Codeinwp / visualizer Public

Code Issues 36 Pull requests 1 Actions Projects Security ...

master

visualizer / classes / Visualizer / Module / Chart.php / Jump to

girishpanchal30 Fix PHP error when polylang plugin activated #960 ✕ History

5 contributors

1597 lines (1434 sloc) | 56.1 KB

```php
<?php
// +----------------------------------------------------------------------+
// | Copyright 2013  Madpixels  (email : visualizer@madpixels.net)        |
// +----------------------------------------------------------------------+
// | This program is free software; you can redistribute it and/or modify |
// | it under the terms of the GNU General Public License, version 2, as  |
// | published by the Free Software Foundation.                           |
// |                                                                      |
// | This program is distributed in the hope that it will be useful,      |
// | but WITHOUT ANY WARRANTY; without even the implied warranty of       |
// | MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the        |
// | GNU General Public License for more details.                         |
// |                                                                      |
// | You should have received a copy of the GNU General Public License    |
// | along with this program; if not, write to the Free Software          |
// | Foundation, Inc., 51 Franklin St, Fifth Floor, Boston,               |
// | MA 02110-1301 USA                                                    |
// +----------------------------------------------------------------------+
// | Author: Eugene Manuilov <eugene@manuilov.org>                        |
// +----------------------------------------------------------------------+
/**
 * The module for all stuff related to getting, editing, creating and deleting charts.
 *
 * @category Visualizer
 * @package Module
 *

 * @since 1.0.0
 */
```

```php
class Visualizer_Module_Chart extends Visualizer_Module {

	const NAME = __CLASS__;

	/**
	 * The chart object.
	 *
	 * @since 1.0.0
	 *
	 * @access private
	 * @var WP_Post
	 */
	private $_chart;

	/**
	 * Constructor.
	 *
	 * @since 1.0.0
	 *
	 * @access public
	 *
	 * @param Visualizer_Plugin $plugin The instance of the plugin.
	 */
	public function __construct( Visualizer_Plugin $plugin ) {
		parent::__construct( $plugin );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_GET_CHARTS, 'getCharts' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_DELETE_CHART, 'deleteChart' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_CREATE_CHART, 'renderChartPages'
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_EDIT_CHART, 'renderChartPages' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_UPLOAD_DATA, 'uploadData' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_CLONE_CHART, 'cloneChart' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_EXPORT_DATA, 'exportData' );

		$this->_addAjaxAction( Visualizer_Plugin::ACTION_FETCH_DB_DATA, 'getQueryData' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_SAVE_DB_QUERY, 'saveQuery' );

		$this->_addAjaxAction( Visualizer_Plugin::ACTION_JSON_GET_ROOTS, 'getJsonRoots' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_JSON_GET_DATA, 'getJsonData' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_JSON_SET_DATA, 'setJsonData' );
		$this->_addAjaxAction( Visualizer_Plugin::ACTION_JSON_SET_SCHEDULE, 'setJsonSchedu

		$this->_addAjaxAction( Visualizer_Plugin::ACTION_SAVE_FILTER_QUERY, 'saveFilter' )

		$this->_addFilter( 'visualizer_get_sidebar', 'getSidebar', 10, 2 );

	}

	/**
	 * Generates the HTML of the sidebar for the chart.
	 *
	 * @since ?
	 *
	 * @access public
	 */
	public function getSidebar( $sidebar, $chart_id ) {
		$chart      = get_post( $chart_id );
		$data          = $this->_getChartArray( $chart );
		$sidebar       = '';
		$sidebar_class = $this->load_chart_class_name( $chart_id );
		if ( class_exists( $sidebar_class, true ) ) {
			$sidebar           = new $sidebar_class( $data['settings'] );
			$sidebar->__series = $data['series'];
			$sidebar->__data   = $data['data'];
		} else {
			$sidebar = apply_filters( 'visualizer_pro_chart_type_sidebar', '', $data )
			if ( $sidebar !== '' ) {
				$sidebar->__series = $data['series'];
				$sidebar->__data   = $data['data'];
			}
		}
		return str_replace( "'", '"', $sidebar->__toString() );
	}

	/**
	 * Sets the schedule for how JSON-endpoint charts should be updated.
	 *
	 * @since ?
	 *
	 * @access public
	 */
	public function setJsonSchedule() {
		check_ajax_referer( Visualizer_Plugin::ACTION_JSON_SET_SCHEDULE . Visualizer_Plugi

		$chart_id = filter_input(
			INPUT_POST,
			'chart',
			FILTER_VALIDATE_INT,
			array(
				'options' => array(
					'min_range' => 1,
				),
			)
		);

		if ( ! $chart_id ) {
			wp_send_json_error();
		}

```

```php
		$time = filter_input(
			INPUT_POST,
			'time',
			FILTER_VALIDATE_INT,
			array(
				'options' => array(
					'min_range' => -1,
				),
			)
		);

		if ( Visualizer_Module::is_pro() ) {
			$is_woocommerce_report = filter_input(
				INPUT_POST,
				'is_woocommerce_report',
				FILTER_VALIDATE_BOOLEAN
			);

			if ( $is_woocommerce_report ) {
				update_post_meta( $chart_id, Visualizer_Plugin::CF_IS_WOOCOMMERCE_
			} else {
				delete_post_meta( $chart_id, Visualizer_Plugin::CF_IS_WOOCOMMERCE_
			}
		}

		delete_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_SCHEDULE );

		if ( -1 < $time ) {
			add_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_SCHEDULE, $time );
			// Update schedules.
			$schedules              = get_option( Visualizer_Plugin::CF_JSON_SCHEDULE,
			$schedules[ $chart_id ] = time() + $time * HOUR_IN_SECONDS;
			update_option( Visualizer_Plugin::CF_JSON_SCHEDULE, $schedules );
		}
		wp_send_json_success();
	}

	/**
	 * Get the root elements for JSON-endpoint.
	 *
	 * @since ?
	 *
	 * @access public
	 */
	public function getJsonRoots() {
		check_ajax_referer( Visualizer_Plugin::ACTION_JSON_GET_ROOTS . Visualizer_Plugin::

		$params     = wp_parse_args( $_POST['params'] );

```

```
                $source = new Visualizer_Source_Json( $params );

                $roots = $source->fetchRoots();
                if ( empty( $roots ) ) {
                        wp_send_json_error( array( 'msg' => $source->get_error() ) );
                }

                wp_send_json_success( array( 'url' => $params['url'], 'roots' => $roots ) );
        }

        /**
         * Get the data for the JSON-endpoint corresponding to the chosen root.
         *
         * @since ?
         *
         * @access public
         */
        public function getJsonData() {
                check_ajax_referer( Visualizer_Plugin::ACTION_JSON_GET_DATA . Visualizer_Plugin::V

                $params = wp_parse_args( $_POST['params'] );

                $chart_id = $params['chart'];

                if ( empty( $chart_id ) ) {
                        wp_die();
                }

                $source = new Visualizer_Source_Json( $params );
                $source->fetch();
                $data   = $source->getRawData();

                if ( empty( $data ) ) {
                        wp_send_json_error( array( 'msg' => esc_html__( 'Unable to fetch data from

                }

                $data   = Visualizer_Render_Layout::show( 'editor-table', $data, $chart_id, 'viz-j
                wp_send_json_success( array( 'table' => $data, 'root' => $params['root'], 'url' =>
        }

        /**
         * Updates the database with the correct post parameters for JSON-endpoint charts.
         *
         * @since ?
         *
         * @access public
         */
        public function setJsonData() {
                check_ajax_referer( Visualizer_Plugin::ACTION_JSON_SET_DATA . Visualizer_Plugin::V
```

```php

		$params = $_POST;
		$chart_id = $_GET['chart'];

		if ( empty( $chart_id ) ) {
			wp_die();
		}

		$chart  = get_post( $chart_id );

		$source = new Visualizer_Source_Json( $params );
		update_post_meta( $chart->ID, Visualizer_Plugin::CF_EDITABLE_TABLE, true );
		$source->fetchFromEditableTable();

		$content    = $source->getData( get_post_meta( $chart->ID, Visualizer_Plugin::CF_E
		$chart->post_content = $content;
		wp_update_post( $chart->to_array() );
		update_post_meta( $chart->ID, Visualizer_Plugin::CF_SERIES, $source->getSeries() )
		update_post_meta( $chart->ID, Visualizer_Plugin::CF_SOURCE, $source->getSourceName
		update_post_meta( $chart->ID, Visualizer_Plugin::CF_DEFAULT_DATA, 0 );
		update_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_URL, $params['url'] );
		update_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_ROOT, $params['root'] );

		delete_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_HEADERS );
		$headers = array( 'method' => $params['method'] );
		if ( ! empty( $params['auth'] ) ) {
			$headers['auth'] = $params['auth'];
		} elseif ( ! empty( $params['username'] ) && ! empty( $params['password'] ) ) {
			$headers['auth'] = array( 'username' => $params['username'], 'password' =>
		}

		add_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_HEADERS, $headers );

		delete_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_PAGING );
		if ( ! empty( $params['paging'] ) ) {
			add_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_PAGING, $params['pag
		}

		if ( Visualizer_Module::is_pro() ) {
			if ( ! empty( $params['vz_woo_source'] ) ) {
				update_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_WOOCOMMER
			} else {
				delete_post_meta( $chart->ID, Visualizer_Plugin::CF_JSON_WOOCOMMER
			}
		}

		$time = filter_input(
			INPUT_POST,
			'time',
					FILTER_VALIDATE_INT,
					array(
						'options' => array(
							'min_range' => -1,
						),
					)
				);

				delete_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_SCHEDULE );

				if ( -1 < $time ) {
					add_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_SCHEDULE, $time );
				}

				// delete other source specific parameters.
				delete_post_meta( $chart_id, Visualizer_Plugin::CF_DB_QUERY );
				delete_post_meta( $chart_id, Visualizer_Plugin::CF_DB_SCHEDULE );
				delete_post_meta( $chart_id, Visualizer_Plugin::CF_CHART_URL );
				delete_post_meta( $chart_id, Visualizer_Plugin::CF_CHART_SCHEDULE );

				$render         = new Visualizer_Render_Page_Update();
				$render->id     = $chart->ID;
				$render->data   = json_encode( $source->getRawData( get_post_meta( $chart_id, Visu
				$render->series = json_encode( $source->getSeries() );
				$render->render();

				defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
			}


			/**
			 * Fetches charts from database.
			 *
			 * This method is also called from the media pop-up (classic editor: create a post and add
			 *
			 * @since 1.0.0
			 *
			 * @access public
			 */
			public function getCharts() {
				$query_args = array(
					'post_type'      => Visualizer_Plugin::CPT_VISUALIZER,
					'posts_per_page' => 9,
					'paged'          => filter_input(
						INPUT_GET,
						'page',
						FILTER_VALIDATE_INT,
						array(
							'options' => array(
```

```php
							'min_range' => 1,
							'default'   => 1,
						),
					)
				),
			);
			$filter     = filter_input( INPUT_GET, 's', FILTER_SANITIZE_STRING );
			if ( empty( $filter ) ) {
				// 'filter' is from the modal from the add media button.
				$filter = filter_input( INPUT_GET, 'filter', FILTER_SANITIZE_STRING );
			}

			if ( $filter && in_array( $filter, Visualizer_Plugin::getChartTypes(), true ) ) {
				$query_args['meta_query'] = array(
					array(
						'key'     => Visualizer_Plugin::CF_CHART_TYPE,
						'value'   => $filter,
						'compare' => '=',
					),
				);
			}
			$query  = new WP_Query( $query_args );
			$charts = array();
			while ( $query->have_posts() ) {
				$chart            = $query->next_post();
				$chart_data       = $this->_getChartArray( $chart );
				$chart_data['id'] = $chart->ID;
				$chart_data['library'] = $this->load_chart_type( $chart->ID );
				$css                = '';
				$settings           = $chart_data['settings'];
				$arguments          = $this->get_inline_custom_css( 'visualizer-chart-' .
				if ( ! empty( $arguments ) ) {
					$css        = $arguments[0];
					$settings   = $arguments[1];
				}
				$chart_data['settings'] = $settings;
				$chart_data['css'] = $css;
				$charts[]         = $chart_data;
			}
			self::_sendResponse(
				array(
					'success' => true,
					'data'    => $charts,
					'total'   => $query->max_num_pages,
				)
			);
		}

		/**
	 * Returns chart data required for rendering.
	 *
	 * @since 1.0.0
	 *
	 * @access private
	 *
	 * @param WP_Post $chart The chart object.
	 *
	 * @return array The array of chart data.
	 */
	private function _getChartArray( WP_Post $chart = null ) {
		if ( is_null( $chart ) ) {
			$chart = $this->_chart;
		}
		$type   = get_post_meta( $chart->ID, Visualizer_Plugin::CF_CHART_TYPE, true );
		$series = apply_filters( Visualizer_Plugin::FILTER_GET_CHART_SERIES, get_post_meta
		$data   = self::get_chart_data( $chart, $type );
		$library = $this->load_chart_type( $chart->ID );

		$settings = get_post_meta( $chart->ID, Visualizer_Plugin::CF_SETTINGS, true );
		$settings = apply_filters( Visualizer_Plugin::FILTER_GET_CHART_SETTINGS, $settings
		if ( ! empty( $atts['settings'] ) ) {
			$settings = apply_filters( $atts['settings'], $settings, $chart->ID, $type
		}

		$css        = '';
		$arguments  = $this->get_inline_custom_css( 'visualizer-' . $chart->ID, $settings
		if ( ! empty( $arguments ) ) {
			$css        = $arguments[0];
			$settings   = $arguments[1];
		}

		$date_formats = Visualizer_Source::get_date_formats_if_exists( $series, $data );

		return array(
			'type'     => $type,
			'series'   => $series,
			'settings' => $settings,
			'data'     => $data,
			'library'  => $library,
			'css'       => $css,
			'date_formats'       => $date_formats,
		);
	}

	/**
	 * Sends json response.
	 *
	 * @since 1.0.0
```

```php
	 *
	 * @access private
	 *
	 * @param array $results The response array.
	 */
	public static function _sendResponse( $results ) {
		header( 'Content-type: application/json' );
		nocache_headers();
		echo json_encode( $results );
		defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
	}

	/**
	 * Deletes a chart from database.
	 *
	 * @since 1.0.0
	 * @uses wp_delete_post() To delete a chart.
	 *
	 * @access public
	 */
	public function deleteChart() {
		$is_post      = $_SERVER['REQUEST_METHOD'] === 'POST';
		$input_method = $is_post ? INPUT_POST : INPUT_GET;
		$chart_id     = $success = false;
		$nonce        = wp_verify_nonce( filter_input( $input_method, 'nonce' ) );
		$capable      = current_user_can( 'delete_posts' );
		if ( $nonce && $capable ) {
			$chart_id = filter_input(
				$input_method,
				'chart',
				FILTER_VALIDATE_INT,
				array(
					'options' => array(
						'min_range' => 1,
					),
				)
			);
			if ( $chart_id ) {
				$chart   = get_post( $chart_id );
				$success = $chart && $chart->post_type === Visualizer_Plugin::CPT_
			}
		}
		if ( $success ) {
			global $sitepress;
			if ( Visualizer_Module::is_pro() && ( function_exists( 'icl_get_languages'
				$trid         = $sitepress->get_element_trid( $chart_id, 'post_' .
				$translations = $sitepress->get_element_translations( $trid );
				if ( ! empty( $translations ) ) {
					foreach ( $translations as $translated_post ) {
```

```
                                                wp_delete_post( $translated_post->element_id, true
                                        }
                                } else {
                                        wp_delete_post( $chart_id, true );
                                }
                        } else {
                                wp_delete_post( $chart_id, true );
                        }
                }
                if ( $is_post ) {
                        self::_sendResponse(
                                array(
                                        'success' => $success,
                                )
                        );
                }
                wp_redirect( remove_query_arg( 'vaction', wp_get_referer() ) );
                exit;
        }

        /**
         * Delete charts that are still in auto-draft mode.
         */
        private function deleteOldCharts() {
                $query = new WP_Query(
                        array(
                                'post_type'    => Visualizer_Plugin::CPT_VISUALIZER,
                                'post_status'  => 'auto-draft',
                                'fields'                => 'ids',
                                'update_post_meta_cache' => false,
                                'update_post_term_cache' => false,
                                'posts_per_page'        => 50,
                                'date_query' => array(
                                        array(
                                                'before' => 'today',
                                        ),
                                ),
                        )
                );

                if ( $query->have_posts() ) {
                        $ids = array();
                        while ( $query->have_posts() ) {
                                wp_delete_post( $query->next_post(), true );
                        }
                }
        }

        /**
```

```
	 * Renders appropriate page for chart builder. Creates new auto draft chart
	 * if no chart has been specified.
	 *
	 * @since 1.0.0
	 *
	 * @access public
	 */
	public function renderChartPages() {
		defined( 'IFRAME_REQUEST' ) || define( 'IFRAME_REQUEST', 1 );
		if ( ! defined( 'ET_BUILDER_PRODUCT_VERSION' ) && function_exists( 'et_get_theme_v
			define( 'ET_BUILDER_PRODUCT_VERSION', et_get_theme_version() );
		}
		// Set current screen for the render chart.
		set_current_screen( 'visualizer_render_chart' );
		// check chart, if chart not exists, will create new one and redirects to the same
		$chart_id = isset( $_GET['chart'] ) ? filter_var( $_GET['chart'], FILTER_VALIDATE_
		if ( ! $chart_id || ! ( $chart = get_post( $chart_id ) ) || $chart->post_type !== '
			if ( empty( $_GET['lang'] ) || empty( $_GET['parent_chart_id'] ) ) {
				$this->deleteOldCharts();
				$default_type = isset( $_GET['type'] ) && ! empty( $_GET['type'] )
				$source       = new Visualizer_Source_Csv( VISUALIZER_ABSPATH . DI
				$source->fetch();
				$chart_id = wp_insert_post(
					array(
						'post_type'    => Visualizer_Plugin::CPT_VISUALIZE
						'post_title'   => 'Visualization',
						'post_author'  => get_current_user_id(),
						'post_status'  => 'auto-draft',
						'post_content' => $source->getData( get_post_meta(
					)
				);
				if ( $chart_id && ! is_wp_error( $chart_id ) ) {
					add_post_meta( $chart_id, Visualizer_Plugin::CF_CHART_TYPE
					add_post_meta( $chart_id, Visualizer_Plugin::CF_DEFAULT_DA
					add_post_meta( $chart_id, Visualizer_Plugin::CF_SOURCE, $s
					add_post_meta( $chart_id, Visualizer_Plugin::CF_SERIES, $s
					add_post_meta( $chart_id, Visualizer_Plugin::CF_CHART_LIBR
					add_post_meta(
						$chart_id,
						Visualizer_Plugin::CF_SETTINGS,
						array(
							'focusTarget' => 'datum',
						)
					);

					do_action( 'visualizer_pro_new_chart_defaults', $chart_id
				}
			} else {
				if ( current_user_can( 'edit_posts' ) ) {
```

```
                    $parent_chart_id = isset( $_GET['parent_chart_id'] ) ? fil
                    $success = false;
                    if ( $parent_chart_id ) {
                        $parent_chart   = get_post( $parent_chart_id );
                        $success = $parent_chart && $parent_chart->post_ty
                    }
                    if ( $success ) {
                        $new_chart_id = wp_insert_post(
                            array(
                                'post_type'    => Visualizer_Plugi
                                'post_title'   => 'Visualization',
                                'post_author'  => get_current_user
                                'post_status'  => $parent_chart->p
                                'post_content' => $parent_chart->p
                            )
                        );

                        if ( is_wp_error( $new_chart_id ) ) {
                            do_action( 'themeisle_log_event', Visualiz
                        } else {
                            $post_meta = get_post_meta( $parent_chart_
                            $chart_id  = $new_chart_id;
                            foreach ( $post_meta as $key => $value ) {
                                if ( strpos( $key, 'visualizer-' ) 
                                    add_post_meta( $new_chart_
                                }
                            }
                        }
                    }
                }
                do_action( 'visualizer_pro_new_chart_defaults', $chart_id );
            }
            wp_redirect( esc_url_raw( add_query_arg( 'chart', (int) $chart_id ) ) );

            if ( defined( 'WP_TESTS_DOMAIN' ) ) {
                wp_die();
            }
            exit();
        }

        $_POST['save_chart_image'] = isset( $_POST['save_chart_image'] ) && 'yes' === $_PO
        $_POST['lazy_load_chart']  = isset( $_POST['lazy_load_chart'] ) && 'yes' === $_POS

        if ( isset( $_POST['chart-img'] ) && ! empty( $_POST['chart-img'] ) ) {
            $attachment_id = $this->save_chart_image( $_POST['chart-img'], $chart_id,
            if ( $attachment_id ) {
                update_post_meta( $chart_id, Visualizer_Plugin::CF_CHART_IMAGE, $a
            }
        }
```

```php
		$lib = $this->load_chart_type( $chart_id );

		// the alpha color picker (RGBA) is not supported by google.
		$color_picker_dep = 'wp-color-picker';
		if ( in_array( $lib, array( 'chartjs', 'datatables' ), true ) && ! wp_script_is( '
			wp_register_script( 'wp-color-picker-alpha', VISUALIZER_ABSURL . 'js/lib/w
			$color_picker_dep = 'wp-color-picker-alpha';
		}

		// enqueue and register scripts and styles
		wp_register_script( 'visualizer-chosen', VISUALIZER_ABSURL . 'js/lib/chosen.jquery
		wp_register_style( 'visualizer-chosen', VISUALIZER_ABSURL . 'css/lib/chosen.min.cs

		wp_register_style( 'visualizer-frame', VISUALIZER_ABSURL . 'css/frame.css', array(
		wp_register_script( 'visualizer-frame', VISUALIZER_ABSURL . 'js/frame.js', array(
		wp_register_script( 'visualizer-customization', $this->get_user_customization_js()
		wp_register_script(
			'visualizer-render',
			VISUALIZER_ABSURL . 'js/render-facade.js',
			apply_filters( 'visualizer_assets_render', array( 'visualizer-frame', 'vis
			Visualizer_Plugin::VERSION,
			true
		);
		wp_register_script(
			'visualizer-preview',
			VISUALIZER_ABSURL . 'js/preview.js',
			array(
				$color_picker_dep,
				'visualizer-render',
			),
			Visualizer_Plugin::VERSION,
			true
		);
		wp_register_script( 'visualizer-editor-simple', VISUALIZER_ABSURL . 'js/simple-edi

		do_action( 'visualizer_add_scripts' );

		if ( Visualizer_Module::is_pro() && Visualizer_Module::is_pro_older_than( '1.9.0'
			global $Visualizer_Pro;
			$Visualizer_Pro->_addScriptsAndStyles();
		}

		// dispatch pages
		$this->_chart = get_post( $chart_id );
		$tab    = isset( $_GET['tab'] ) && ! empty( $_GET['tab'] ) ? $_GET['tab'] : 'visua

		// skip chart type pages only for existing charts.
		if ( VISUALIZER_SKIP_CHART_TYPE_PAGE && 'auto-draft' !== $this->_chart->post_statu
			$tab = 'settings';
			}

			if ( isset( $_POST['cancel'] ) && 1 === intval( $_POST['cancel'] ) ) {
				// if the cancel button is clicked.
				$this->undoRevisions( $chart_id, true );
			} elseif ( isset( $_POST['save'] ) && 1 === intval( $_POST['save'] ) ) {
				$this->handlePermissions();
				// if the save button is clicked.
				$this->undoRevisions( $chart_id, false );
			} else {
				// if the edit button is clicked.
				$this->_chart = $this->handleExistingRevisions( $chart_id, $this->_chart )
			}

			// Clear existing chart cache.
			if ( isset( $_POST['save'] ) && 1 === intval( $_POST['save'] ) ) {
				$cache_key = Visualizer_Plugin::CF_CHART_CACHE . '_' . $chart_id;
				if ( get_transient( $cache_key ) ) {
					delete_transient( $cache_key );
				}
			}

			switch ( $tab ) {
				case 'settings':
					$this->_handleDataAndSettingsPage();
					break;
				case 'type': // fall through.
				case 'visualizer': // fall through.
					$this->_handleTypesPage();
					break;
				default:
					// this should never happen.
					break;
			}
			defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
		}

		/**
		 * Load code editor assets.
		 */
		private function loadCodeEditorAssets( $chart_id ) {
			global $wp_version;

			$wp_scripts = wp_scripts();

			// data tables assets.
			wp_register_script( 'visualizer-datatables', VISUALIZER_ABSURL . 'js/lib/datatable
			wp_register_style( 'visualizer-datatables', VISUALIZER_ABSURL . 'css/lib/datatable
			wp_register_style( 'visualizer-jquery-ui', sprintf( '//code.jquery.com/ui/%s/theme
			wp_enqueue_script( 'visualizer-datatables' );
			wp_enqueue_style( 'visualizer-jquery-ui' );

			if ( ! Visualizer_Module::is_pro() ) {
				return;
			}

			$table_col_mapping  = Visualizer_Source_Query_Params::get_all_db_tables_column_map

			if ( version_compare( $wp_version, '4.9.0', '<' ) ) {
				// code mirror assets.
				wp_register_script( 'visualizer-codemirror-core', '//codemirror.net/lib/co
				wp_register_script( 'visualizer-codemirror-placeholder', '//codemirror.net
				wp_register_script( 'visualizer-codemirror-matchbrackets', '//codemirror.n
				wp_register_script( 'visualizer-codemirror-closebrackets', '//codemirror.n
				wp_register_script( 'visualizer-codemirror-sql', '//codemirror.net/mode/sq
				wp_register_script( 'visualizer-codemirror-sql-hint', '//codemirror.net/ad
				wp_register_script( 'visualizer-codemirror-hint', '//codemirror.net/addon/
				wp_register_style( 'visualizer-codemirror-core', '//codemirror.net/lib/cod
				wp_register_style( 'visualizer-codemirror-hint', '//codemirror.net/addon/h

				wp_enqueue_script( 'visualizer-codemirror-hint' );
				wp_enqueue_style( 'visualizer-codemirror-hint' );
			} else {
				wp_enqueue_code_editor(
					array(
						'type' => 'sql',
						'codemirror' => array(
							'autofocus'         => true,
							'lineWrapping'      => true,
							'dragDrop'          => false,
							'matchBrackets'     => true,
							'autoCloseBrackets' => true,
							'extraKeys'         => array( 'Ctrl-Space' => 'aut
							'hintOptions'       => array( 'tables' => $table_c
						),
					)
				);
			}

			return $table_col_mapping;
		}

		/**
		 * Handle permissions from the new conslidated settings sidebar.
		 */
		private function handlePermissions() {
			if ( ! Visualizer_Module::is_pro() ) {
				return;
		}

		// we will not support old free and new pro.
		// handling new free and old pro.
		if ( has_action( 'visualizer_pro_handle_tab' ) ) {
			do_action( 'visualizer_pro_handle_tab', 'permissions', $this->_chart );
		} else {
			do_action( 'visualizer_handle_permissions', $this->_chart );
		}
	}

	/**
	 * Handle data and settings page
	 */
	private function _handleDataAndSettingsPage() {
		if ( isset( $_POST['map_api_key'] ) ) {
			update_option( 'visualizer-map-api-key', $_POST['map_api_key'] );
		}

		if ( $_SERVER['REQUEST_METHOD'] === 'POST' && isset( $_GET['nonce'] ) && wp_verify
			if ( $this->_chart->post_status === 'auto-draft' ) {
				$this->_chart->post_status = 'publish';

				// ensure that a revision is not created. If a revision is created
				// we do not want any difference in data so disable revisions temp
				$this->disableRevisionsTemporarily();

				wp_update_post( $this->_chart->to_array() );
			}
			// save meta data only when it is NOT being canceled.
			if ( ! ( isset( $_POST['cancel'] ) && 1 === intval( $_POST['cancel'] ) ) )
				update_post_meta( $this->_chart->ID, Visualizer_Plugin::CF_SETTING

				// we will keep a parameter called 'internal_title' that will be s
				// this will help in searching with the chart id.
				$settings = get_post_meta( $this->_chart->ID, Visualizer_Plugin::C
				$title = null;
				if ( isset( $settings['title'] ) && ! empty( $settings['title'] )
					$title  = $settings['title'];
					if ( is_array( $title ) ) {
						$title  = $settings['title']['text'];
					}
				}
				if ( empty( $title ) ) {
					$title  = $this->_chart->ID;
				}
				$settings['internal_title'] = $title;
				$settings_label = isset( $settings['pieResidueSliceLabel'] ) ? $se
				if ( empty( $settings_label ) ) {
```

```php
					$settings['pieResidueSliceLabel'] = esc_html__( 'Other', '
				} else {
					$settings['pieResidueSliceLabel'] = $settings_label;
				}
				update_post_meta( $this->_chart->ID, Visualizer_Plugin::CF_SETTING
			}
			$render       = new Visualizer_Render_Page_Send();
			$render->text = sprintf( '[visualizer id="%d"]', $this->_chart->ID );
			wp_iframe( array( $render, 'render' ) );
			return;
		}
		$data          = $this->_getChartArray();
		$sidebar       = '';
		$sidebar_class = $this->load_chart_class_name( $this->_chart->ID );
		if ( class_exists( $sidebar_class, true ) ) {
			$sidebar           = new $sidebar_class( $data['settings'] );
			$sidebar->__series = $data['series'];
			$sidebar->__data   = $data['data'];
		} else {
			$sidebar = apply_filters( 'visualizer_pro_chart_type_sidebar', '', $data )
			if ( $sidebar !== '' ) {
				$sidebar->__series = $data['series'];
				$sidebar->__data   = $data['data'];
			}
		}
		unset( $data['settings']['width'], $data['settings']['height'], $data['settings'][
		wp_enqueue_style( 'wp-color-picker' );
		wp_enqueue_style( 'visualizer-frame' );
		wp_enqueue_script( 'visualizer-preview' );
		wp_enqueue_script( 'visualizer-chosen' );
		wp_enqueue_script( 'visualizer-render' );

		if ( Visualizer_Module::can_show_feature( 'simple-editor' ) ) {
			wp_enqueue_script( 'visualizer-editor-simple' );
			wp_localize_script(
				'visualizer-editor-simple',
				'visualizer1',
				array(
					'ajax'      => array(
						'url'     => admin_url( 'admin-ajax.php' ),
						'nonces'  => array(
						),
						'actions' => array(
						),
					),
				)
			);
		}

```

```
                $table_col_mapping  = $this->loadCodeEditorAssets( $this->_chart->ID );

                wp_localize_script(
                        'visualizer-render',
                        'visualizer',
                        array(
                                'l10n'   => array(
                                        'invalid_source' => esc_html__( 'You have entered an inval
                                        'loading'       => esc_html__( 'Loading...', 'visualizer'
                                        'json_error'    => esc_html__( 'An error occured in fetchi
                                        'select_columns'    => esc_html__( 'Please select a few co
                                        'save_settings'    => __( 'You have modified the chart\'s
                                ),
                                'charts' => array(
                                        'canvas' => $data,
                                        'id' => $this->_chart->ID,
                                ),
                                'language'  => $this->get_language(),
                                'map_api_key' => get_option( 'visualizer-map-api-key' ),
                                'ajax'      => array(
                                        'url'     => admin_url( 'admin-ajax.php' ),
                                        'nonces'  => array(
                                                'permissions'   => wp_create_nonce( Visualizer_Plu
                                                'db_get_data'   => wp_create_nonce( Visualizer_Plu
                                                'json_get_roots'   => wp_create_nonce( Visualizer_
                                                'json_get_data'   => wp_create_nonce( Visualizer_P
                                                'json_set_schedule'   => wp_create_nonce( Visualiz
                                        ),
                                        'actions' => array(
                                                'permissions'   => Visualizer_Plugin::ACTION_FETCH
                                                'db_get_data'   => Visualizer_Plugin::ACTION_FETCH
                                                'json_get_roots'   => Visualizer_Plugin::ACTION_JS
                                                'json_get_data'   => Visualizer_Plugin::ACTION_JSO
                                                'json_set_schedule'   => Visualizer_Plugin::ACTION
                                        ),
                                ),
                                'db_query' => array(
                                        'tables'    => $table_col_mapping,
                                ),
                                'is_pro'    => Visualizer_Module::is_pro(),
                                'page_type' => 'chart',
                                'json_tag_separator' => Visualizer_Source_Json::TAG_SEPARATOR,
                                'json_tag_separator_view' => Visualizer_Source_Json::TAG_SEPARATOR
                                'is_front'  => false,
                                'rest_base' => get_rest_url( null, 'wc/v3/reports/' ),
                        )
                );

                $render          = new Visualizer_Render_Page_Data();
```

```
			$render->chart    = $this->_chart;
			$render->type     = $data['type'];
			$render->custom_css  = $data['css'];
			$render->sidebar = $sidebar;
			if ( filter_input( INPUT_GET, 'library', FILTER_VALIDATE_BOOLEAN ) ) {
				$render->button = filter_input( INPUT_GET, 'action' ) === Visualizer_Plugi
					? esc_html__( 'Save Chart', 'visualizer' )
					: esc_html__( 'Create Chart', 'visualizer' );
				if ( filter_input( INPUT_GET, 'action' ) === Visualizer_Plugin::ACTION_EDI
					$render->cancel_button = esc_html__( 'Cancel', 'visualizer' );
				}
			} else {
				$render->button = esc_attr__( 'Insert Chart', 'visualizer' );
			}

			do_action( 'visualizer_enqueue_scripts_and_styles', $data, $this->_chart->ID );

			if ( Visualizer_Module::is_pro() && Visualizer_Module::is_pro_older_than( '1.9.0'
				global $Visualizer_Pro;
				$Visualizer_Pro->_enqueueScriptsAndStyles( $data, $this->_chart->ID );
			}

			$this->_addAction( 'admin_head', 'renderFlattrScript' );
			wp_iframe( array( $render, 'render' ) );
		}

		/**
		 * Handles chart type selection page.
		 *
		 * @since 1.0.0
		 *
		 * @access private
		 */
		private function _handleTypesPage() {
			// process post request
			if ( $_SERVER['REQUEST_METHOD'] === 'POST' && wp_verify_nonce( filter_input( INPUT
				$type = filter_input( INPUT_POST, 'type' );
				$library = filter_input( INPUT_POST, 'chart-library' );
				if ( in_array( $type, Visualizer_Plugin::getChartTypes(), true ) ) {
					if ( empty( $library ) ) {
						// library cannot be empty.
						do_action( 'themeisle_log_event', Visualizer_Plugin::NAME,
						return;
					}

					// save new chart type
					update_post_meta( $this->_chart->ID, Visualizer_Plugin::CF_CHART_T
					update_post_meta( $this->_chart->ID, Visualizer_Plugin::CF_CHART_L
					// if the chart has default data, update it with appropriate defau
```

```php
                                if ( filter_var( get_post_meta( $this->_chart->ID, Visualizer_Plug
                                        $source = new Visualizer_Source_Csv( VISUALIZER_ABSPATH .
                                        $source->fetch();
                                        $this->_chart->post_content = $source->getData( get_post_m
                                        wp_update_post( $this->_chart->to_array() );
                                        update_post_meta( $this->_chart->ID, Visualizer_Plugin::CF
                                }

                                Visualizer_Module_Utility::set_defaults( $this->_chart );

                                // redirect to next tab
                                // changed by Ash/Upwork
                                wp_redirect( esc_url_raw( add_query_arg( 'tab', 'settings' ) ) );

                                return;
                        }
                }
                $render        = new Visualizer_Render_Page_Types();
                $render->type  = get_post_meta( $this->_chart->ID, Visualizer_Plugin::CF_CHART_TYP
                $render->types = Visualizer_Module_Admin::_getChartTypesLocalized( false, false, f
                $render->chart = $this->_chart;
                wp_enqueue_style( 'visualizer-frame' );
                wp_enqueue_script( 'visualizer-frame' );
                wp_iframe( array( $render, 'render' ) );
        }

        /**
         * Renders flattr script in the iframe <head>
         *
         * @since 1.4.2
         * @action admin_head
         *
         * @access public
         */
        public function renderFlattrScript() {
                echo '';
        }

        /**
         * Processes the CSV that is sent in the request as a string.
         *
         * @since 3.2.0
         */
        private function handleCSVasString( $data, $editor_type ) {
                $source = null;

                switch ( $editor_type ) {
                        case 'text':
                                // data coming in from the text editor.
```

```php
				$tmpfile = tempnam( get_temp_dir(), Visualizer_Plugin::NAME );
				$handle  = fopen( $tmpfile, 'w' );
				$values = preg_split( '/[\n\r]+/', stripslashes( trim( $data ) ) )
				if ( $values ) {
					foreach ( $values as $row ) {
						if ( empty( $row ) ) {
							continue;
						}
						$row = explode( ',', $row );
						$row = array_map(
							function( $r ) {
								return '' === $r ? ' ' : $r;
							},
							$row
						);
						$row = implode( ',', $row );
						// don't use fpucsv here because we need to just d
						// minus the empty rows
						// because fputcsv needs to tokenize
						// we can standardize the CSV enclosure here and r
						// we can assume that if there are an even number
						// but that will screw up words like fo'c'sle
						// so here let's just assume ' will NOT be used fo
						fwrite( $handle, $row );
						fwrite( $handle, PHP_EOL );
					}
				}
				$source = new Visualizer_Source_Csv( $tmpfile );
				fclose( $handle );
				break;
			case 'table':
				// Import from Chart
				// fall-through.
			case 'excel':
				// data coming in from the excel editor.
				$source = apply_filters( 'visualizer_pro_handle_chart_data', $data
				break;
		}
		return $source;
	}

	/**
	 * Parses the data uploaded as an HTML table.
	 *
	 * @since 3.2.0
	 *
	 * @access private
	 */
	private function handleTabularData() {
		$csv        = array();
		// the datatable mentions the headers twice, so lets remove the duplicates.
		$headers    = array_unique( array_filter( $_POST['header'] ) );
		$types      = $_POST['type'];

		// capture all the indexes that correspond to excluded columns.
		$exclude    = array();
		$index      = 0;
		foreach ( $types as $type ) {
			if ( empty( $type ) ) {
				$exclude[] = $index;
			}
			$index++;
		}

		// when N headers are being renamed, the number of headers increases by N
		// because of the way datatable duplicates header information
		// so unset the headers that have been renamed.
		if ( count( $headers ) !== count( $types ) ) {
			$to = count( $headers );
			for ( $i = count( $types ); $i < $to; $i++ ) {
				unset( $headers[ $i + 1 ] );
			}
		}

		$columns    = array();
		for ( $i = 0; $i < count( $headers ); $i++ ) {
			if ( ! isset( $_POST[ 'data' . $i ] ) ) {
				continue;
			}
			$columns[ $i ] = $_POST[ 'data' . $i ];
		}

		$csv[]      = $headers;
		$csv[]      = $types;
		for ( $j = 0; $j < count( $columns[0] ); $j++ ) {
			$row = array();
			for ( $i = 0; $i < count( $headers ); $i++ ) {
				$row[] = sanitize_text_field( $columns[ $i ][ $j ] );
			}
			$csv[]  = $row;
		}

		$tmpfile = tempnam( get_temp_dir(), Visualizer_Plugin::NAME );
		$handle  = fopen( $tmpfile, 'w' );

		if ( $csv ) {
			$index = 0;
			foreach ( $csv as $row ) {
				// remove all the cells corresponding to the excluded headers.
				foreach ( $exclude as $j ) {
					unset( $row[ $j ] );
				}
				fputcsv( $handle, $row );
			}
		}
		$source = new Visualizer_Source_Csv( $tmpfile );
		fclose( $handle );
		return $source;
	}

	/**
	 * Parses uploaded CSV file and saves new data for the chart.
	 *
	 * @since 1.0.0
	 *
	 * @access public
	 */
	public function uploadData() {
		// if this is being called internally from pro and VISUALIZER_DO_NOT_DIE is set.
		// otherwise, assume this is a normal web request.
		$can_die    = ! ( defined( 'VISUALIZER_DO_NOT_DIE' ) && VISUALIZER_DO_NOT_DIE );

		// validate nonce
		if ( ! isset( $_GET['nonce'] ) || ! wp_verify_nonce( $_GET['nonce'] ) ) {
			if ( ! $can_die ) {
				return;
			}
			status_header( 403 );
			exit;
		}

		// check chart, if chart exists
		// do not use filter_input as it does not work for phpunit test cases, use filter_
		$chart_id = isset( $_GET['chart'] ) ? filter_var( $_GET['chart'], FILTER_VALIDATE_
		if ( ! $chart_id || ! ( $chart = get_post( $chart_id ) ) || $chart->post_type !==
			if ( ! $can_die ) {
				return;
			}
			status_header( 400 );
			exit;
		}

		do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf( 'Uploading dat

		if ( ! isset( $_POST['vz-import-time'] ) ) {
			apply_filters( 'visualizer_pro_remove_schedule', $chart_id );
		}
```

```php

		if ( ! isset( $_POST['chart_data_src'] ) || Visualizer_Plugin::CF_SOURCE_FILTER !=
			// delete the filters in case this chart is being uploaded from other data
			delete_post_meta( $chart_id, Visualizer_Plugin::CF_FILTER_CONFIG );
			delete_post_meta( $chart_id, '__transient-' . Visualizer_Plugin::CF_FILTER
			delete_post_meta( $chart_id, '__transient-' . Visualizer_Plugin::CF_DB_QUE

			// delete "import from db" specific parameters.
			delete_post_meta( $chart_id, Visualizer_Plugin::CF_DB_QUERY );
			delete_post_meta( $chart_id, Visualizer_Plugin::CF_DB_SCHEDULE );
			delete_post_meta( $chart_id, Visualizer_Plugin::CF_REMOTE_DB_PARAMS );
		}

		// delete json related data.
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_URL );
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_ROOT );
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_PAGING );
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_JSON_HEADERS );

		// delete last error
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_ERROR );

		// delete editor related data.
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_EDITOR );

		// delete this so that a JSON import can be later edited manually without a proble
		delete_post_meta( $chart_id, Visualizer_Plugin::CF_EDITABLE_TABLE );

		$source = null;
		$render = new Visualizer_Render_Page_Update();

		$remote_data = false;
		if ( isset( $_POST['remote_data'] ) && function_exists( 'wp_http_validate_url' ) )
			$remote_data = wp_http_validate_url( $_POST['remote_data'] );
		}
		if ( false !== $remote_data ) {
			$source = new Visualizer_Source_Csv_Remote( $remote_data );
			if ( isset( $_POST['vz-import-time'] ) ) {
				apply_filters( 'visualizer_pro_chart_schedule', $chart_id, $remote
			}
			// phpcs:ignore WordPress.PHP.StrictComparisons.LooseComparison
		} elseif ( isset( $_FILES['local_data'] ) && $_FILES['local_data']['error'] == 0 )
			$source = new Visualizer_Source_Csv( $_FILES['local_data']['tmp_name'] );
		} elseif ( isset( $_POST['chart_data'] ) && strlen( $_POST['chart_data'] ) > 0 ) {
			$source = $this->handleCSVasString( $_POST['chart_data'], $_POST['editor-t
			update_post_meta( $chart_id, Visualizer_Plugin::CF_EDITOR, $_POST['editor-
		} elseif ( isset( $_POST['table_data'] ) && 'yes' === $_POST['table_data'] ) {
			$source = $this->handleTabularData();
			update_post_meta( $chart_id, Visualizer_Plugin::CF_EDITOR, $_POST['editor-
		} else {
			do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf( 'CSV f
			$render->message = esc_html__( 'CSV file with chart data was not uploaded.
			update_post_meta( $chart_id, Visualizer_Plugin::CF_ERROR, esc_html__( 'CSV
		}

		do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf( 'Uploaded data

		if ( $source ) {
			if ( $source->fetch() ) {
				$content    = $source->getData( get_post_meta( $chart_id, Visualiz
				$populate   = true;
				if ( is_string( $content ) && is_array( unserialize( $content ) ) )
					$json   = unserialize( $content );
					// if source exists, so should data. if source exists but
					// if we populate the data even if it is empty, the chart
					if ( array_key_exists( 'source', $json ) && ! empty( $json
						do_action( 'themeisle_log_event', Visualizer_Plugi
						update_post_meta( $chart_id, Visualizer_Plugin::CF
						$populate   = false;
					}
				}
				if ( $populate ) {
					$chart->post_content = $content;
				}
				wp_update_post( $chart->to_array() );

				update_post_meta( $chart->ID, Visualizer_Plugin::CF_SERIES, $sourc
				update_post_meta( $chart->ID, Visualizer_Plugin::CF_SOURCE, $sourc
				update_post_meta( $chart->ID, Visualizer_Plugin::CF_DEFAULT_DATA,

				do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf

				Visualizer_Module_Utility::set_defaults( $chart, null );

				$settings = get_post_meta( $chart->ID, Visualizer_Plugin::CF_SETTI
				if ( isset( $settings['series'] ) && ! ( count( $settings['series'
					$diff_total_series = abs( count( $settings['series'] ) - c
					if ( $diff_total_series ) {
						foreach ( range( 1, $diff_total_series ) as $k =>
							$settings['series'][] = end( $settings['se
						}
						update_post_meta( $chart->ID, Visualizer_Plugin::C
					}
				}

				$render->id     = $chart->ID;
				$render->data   = json_encode( $source->getRawData( get_post_meta(
				$render->series = json_encode( $source->getSeries() );
```

```
					$render->settings = json_encode( $settings );
				} else {
					$error = $source->get_error();
					if ( empty( $error ) ) {
						$error = esc_html__( 'CSV file is broken or invalid. Pleas
					}
					$render->message = $error;
					do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf
					update_post_meta( $chart_id, Visualizer_Plugin::CF_ERROR, $error )
				}
			} else {
				do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf( 'Unkno
			}
			$render->render();
			if ( ! $can_die ) {
				return;
			}
			defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
		}

		/**
		 * Clones the chart.
		 *
		 * @since 1.0.0
		 *
		 * @access public
		 */
		public function cloneChart() {
			$chart_id = $success = false;
			$nonce    = isset( $_GET['nonce'] ) && wp_verify_nonce( $_GET['nonce'], Visualizer
			$capable  = current_user_can( 'edit_posts' );
			if ( $nonce && $capable ) {
				$chart_id = isset( $_GET['chart'] ) ? filter_var( $_GET['chart'], FILTER_V
				if ( $chart_id ) {
					$chart   = get_post( $chart_id );
					$success = $chart && $chart->post_type === Visualizer_Plugin::CPT_
				}
			}
			$redirect = remove_query_arg( 'vaction', wp_get_referer() );
			if ( $success ) {
				$new_chart_id = wp_insert_post(
					array(
						'post_type'    => Visualizer_Plugin::CPT_VISUALIZER,
						'post_title'   => 'Visualization',
						'post_author'  => get_current_user_id(),
						'post_status'  => $chart->post_status,
						'post_content' => $chart->post_content,
					)
				);
```

```
					if ( is_wp_error( $new_chart_id ) ) {
						do_action( 'themeisle_log_event', Visualizer_Plugin::NAME, sprintf
					} else {
						$post_meta = get_post_meta( $chart_id );
						foreach ( $post_meta as $key => $value ) {
							if ( strpos( $key, 'visualizer-' ) !== false ) {
								add_post_meta( $new_chart_id, $key, maybe_unserial
							}
						}
						$redirect = esc_url(
							add_query_arg(
								array(
									'page' => 'visualizer',
									'type' => filter_input( INPUT_GET, 'type'
									'vaction' => false,
								),
								admin_url( 'admin.php' )
							),
							null,
							'db'
						);
					}
				}

				if ( defined( 'WP_TESTS_DOMAIN' ) ) {
					wp_die();
				}
				wp_redirect( $redirect );
				exit;
			}

			/**
			 * Exports the chart data
			 *
			 * @since 1.0.0
			 *
			 * @access public
			 */
			public function exportData() {
				check_ajax_referer( Visualizer_Plugin::ACTION_EXPORT_DATA . Visualizer_Plugin::VER
				$capable  = current_user_can( 'edit_posts' );
				if ( $capable ) {
					$chart_id = isset( $_GET['chart'] ) ? filter_var(
						$_GET['chart'],
						FILTER_VALIDATE_INT,
						array(
							'options' => array(
								'min_range' => 1,
							),
                                )
                        ) : '';
                        if ( $chart_id ) {
                                $data   = $this->_getDataAs( $chart_id, 'csv' );
                                if ( $data ) {
                                        echo wp_send_json_success( $data );
                                }
                        }
                }

                defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
        }

        /**
         * Handles chart data page.
         *
         * @since 1.0.0
         *
         * @access private
         */
        private function _handleDataPage() {
                $data          = $this->_getChartArray();
                $render        = new Visualizer_Render_Page_Data();
                $render->chart = $this->_chart;
                $render->type  = $data['type'];

                if ( $data && $data['settings'] ) {
                        unset( $data['settings']['width'], $data['settings']['height'], $data['set
                }
                wp_enqueue_style( 'visualizer-frame' );
                wp_enqueue_script( 'visualizer-render' );
                wp_localize_script(
                        'visualizer-render',
                        'visualizer',
                        array(
                                'l10n'   => array(
                                        'invalid_source' => esc_html__( 'You have entered an inval
                                        'loading'       => esc_html__( 'Loading...', 'visualizer'
                                ),
                                'charts' => array(
                                        'canvas' => $data,
                                ),
                        )
                );

                do_action( 'visualizer_enqueue_scripts_and_styles', $data, $this->_chart->ID );

                if ( Visualizer_Module::is_pro() && Visualizer_Module::is_pro_older_than( '1.9.0'
                        global $Visualizer_Pro;
```

```
				$Visualizer_Pro->_enqueueScriptsAndStyles( $data, $this->_chart->ID );
			}

			// Added by Ash/Upwork
			$this->_addAction( 'admin_head', 'renderFlattrScript' );
			wp_iframe( array( $render, 'render' ) );
		}

		/**
		 * Returns the data for the query.
		 *
		 * @access public
		 */
		public function getQueryData() {
			check_ajax_referer( Visualizer_Plugin::ACTION_FETCH_DB_DATA . Visualizer_Plugin::V

			$params     = wp_parse_args( $_POST['params'] );
			$chart_id   = filter_var( $params['chart_id'], FILTER_VALIDATE_INT );

			$source     = new Visualizer_Source_Query( stripslashes( $params['query'] ), $char
			$html       = $source->fetch( true );
			$error      = $source->get_error();
			if ( ! empty( $error ) ) {
				wp_send_json_error( array( 'msg' => $error ) );
			}
			wp_send_json_success( array( 'table' => $html ) );
		}

		/**
		 * Saves the query and the schedule.
		 *
		 * @access public
		 */
		public function saveQuery() {
			check_ajax_referer( Visualizer_Plugin::ACTION_SAVE_DB_QUERY . Visualizer_Plugin::V

			$chart_id   = filter_input(
				INPUT_GET,
				'chart',
				FILTER_VALIDATE_INT,
				array(
					'options' => array(
						'min_range' => 1,
					),
				)
			);

			$hours = filter_input(
				INPUT_POST,
					'refresh',
					FILTER_VALIDATE_FLOAT,
					array(
						'options' => array(
							'min_range' => -1,
							'max_range' => apply_filters( 'visualizer_is_business', fa
						),
					)
				);

				if ( ! is_numeric( $hours ) ) {
					$hours = -1;
				}

				$render = new Visualizer_Render_Page_Update();
				if ( $chart_id ) {
					$params     = wp_parse_args( $_POST['params'] );
					$source     = new Visualizer_Source_Query( stripslashes( $params['query']
					$source->fetch( false );
					$error      = $source->get_error();
					if ( empty( $error ) ) {
						update_post_meta( $chart_id, Visualizer_Plugin::CF_DB_QUERY, strip
						update_post_meta( $chart_id, Visualizer_Plugin::CF_SOURCE, $source
						update_post_meta( $chart_id, Visualizer_Plugin::CF_SERIES, $source
						update_post_meta( $chart_id, Visualizer_Plugin::CF_DB_SCHEDULE, $h
						update_post_meta( $chart_id, Visualizer_Plugin::CF_DEFAULT_DATA, 0
						if ( isset( $params['db_type'] ) && $params['db_type'] !== Visuali
							$remote_db_params   = $params;
							unset( $remote_db_params['query'] );
							unset( $remote_db_params['chart_id'] );
							update_post_meta( $chart_id, Visualizer_Plugin::CF_REMOTE_
						}

						$schedules              = get_option( Visualizer_Plugin::CF_DB_SCH
						$schedules[ $chart_id ] = time() + $hours * HOUR_IN_SECONDS;
						update_option( Visualizer_Plugin::CF_DB_SCHEDULE, $schedules );

						wp_update_post(
							array(
								'ID'           => $chart_id,
								'post_content'  => $source->getData( get_post_meta
							)
						);
						$render->data   = json_encode( $source->getRawData( get_post_meta(
						$render->series = json_encode( $source->getSeries() );
						$render->id = $chart_id;
					} else {
						$render->message = $error;
					}
```

```php
			}
			$render->render();
			if ( ! ( defined( 'VISUALIZER_DO_NOT_DIE' ) && VISUALIZER_DO_NOT_DIE ) ) {
				defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
			}
		}


		/**
		 * Saves the filter query and the schedule.
		 *
		 * @access public
		 */
		public function saveFilter() {
			check_ajax_referer( Visualizer_Plugin::ACTION_SAVE_FILTER_QUERY . Visualizer_Plugi

			$chart_id   = filter_input(
				INPUT_GET,
				'chart',
				FILTER_VALIDATE_INT,
				array(
					'options' => array(
						'min_range' => 1,
					),
				)
			);

			$hours = filter_input(
				INPUT_POST,
				'refresh',
				FILTER_VALIDATE_INT,
				array(
					'options' => array(
						'min_range' => -1,
						'max_range' => apply_filters( 'visualizer_is_business', fa
					),
				)
			);

			if ( 0 !== $hours && empty( $hours ) ) {
				$hours = -1;
			}

			do_action( 'visualizer_save_filter', $chart_id, $hours );

			if ( ! ( defined( 'VISUALIZER_DO_NOT_DIE' ) && VISUALIZER_DO_NOT_DIE ) ) {
				defined( 'WP_TESTS_DOMAIN' ) ? wp_die() : exit();
			}
		}
```

```php

        /**
         * Save chart image.
         *
         * @param string $base64_img Chart image.
         * @param int    $chart_id Chart ID.
         * @param bool   $save_attachment Save attachment.
         * @return attachment ID
         */
        public function save_chart_image( $base64_img, $chart_id, $save_attachment = true ) {
                // Delete old chart image.
                $old_attachment_id = get_post_meta( $chart_id, Visualizer_Plugin::CF_CHART_IMAGE,
                if ( $old_attachment_id ) {
                        wp_delete_attachment( $old_attachment_id, true );
                }

                if ( ! $save_attachment ) {
                        return 0;
                }

                // Upload dir.
                $upload_dir  = wp_upload_dir();
                $upload_path = str_replace( '/', DIRECTORY_SEPARATOR, $upload_dir['path'] ) . DIRE

                $img             = str_replace( 'data:image/png;base64,', '', $base64_img );
                $img             = str_replace( ' ', '+', $img );
                $decoded         = base64_decode( $img );
                $filename        = 'visualization-' . $chart_id . '.png';
                $file_type       = 'image/png';
                $hashed_filename = $filename;

                // Save the image in the uploads directory.
                require_once ABSPATH . '/wp-admin/includes/file.php';
                \WP_Filesystem();
                global $wp_filesystem;
                $upload_file = $wp_filesystem->put_contents( $upload_path . $hashed_filename, $dec

                // Insert new chart image.
                $attachment = array(
                        'post_mime_type' => $file_type,
                        'post_title'     => preg_replace( '/\.[^.]+$/', '', basename( $hashed_file
                        'post_content'   => '',
                        'post_status'    => 'inherit',
                        'guid'           => $upload_dir['url'] . '/' . basename( $hashed_filename
                );

                $attach_id = wp_insert_attachment( $attachment, $upload_dir['path'] . '/' . $hashe
                return $attach_id;
        }
```

```
}
```