```php
<?php
/**
 * @version    4.2.0
 * @package    JEM
 * @copyright  (C) 2013-2023 joomlaeventmanager.net
 * @copyright  (C) 2005-2009 Christoph Lukes
 * @license    https://www.gnu.org/licenses/gpl-3.0 GNU/GPL
 */

defined('_JEXEC') or die;

use Joomla\CMS\Factory;
use Joomla\CMS\MVC\Controller\BaseController;
use Joomla\CMS\Language\Text;

/**
 * Source controller class
 */
class JemControllerSource extends BaseController
{
	/**
	 * Constructor.
	 *
	 * @param  array  An optional associative array of configuration settings.
	 * @see    JController
	 */
	public function __construct($config = array())
	{
		parent::__construct($config);

		// Apply, Save & New, and Save As copy should be standard on forms.
		$this->registerTask('apply', 'save');
	}

	/**
	 * Method to check if you can add a new record.
	 *
	 * @return boolean
	 */
	protected function allowEdit()
	{
		return JemFactory::getUser()->authorise('core.edit', 'com_jem');
	}

	/**
	 * Method to check if you can save a new or existing record.
	 *
	 * @return boolean
	 */
	protected function allowSave()
	{
		return $this->allowEdit();
```

```
	}

	/**
	 * Method to get a model object, loading it if required.
	 *
	 * @param  string  The model name. Optional.
	 * @param  string  The class prefix. Optional.
	 * @param  array   Configuration array for model. Optional.
	 *
	 * @return object  The model.
	 */
	public function getModel($name = 'Source', $prefix = 'JemModel', $config = array())
	{
		$model = parent::getModel($name, $prefix, $config);
		return $model;
	}

	/**
	 * This controller does not have a display method. Redirect back to the list view of the component.
	 *
	 * @param  boolean  If true, the view output will be cached
	 * @param  array    An array of safe url parameters and their variable types, for valid values see {@link JFilterInput::clean()}.
	 *
	 * @return JController  This object to support chaining.
	 *
	 */
	public function display($cachable = false, $urlparams = array())
	{
		$this->setRedirect(JRoute::_('index.php?option=com_jem&view=cssmanager', false));
	}

	/**
	 * Method to edit an existing record.
	 *
	 * @return boolean  True on success.
	 */
	public function edit()
	{
		// Initialise variables.
		$app      = Factory::getApplication();
		$model    = $this->getModel();
		$recordId = $app->input->get('id', '');
		$context  = 'com_jem.edit.source';

		if (preg_match('#\.\.#', base64_decode($recordId))) {
			Factory::getApplication()->enqueueMessage(Text::_('COM_JEM_CSSMANAGER_ERROR_SOURCE_FILE_NOT_FOUND'), 'warning');
		}
```

```
		// Access check.
		if (!$this->allowEdit()) {
			Factory::getApplication()->enqueueMessage(Text::_('JLIB_APPLICATION_ERROR_EDIT_NOT_PERMITTED'), 'warning');
		}

		// Check-out succeeded, push the new record id into the session.
		$app->setUserState($context.'.id', $recordId);
		$app->setUserState($context.'.data', null);
		$this->setRedirect('index.php?option=com_jem&view=source&layout=edit');
		return true;
	}

	/**
	 * Method to cancel an edit
	 */
	public function cancel()
	{
		// Check for request forgeries.
		JSession::checkToken() or jexit(Text::_('JINVALID_TOKEN'));

		// Initialise variables.
		$app     = Factory::getApplication();
		$model   = $this->getModel();
		$context = 'com_jem.edit.source';

		// Clean the session data and redirect.
		$app->setUserState($context.'.id', null);
		$app->setUserState($context.'.data', null);
		$this->setRedirect(JRoute::_('index.php?option=com_jem&view=cssmanager', false));
	}

	/**
	 * Saves a template source file.
	 *
	 * @return boolean  True on success.
	 */
	public function save()
	{
		// Check for request forgeries.
		JSession::checkToken() or jexit(Text::_('JINVALID_TOKEN'));

		// Initialise variables.
		$app     = Factory::getApplication();
		$data    = $app->input->get('jform', array(), 'array');
		$context = 'com_jem.edit.source';
		$task    = $this->getTask();
		$model   = $this->getModel();
```

```
		$file    = $model->getState('filename');
		$custom  = stripos($file, 'custom#:');

		# custom file?
		if ($custom !== false) {
			$file = str_replace('custom#:', '', $file);
		}

		// Access check.
		if (!$this->allowSave()) {
			Factory::getApplication()-
>enqueueMessage(Text::_('JERROR_SAVE_NOT_PERMITTED'), 'warning');
		}

		// Match the stored id's with the submitted.
		if (empty($data['filename']) || ($data['filename'] != $file)) {
			throw new
Exception(Text::_('COM_JEM_CSSMANAGER_ERROR_SOURCE_ID_FILENAME_MISMAT
CH'), 500);
		}

		// Validate the posted data.
		$form = $model->getForm();
		if (!$form)
		{
			Factory::getApplication()->enqueueMessage($model->getError(), 'error');
			return false;
		}

		$data = $model->validate($form, $data);

		// Check for validation errors.
		if ($data === false)
		{
			// Get the validation messages.
			$errors = $model->getErrors();

			// Push up to three validation messages out to the user.
			for ($i = 0, $n = count($errors); $i < $n && $i < 3; $i++)
			{
				if ($errors[$i] instanceof Exception) {
					$app->enqueueMessage($errors[$i]->getMessage(), 'warning');
				}
				else {
					$app->enqueueMessage($errors[$i], 'warning');
				}
			}

			// Save the data in the session.
			$app->setUserState($context.'.data', $data);

			// Redirect back to the edit screen.
```

```
                        $this->setRedirect(JRoute::_('index.php?
option=com_jem&view=source&layout=edit', false));
                        return false;
                }

                // Attempt to save the data.
                if (!$model->save($data))
                {
                        // Save the data in the session.
                        $app->setUserState($context.'.data', $data);

                        // Redirect back to the edit screen.
                        $this->setMessage(Text::sprintf('JERROR_SAVE_FAILED', $model-
>getError()), 'warning');
                        $this->setRedirect(JRoute::_('index.php?
option=com_jem&view=source&layout=edit', false));
                        return false;
                }

                $this-
>setMessage(Text::_('COM_JEM_CSSMANAGER_FILE_SAVE_SUCCESS'));

                // Redirect the user and adjust session state based on the chosen task.
                switch ($task)
                {
                        case 'apply':
                                // Reset the record data in the session.
                                $app->setUserState($context.'.data', null);

                                // Redirect back to the edit screen.
                                $this->setRedirect(JRoute::_('index.php?
option=com_jem&view=source&layout=edit', false));
                                break;

                        default:
                                // Clear the record id and data from the session.
                                $app->setUserState($context.'.id', null);
                                $app->setUserState($context.'.data', null);

                                // Redirect to the list screen.
                                $this->setRedirect(JRoute::_('index.php?
option=com_jem&view=cssmanager', false));
                                break;
                }
        }
}
```