**Plugin Name:** WP Maintenance Scheduler

**Plugin Description:** Automates essential WordPress maintenance tasks like plugin updates, cache clearing, and optimization.

**Folder Structure:**

wp-maintenance-scheduler/  
├── wp-maintenance-scheduler.php  
├── includes/  
│ ├── settings.php  
│ ├── scheduler.php  
│ └── reporter.php  
└── languages/  
 └── wp-maintenance-scheduler.pot

**wp-maintenance-scheduler.php:**

PHP

<?php  
/\*\*  
 \* Plugin Name: WP Maintenance Scheduler  
 \* Plugin URI: https://www.example.com/wp-maintenance-scheduler  
 \* Description: Automates essential WordPress maintenance tasks.  
 \* Version: 1.0  
 \* Author: Your Name  
 \* Author URI: https://www.example.com/  
 \* License: GPL2  
 \* License URI: https://www.gnu.org/licenses/gpl-2.0.html  
 \* Text Domain: wp-maintenance-scheduler  
 \* Domain Path: /languages  
 \*/  
  
// If this file is called directly, abort.  
if ( ! defined( 'WPINC' ) ) {  
 die;  
}  
  
// Define plugin constants.  
define( 'WPMS\_PLUGIN\_PATH', plugin\_dir\_path( \_\_FILE\_\_ ) );  
define( 'WPMS\_PLUGIN\_URL', plugin\_dir\_url( \_\_FILE\_\_ ) );  
  
// Include necessary files.  
require\_once WPMS\_PLUGIN\_PATH . 'includes/settings.php';  
require\_once WPMS\_PLUGIN\_PATH . 'includes/scheduler.php';  
require\_once WPMS\_PLUGIN\_PATH . 'includes/reporter.php';  
  
// Register activation and deactivation hooks.  
register\_activation\_hook( \_\_FILE\_\_, 'wpms\_activate' );  
register\_deactivation\_hook( \_\_FILE\_\_, 'wpms\_deactivate' );  
  
/\*\*  
 \* Plugin activation callback.  
 \*/  
function wpms\_activate() {  
 // Schedule the cron event.  
 wpms\_schedule\_event();  
}  
  
/\*\*  
 \* Plugin deactivation callback.  
 \*/  
function wpms\_deactivate() {  
 // Clear the scheduled cron event.  
 wp\_schedule\_event( time() - DAY\_IN\_SECONDS, 'daily', 'wpms\_cron\_hook' );  
}  
  
/\*\*  
 \* Schedule the cron event.  
 \*/  
function wpms\_schedule\_event() {  
 if ( ! wp\_next\_scheduled( 'wpms\_cron\_hook' ) ) {  
 $schedule\_time = wp\_next\_scheduled( 'wpms\_cron\_hook', array(), 'wpms\_schedule' );  
 wp\_schedule\_event( $schedule\_time, 'wpms\_schedule', 'wpms\_cron\_hook' );  
 }  
}  
  
add\_action( 'wpms\_cron\_hook', 'wpms\_run\_maintenance' );  
  
/\*\*  
 \* Run the maintenance tasks.  
 \*/  
function wpms\_run\_maintenance() {  
 // 1. Update outdated plugins.  
 wpms\_update\_plugins();  
  
 // 2. Clear W3 Total Cache.  
 wpms\_clear\_w3tc\_cache();  
  
 // 3. Run WP-Optimize optimizations.  
 wpms\_run\_wpoptimize();  
  
 // 4. Send the report.  
 wpms\_send\_report();  
}  
  
/\*\*  
 \* Update outdated plugins.  
 \*/  
function wpms\_update\_plugins() {  
 // Include the plugin updater.  
 include\_once ABSPATH . 'wp-admin/includes/class-wp-upgrader.php';  
  
 // Get outdated plugins.  
 $outdated\_plugins = get\_site\_transient( 'update\_plugins' );  
  
 if ( isset( $outdated\_plugins->response ) && is\_array( $outdated\_plugins->response ) ) {  
 // Loop through outdated plugins and update them.  
 foreach ( $outdated\_plugins->response as $plugin\_file => $plugin\_data ) {  
 $upgrader = new Plugin\_Upgrader( new Automatic\_Upgrader\_Skin() );  
 $upgrader->upgrade( $plugin\_file );  
 }  
 }  
}  
  
/\*\*  
 \* Clear W3 Total Cache.  
 \*/  
function wpms\_clear\_w3tc\_cache() {  
 if ( function\_exists( 'w3tc\_flush\_all' ) ) {  
 w3tc\_flush\_all();  
 }  
}  
  
/\*\*  
 \* Run WP-Optimize optimizations.  
 \*/  
function wpms\_run\_wpoptimize() {  
 if ( class\_exists( 'WP\_Optimize' ) ) {  
 WP\_Optimize()->get\_page\_cache()->purge();  
 WP\_Optimize()->get\_db\_cleaner()->clean\_all();  
 }  
}  
  
/\*\*  
 \* Send the report.  
 \*/  
function wpms\_send\_report() {  
 // Get the recipient email address from the settings.  
 $recipient\_email = get\_option( 'wpms\_email\_address' );  
  
 // Prepare the report content.  
 $report\_content = 'WP Maintenance Scheduler Report:' . "\n\n";  
 $report\_content .= 'Plugins updated: ' . ( isset( $outdated\_plugins->response ) ? count( $outdated\_plugins->response ) : 0 ) . "\n";  
 $report\_content .= 'W3 Total Cache cleared: ' . ( function\_exists( 'w3tc\_flush\_all' ) ? 'Yes' : 'No' ) . "\n";  
 $report\_content .= 'WP-Optimize optimizations run: ' . ( class\_exists( 'WP\_Optimize' ) ? 'Yes' : 'No' ) . "\n";  
  
 // Send the email.  
 wp\_mail( $recipient\_email, 'WP Maintenance Scheduler Report', $report\_content );  
}

**includes/settings.php:**

PHP

<?php  
// Add the admin menu page.  
add\_action( 'admin\_menu', 'wpms\_add\_admin\_menu' );  
  
/\*\*  
 \* Add the admin menu page.  
 \*/  
function wpms\_add\_admin\_menu() {  
 add\_options\_page(  
 'WP Maintenance Scheduler',  
 'WP Maintenance Scheduler',  
 'manage\_options',  
 'wp-maintenance-scheduler',  
 'wpms\_settings\_page'  
 );  
}  
  
/\*\*  
 \* Render the settings page.  
 \*/  
function wpms\_settings\_page() {  
 ?>  
 <div class="wrap">  
 <h1>WP Maintenance Scheduler Settings</h1>  
  
 <form method="post" action="options.php">  
 <?php settings\_fields( 'wpms\_settings\_group' ); ?>  
 <?php do\_settings\_sections( 'wp-maintenance-scheduler' ); ?>  
  
 <table class="form-table">  
 <tr valign="top">  
 <th scope="row">Schedule Kickoff Time</th>  
 <td>  
 <input type="time" name="wpms\_kickoff\_time" value="<?php echo esc\_attr( get\_option( 'wpms\_kickoff\_time' ) ); ?>" />  
 </td>  
 </tr>  
 <tr valign="top">  
 <th scope="row">Frequency</th>  
 <td>  
 <select name="wpms\_frequency">  
 <option value="daily" <?php selected( get\_option( 'wpms\_frequency' ), 'daily' ); ?>>Daily</option>  
 <option value="weekly" <?php selected( get\_option( 'wpms\_frequency' ), 'weekly' ); ?>>Weekly</option>  
 <option value="monthly" <?php selected( get\_option( 'wpms\_frequency' ), 'monthly' ); ?>>Monthly</option>  
 </select>  
 </td>  
 </tr>  
 <tr valign="top">  
 <th scope="row">Email Address</th>  
 <td>  
 <input type="email" name="wpms\_email\_address" value="<?php echo esc\_attr( get\_option( 'wpms\_email\_address' ) ); ?>" />  
 </td>  
 </tr>  
 </table>  
  
 <?php submit\_button(); ?>  
 </form>  
 </div>  
 <?php  
}  
  
// Register settings.  
add\_action( 'admin\_init', 'wpms\_register\_settings' );  
  
/\*\*  
 \* Register settings.  
 \*/  
function wpms\_register\_settings() {  
 register\_setting( 'wpms\_settings\_group', 'wpms\_kickoff\_time' );  
 register\_setting( 'wpms\_settings\_group', 'wpms\_frequency' );  
 register\_setting( 'wpms\_settings\_group', 'wpms\_email\_address' );  
}

**includes/scheduler.php:**

PHP

<?php  
// Schedule the cron event based on user settings.  
add\_filter( 'cron\_schedules', 'wpms\_add\_cron\_schedule' );  
  
/\*\*  
 \* Add custom cron schedule.  
 \*/  
function wpms\_add\_cron\_schedule( $schedules ) {  
 $schedules['wpms\_schedule'] = array(  
 'interval' => wpms\_get\_schedule\_interval(),  
 'display' => 'WP Maintenance Schedule',  
 );  
  
 return $schedules;  
}  
  
/\*\*  
 \* Get the schedule interval in seconds.  
 \*/  
function wpms\_get\_schedule\_interval() {  
 $frequency = get\_option( 'wpms\_frequency', 'daily' );  
  
 switch ( $frequency ) {  
 case 'weekly':  
 return WEEK\_IN\_SECONDS;  
 case 'monthly':  
 return MONTH\_IN\_SECONDS;  
 default:  
 return DAY\_IN\_SECONDS;  
 }  
}  
  
// Schedule the event at the specified kickoff time.  
add\_action( 'wpms\_cron\_hook', 'wpms\_schedule\_event\_at\_kickoff\_time' );  
  
/\*\*  
 \* Schedule the event at the kickoff time.  
 \*/  
function wpms\_schedule\_event\_at\_kickoff\_time() {  
 $kickoff\_time = get\_option( 'wpms\_kickoff\_time', '00:00' );  
 $timestamp = strtotime( $kickoff\_time );  
  
 if ( $timestamp !== false ) {  
 wp\_schedule\_event( $timestamp, 'wpms\_schedule', 'wpms\_cron\_hook' );  
 }  
}

**includes/reporter.php:**

PHP

<?php  
/\*\*  
 \* Send the report email.  
 \*/  
function wpms\_send\_report() {  
 // Get the recipient email address from the settings.  
 $recipient\_email = get\_option( 'wpms\_email\_address' );  
  
 // Prepare the report content.  
 $report\_content = 'WP Maintenance Scheduler Report:' . "\n\n";  
 $report\_content .= 'Plugins updated: ' . ( isset( $GLOBALS['wpms\_updated\_plugins'] ) ? count( $GLOBALS['wpms\_updated\_plugins'] ) : 0 ) . "\n";  
 $report\_content .= 'W3 Total Cache cleared: ' . ( function\_exists( 'w3tc\_flush\_all' ) ? 'Yes' : 'No' ) . "\n";  
 $report\_content .= 'WP-Optimize optimizations run: ' . ( class\_exists( 'WP\_Optimize' ) ? 'Yes' : 'No' ) . "\n";  
  
 // Send the email.  
 wp\_mail( $recipient\_email, 'WP Maintenance Scheduler Report', $report\_content );  
}

**languages/wp-maintenance-scheduler.pot:**

This file is a template for translations. You can use a tool like Poedit to generate translation files for different languages.

**Documentation:**

1. **Installation:**
   * Download the plugin ZIP file.
   * In your WordPress dashboard, go to "Plugins" -> "Add New".
   * Click "Upload Plugin" and select the downloaded ZIP file.
   * Click "Install Now" and then "Activate".
2. **Configuration:**
   * Go to "Settings" -> "WP Maintenance Scheduler".
   * Set the desired "Schedule Kickoff Time" (e.g., "03:00" for 3:00 AM).
   * Choose the "Frequency" (Daily, Weekly, or Monthly).
   * Enter the "Email Address" where you want to receive the reports.
   * Click "Save Changes".
3. **How it Works:**
   * The plugin uses WordPress cron to schedule the maintenance tasks.
   * At the specified kickoff time and frequency, the plugin will:
     + Check for outdated plugins and update them automatically.
     + Clear the W3 Total Cache (if installed).
     + Run WP-Optimize optimizations (if installed).
     + Send a report email with details of the run.

**Explanation:**

* The main plugin file (wp-maintenance-scheduler.php) sets up the plugin, includes necessary files, and registers activation/deactivation hooks.
* The settings.php file handles the admin settings page.
* The scheduler.php file manages the cron scheduling based on user preferences.
* The reporter.php file generates and sends the report email.

This detailed breakdown should provide a good starting point for creating your WordPress maintenance plugin. Remember to test it thoroughly and consider adding more features or customizations as needed.