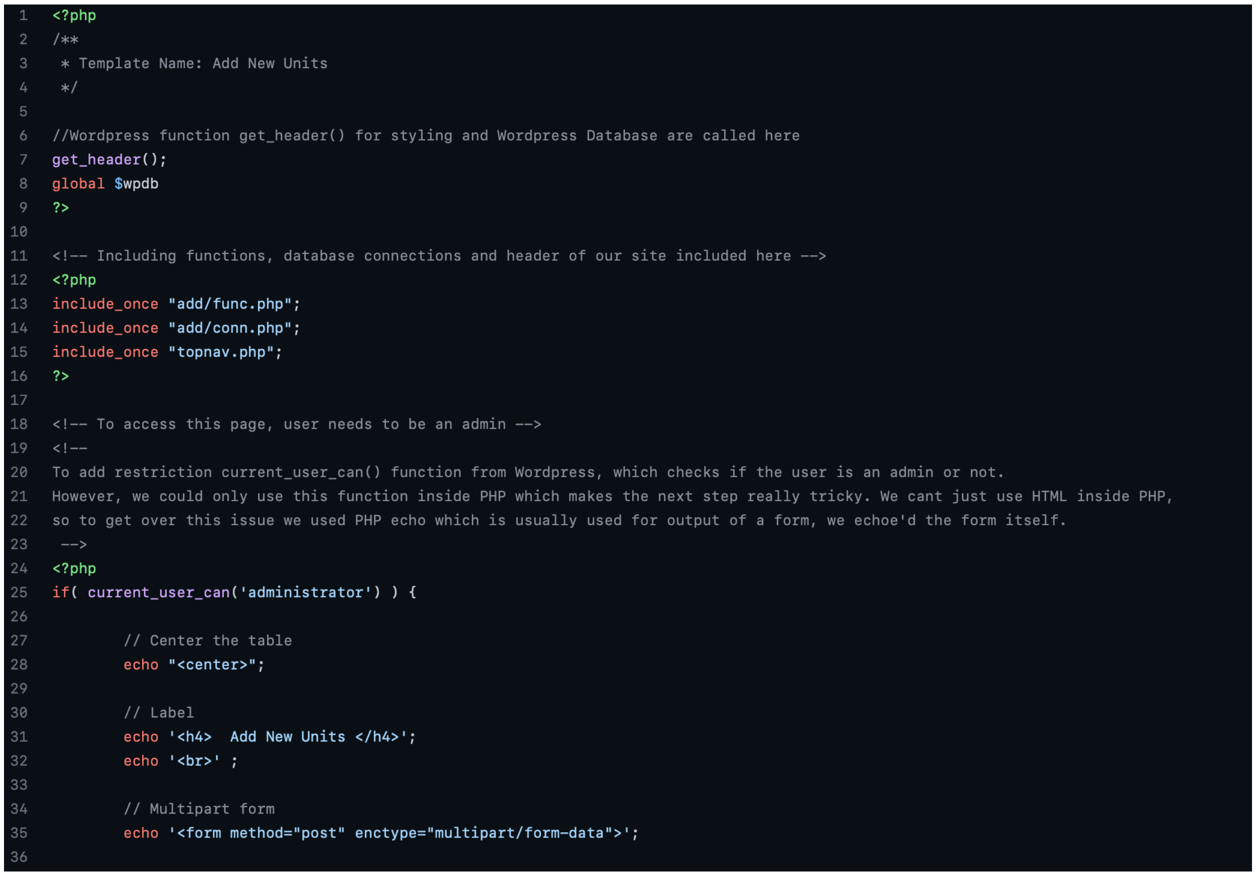
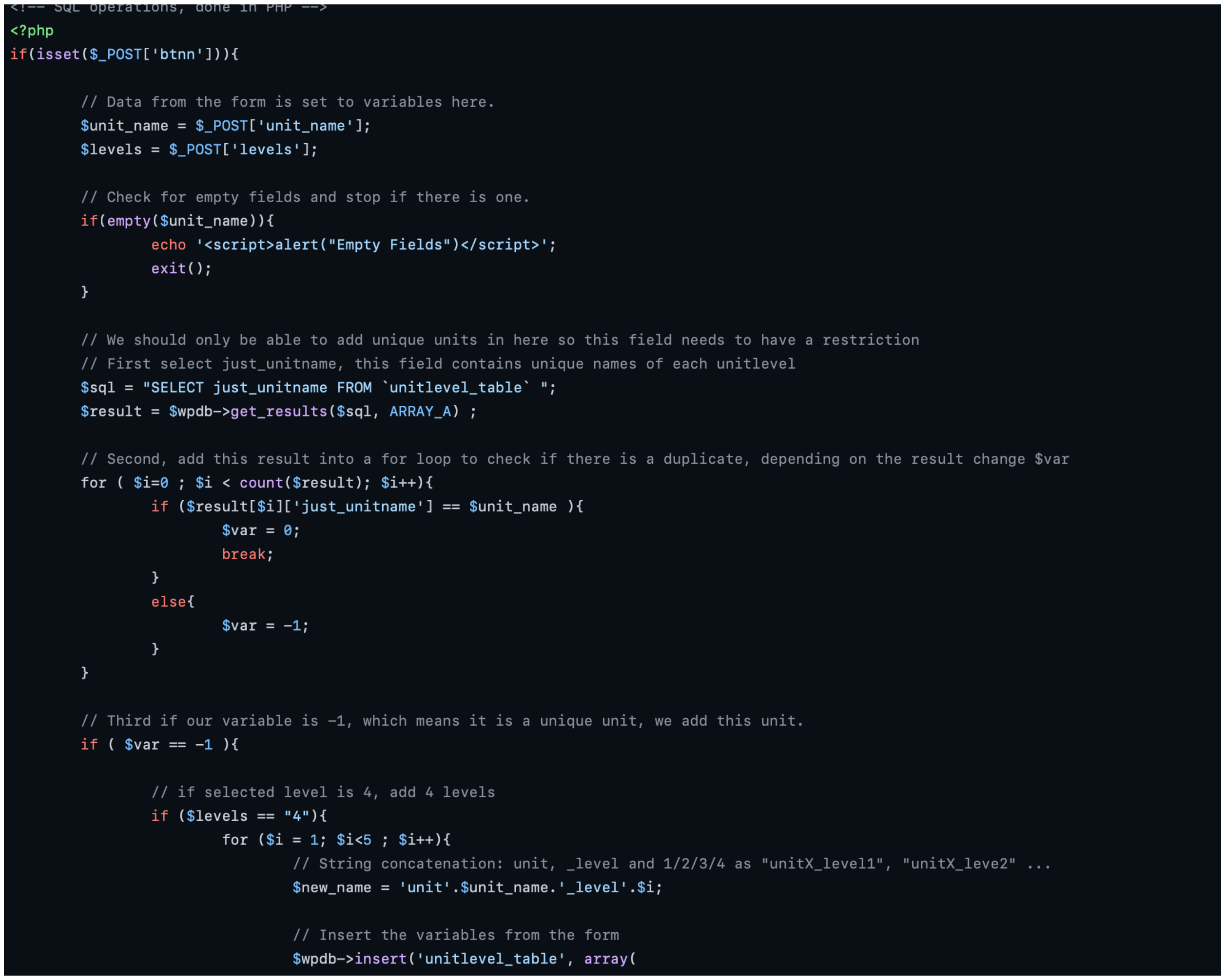
add-new-units.php



```php
<?php
/**
 * Template Name: Add New Units
 */

//Wordpress function get_header() for styling and Wordpress Database are called here
get_header();
global $wpdb
?>

<!-- Including functions, database connections and header of our site included here -->
<?php
include_once "add/func.php";
include_once "add/conn.php";
include_once "topnav.php";
?>

<!-- To access this page, user needs to be an admin -->
<!--
To add restriction current_user_can() function from Wordpress, which checks if the user is an admin or not.
However, we could only use this function inside PHP which makes the next step really tricky. We cant just use HTML inside PHP,
so to get over this issue we used PHP echo which is usually used for output of a form, we echoe'd the form itself.
 -->
<?php
if( current_user_can('administrator') ) {

        // Center the table
        echo "<center>";

        // Label
        echo '<h4>  Add New Units </h4>';
        echo '<br>' ;

        // Multipart form
        echo '<form method="post" enctype="multipart/form-data">';

```

The php code above is used to add new units to a table. The first part of the code checks if the user is logged in and has the correct permissions. If not, they are redirected to the login page. If they do have the correct permissions, they are given a form to fill out.



```php
<!-- SQL operations, done in PHP -->
<?php
if(isset($_POST['btnn'])){

        // Data from the form is set to variables here.
        $unit_name = $_POST['unit_name'];
        $levels = $_POST['levels'];

        // Check for empty fields and stop if there is one.
        if(empty($unit_name)){
                echo '<script>alert("Empty Fields")</script>';
                exit();
        }

        // We should only be able to add unique units in here so this field needs to have a restriction
        // First select just_unitname, this field contains unique names of each unitlevel
        $sql = "SELECT just_unitname FROM `unitlevel_table` ";
        $result = $wpdb->get_results($sql, ARRAY_A) ;

        // Second, add this result into a for loop to check if there is a duplicate, depending on the result change $var
        for ( $i=0 ; $i < count($result); $i++){
                if ($result[$i]['just_unitname'] == $unit_name ){
                        $var = 0;
                        break;
                }
                else{
                        $var = -1;
                }
        }

        // Third if our variable is -1, which means it is a unique unit, we add this unit.
        if ( $var == -1 ){

                // if selected level is 4, add 4 levels
                if ($levels == "4"){
                        for ($i = 1; $i<5 ; $i++){
                                // String concatenation: unit, _level and 1/2/3/4 as "unitX_level1", "unitX_leve2" ...
                                $new_name = 'unit'.$unit_name.'_level'.$i;

                                // Insert the variables from the form
                                $wpdb->insert('unitlevel_table', array(
```

The form asks for a unit name and level count. The level count is used in a for loop to create multiple rows in the table, one for each level of the unit. The form also has a button that, when clicked, inserts the data into the table.

The code also includes some CSS styling to make the form look nicer. Finally, the Wordpress function get_footer() is called to include the footer.php file.

The php code above is used to add new units to the unitlevel_table. First, the code checks if the unit_name is unique. If it is not unique, an error message appears. If it is unique, the code then inserts the unit into the table. The user can choose to add three or four levels for each unit.