fileName = $\_FILES['image']['name'];       // Original name of the uploaded file

                $fileType = $\_FILES['image']['type'];       // MIME type of the uploaded file

                $fileSize = $\_FILES['image']['size'];       // Size of the uploaded file in bytes

                // Check file type and size

                if (in\_array($fileType, $allowedTypes) && $fileSize <= $maxSize) {

                    // Create a unique file name to avoid overwriting existing files

                    $newFileName = uniqid() . '-' . $fileName;

                    // Set the upload path

                    $uploadPath = 'S:/xampp/htdocs/php8blogapp/public/images/';

                    $destination = $uploadPath . $newFileName;

                    // Ensure the upload directory exists

                    if (!file\_exists($uploadPath)) {

                        mkdir($uploadPath, 0777, true);

                    }

                    // Move the uploaded file

                    if (move\_uploaded\_file($fileTmpPath, $destination)) {

                        // File uploaded successfully

                        $imageFilename = $newFileName; // Update $imageFilename with new filename for database storage

                    } else {

                        // Handle error (file move failed)

                        $errors[] = 'Failed to upload image.';

                    }

                } else {

                    // Handle invalid file type or size

                    $errors[] = 'Invalid file type or size.';

                }

            } elseif ($\_FILES['image']['error'] != UPLOAD\_ERR\_NO\_FILE) {

                $errors[] = 'Error in file upload.';

            }

         }

        // Check again for any errors including file upload errors

        if (empty($errors)) {

            try {

                // Call the updatePost method

                $this->model->updatePost($id, $title, $content, $category\_id, $imageFilename);

 // Redirect to the posts page after successful update

             $\_SESSION['message'] = 'Post updated successfully!.';

              header('Location: /admin/posts');

              exit;

            } catch (Exception $e) {

                // If there's a database error, display it

                echo 'Error updating post: ' . $e->getMessage();

                exit;

            }

            } else {

                // Handle validation errors

                $\_SESSION['form\_errors'] = $errors;

                header('Location: /admin/posts/edit/' . $id);

                exit;

            }

        }

**Update The View File: (admin/posts/edit)**

<div>

    <?php if ($post === null): ?>

        <div>Post not found</div>

        <?php return; ?>

        <?php endif; ?>

        <!-- Validation error messages -->

        <?php if (isset($\_SESSION['form\_errors'])): ?>

        <div class="alert alert-danger">

            <?php foreach ($\_SESSION['form\_errors'] as $error): ?>

                <p><?php echo $error; ?></p>

            <?php endforeach; ?>

            <?php unset($\_SESSION['form\_errors']); ?>

        </div>

       <?php endif; ?>

      <form method="POST" action="/admin/posts/update/<?php echo $post->id ?>" enctype="multipart/form-data" style="margin-bottom: 20px;">

        <input type="hidden" name="id" value="<?php echo $post->id ?>">

        <div style="margin-bottom: 20px;">

            <label for="title">Title</label>

            <input type="text" id="title" name="title" value="<?php echo htmlspecialchars($post->title, ENT\_QUOTES); ?>">

        </div>

        <div style="margin-bottom: 20px;">

            <label for="content">Content</label>

            <textarea id="content" name="content"><?php echo htmlspecialchars($post->content, ENT\_QUOTES); ?></textarea>

        </div>

        <div>

            <label for="image">Image</label>

            <input type="file" id="image" name="image">

            <?php if (!empty($post->image)): ?>

            <img src="/images/<?php echo htmlspecialchars($post->image, ENT\_QUOTES); ?>" alt="Post Image" style="max-width: 200px;">

            <?php endif; ?>

        </div>

        <!-- Populate Categories in this view -->

        <div>

            <label for="category\_id">Category:</label>

            <select id="category\_id" name="category\_id">

            <?php foreach ($categories as $category): ?>

            <option value="<?php echo $category->id; ?>" <?php echo $category->id == $post->category\_id ? 'selected' : ''; ?>>

            <?php echo $category->name; ?>

            </option>

            <?php endforeach; ?>

            </select>

        </div>

        <button type="submit">Update Post</button>

    </form>

</div>