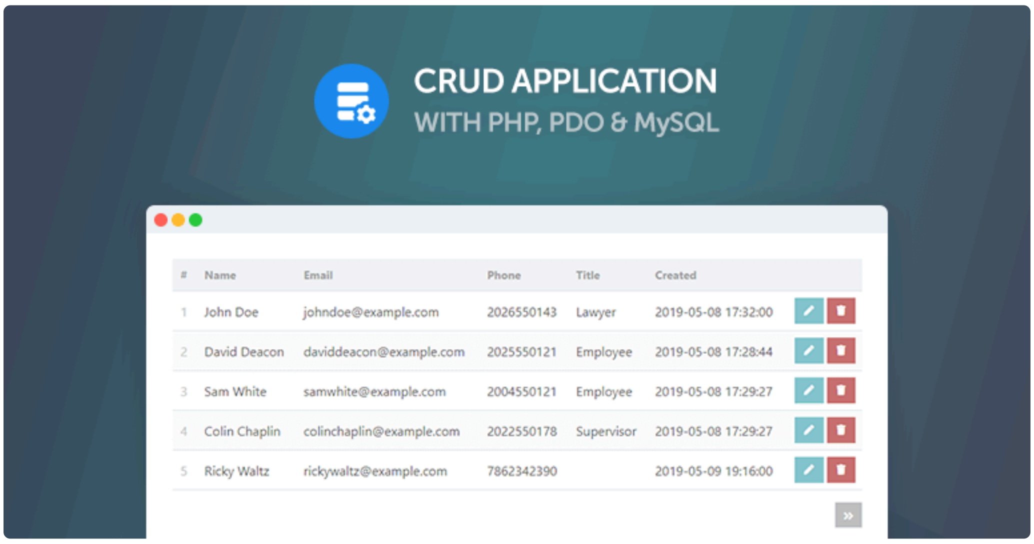
CodeShack Tutorials Tools Snippets References Packages

HTML CSS JavaScript PHP Python MySQL

CRUD Application with PHP, PDO, and MySQL

Updated on September 13, 2023 by David Adams



#	Name	Email	Phone	Title	Created
1	John Doe	johndoe@example.com	2026550143	Lawyer	2019-05-08 17:32:00
2	David Deacon	daviddeacon@example.com	2025550121	Employee	2019-05-08 17:28:44
3	Sam White	samwhite@example.com	2004550121	Employee	2019-05-08 17:29:27
4	Colin Chaplin	colinchaplin@example.com	2022550178	Supervisor	2019-05-08 17:29:27
5	Ricky Waltz	rickywaltz@example.com	7862342390		2019-05-09 19:16:00

In this tutorial, we'll be developing a complete Create, Read, Update, and Delete application with PHP, PDO, and MySQL. We'll be creating the app completely from scratch. No additional frameworks are required.

A CRUD app is often used in conjunction with a database, interacting with records in a table and populating them in an HTML table element. We'll be using MySQL as our database management system in our app.

For this tutorial, we'll create a MySQL database with a contacts table, which will consist of a variety of different columns (name, email, phone, etc.).

The Advanced package includes additional features and a download link to the source code.

Contents

1. Getting Started

Before we jump into programming our CRUD app, we need to install our web server and set up our app.

1.1. What You Will Learn in this Tutorial

- **Create MySQL Records** — Insert new records into the **Contacts** table.
- **Read MySQL Records** — Reading MySQL records and populating them in an HTML table.
- **Update MySQL Records** — Update existing MySQL records in the **Contacts** table.
- **Delete MySQL Records** — Confirm and delete records from the **Contacts** table.
- **Implementing GET and POST Requests** — Send data to our app from an HTML form and URL parameters.
- **Prepared Statements** — Secure our SQL statements with prepared statements.

1.2. Requirements

- **Web Server** — I recommend you download and install XAMPP on your local computer system, this server package includes MySQL, PHP, phpMyAdmin, and the PDO extension.
- **PHP** — I recommend you use the latest version of PHP, but older versions should work just fine (skip if you installed XAMPP).
- **PDO Extension** — Should be enabled by default if you're using XAMPP, but if it's not you'll need to enable/install it.

1.3. File Structure & Setup

Navigate to **C:\xampp\htdocs** (XAMPP) and create the below directories and files.

File Structure

```
\-- phpcrud
    |-- index.php
    |-- create.php
    |-- read.php
    |-- update.php
    |-- delete.php
    |-- functions.php
    |-- style.css
```

What each file will contain:

- **index.php** — Home page for our CRUD app.

- **read.php** — Display records from our database table and navigate with pagination.
- **update.php** — Update existing records with an HTML form and send data to the server with a POST request.
- **delete.php** — Confirm and delete records by ID (GET request to get the ID).
- **functions.php** — Basic templating functions and MySQL connection function (so we don't have to repeat code in every file).
- **style.css** — The stylesheet for our app, which will change the appearance of our app.

2. Creating the Database and setting-up Tables

The MySQL database we'll use to store contacts and retrieve them with PHP. If you're using XAMPP, follow the below instructions.

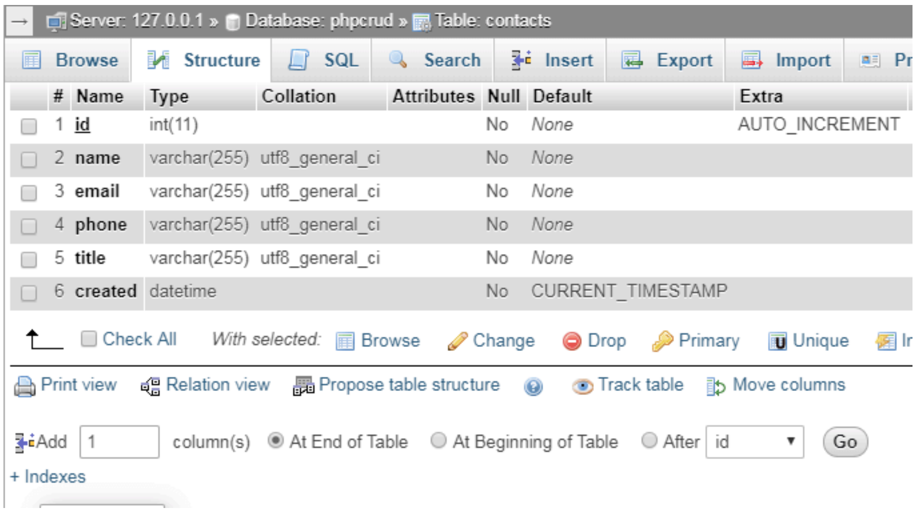
- Navigate to **http://localhost/phpmyadmin/**
- Click **Databases** at the top
- Under **Create database** input **phpcrud** and select **utf8_general_ci** as the collation
- Click **Create**
- Select the newly created database
- Click the **SQL** tab and execute the below SQL:

```
CREATE TABLE IF NOT EXISTS `contacts` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `name` varchar(255) NOT NULL,
    `email` varchar(255) NOT NULL,
    `phone` varchar(255) NOT NULL,
    `title` varchar(255) NOT NULL,
    `created` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8;

INSERT INTO `contacts` (`id`, `name`, `email`, `phone`, `title`, `created`) VALUES
(1, 'John Doe', 'johndoe@example.com', '2026550143', 'Lawyer', '2019-05-08 17:32:00'),
(2, 'David Deacon', 'daviddeacon@example.com', '2025550121', 'Employee', '2019-05-08 1
(3, 'Sam White', 'samwhite@example.com', '2004550121', 'Employee', '2019-05-08 17:29:2
(4, 'Colin Chaplin', 'colinchaplin@example.com', '2022550178', 'Supervisor', '2019-05-
(5, 'Ricky Waltz', 'rickywaltz@example.com', '7862342390', '', '2019-05-09 19:16:00'),
(6, 'Arnold Hall', 'arnoldhall@example.com', '5089573579', 'Manager', '2019-05-09 19:1
(7, 'Toni Adams', 'alvah1981@example.com', '2603668738', '', '2019-05-09 19:19:00'),
(8, 'Donald Perry', 'donald1983@example.com', '7019007916', 'Employee', '2019-05-09 19
(9, 'Joe McKinney', 'nadia.doole0@example.com', '6153353674', 'Employee', '2019-05-09
(10, 'Angela Horst', 'angela1977@example.com', '3094234980', 'Assistant', '2019-05-09
(11, 'James Jameson', 'james1965@example.com', '4002349823', 'Assistant', '2019-05-09
(12, 'Daniel Deacon', 'danieldeacon@example.com', '5003423549', 'Manager', '2019-05-09
```

The above SQL will create the table: **contacts**, which we'll be using in our app, and included in the SQL is sample data — this data will be used for testing purposes to make sure everything is working as it should, you can delete it later on.

There are six columns in the **contacts** table (id, name, email, phone, title, and created). The title column is basically the role of each contact. You can change this to anything you want. The

In **phpMyAdmin**, the database should look like the following:



#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id	int(11)			No	None	AUTO_INCREMENT
2	name	varchar(255)	utf8_general_ci		No	None	
3	email	varchar(255)	utf8_general_ci		No	None	
4	phone	varchar(255)	utf8_general_ci		No	None	
5	title	varchar(255)	utf8_general_ci		No	None	
6	created	datetime			No	CURRENT_TIMESTAMP	

3. Creating the Stylesheet (CSS3)

The stylesheet will change the appearance of our app, edit the **style.css** file and add the following code:

```
* {
    box-sizing: border-box;
    font-family: -apple-system, BlinkMacSystemFont, "segoe ui", roboto, oxygen, ubuntu
    font-size: 16px;
    -webkit-font-smoothing: antialiased;
    -moz-osx-font-smoothing: grayscale;
}
body {
    background-color: #FFFFFF;
    margin: 0;
}
.navtop {
    background-color: #3f69a8;
    height: 60px;
    width: 100%;
    border: 0;
}
.navtop div {
    display: flex;
    margin: 0 auto;
    width: 1000px;
    height: 100%;
}
.navtop div h1, .navtop div a {
    display: inline-flex;
    align-items: center;
}
.navtop div h1 {
    flex: 1;
    font-size: 24px;
    padding: 0;
    margin: 0;
    color: #ecf0f6;
    font-weight: normal;
}
.navtop div a {
    font-weight: bold;
}
.navtop div a i {
    padding: 2px 8px 0 0;
}
.navtop div a:hover {
    color: #ecf0f6;
}
.content {
    width: 1000px;
    margin: 0 auto;
}
.content h2 {
    margin: 0;
    padding: 25px 0;
    font-size: 22px;
    border-bottom: 1px solid #ebebeb;
    color: #666666;
}
.read .create-contact {
    display: inline-block;
    text-decoration: none;
    background-color: #38b673;
    font-weight: bold;
    font-size: 14px;
    color: #FFFFFF;
    padding: 10px 15px;
    margin: 15px 0;
}
.read .create-contact:hover {
    background-color: #32a367;
}
.read .pagination {
    display: flex;
    justify-content: flex-end;
}
.read .pagination a {
    display: inline-block;
    text-decoration: none;
    background-color: #a5a7a9;
    font-weight: bold;
    color: #FFFFFF;
    padding: 5px 10px;
    margin: 15px 0 15px 5px;
}
.read .pagination a:hover {
    background-color: #999b9d;
}
.read table {
    width: 100%;
    padding-top: 30px;
    border-collapse: collapse;
}
.read table thead {
    background-color: #ebeef1;
    border-bottom: 1px solid #d3dae0;
}
.read table thead td {
    padding: 10px;
    font-weight: bold;
    color: #767779;
    font-size: 14px;
```

```
}
.read table tbody tr:nth-child(even) {
    background-color: #fbfcfc;
}
.read table tbody tr:hover {
    background-color: #376ab7;
}
.read table tbody tr:hover td {
    color: #FFFFFF;
}
.read table tbody tr:hover td:nth-child(1) {
    color: #FFFFFF;
}
.read table tbody tr td {
    padding: 10px;
}
.read table tbody tr td:nth-child(1) {
    color: #a5a7a9;
}
.read table tbody tr td.actions {
    padding: 8px;
    text-align: right;
}
.read table tbody tr td.actions .edit, .read table tbody tr td.actions .trash {
    display: inline-flex;
    text-align: right;
    text-decoration: none;
    color: #FFFFFF;
    padding: 10px 12px;
}
.read table tbody tr td.actions .trash {
    background-color: #b73737;
}
.read table tbody tr td.actions .trash:hover {
    background-color: #a33131;
}
.read table tbody tr td.actions .edit {
    background-color: #37afb7;
}
.read table tbody tr td.actions .edit:hover {
    background-color: #319ca3;
}
.update form {
    padding: 15px 0;
    display: flex;
    flex-flow: wrap;
}
.update form label {
    display: inline-flex;
    width: 400px;
    padding: 10px 0;
    margin-right: 25px;
}
.update form input {
    padding: 10px;
    width: 400px;
    margin-right: 25px;
    margin-bottom: 15px;
    border: 1px solid #cccccc;
}
.update form input[type="submit"] {
    display: block;
    font-size: 14px;
    color: #FFFFFF;
    cursor: pointer;
    width: 200px;
    margin-top: 15px;
}
.update form input[type="submit"]:hover {
    background-color: #32a367;
}
.delete .yesno {
    display: flex;
}
.delete .yesno a {
    display: inline-block;
    text-decoration: none;
    background-color: #38b673;
    font-weight: bold;
    color: #FFFFFF;
    padding: 10px 15px;
    margin: 15px 10px 15px 0;
}
.delete .yesno a:hover {
    background-color: #32a367;
}
```

Feel free to customize the stylesheet. This is what I've put together to make the CRUD app more appealing.

4. Creating the CRUD App

We can finally start to code the CRUD app with PHP. Before we start, ensure you've completed the previous steps and have the MySQL database ready. Otherwise, without the database, our app will be redundant.

4.1. Creating the Functions

This file will contain functions that we can execute in all our PHP files. This is so we don't have to write the same code in every PHP file. The shorter the code, the better, right? We'll create three functions — one function will connect to the database, and the other two will be the templates for the header and footer that will appear on every page we create and will contain the HTML layout.

Edit the **functions.php** file and add the following code:

```
<?php
function pdo_connect_mysql() {
    $DATABASE_HOST = 'localhost';
    $DATABASE_USER = 'root';
    $DATABASE_PASS = '';
    $DATABASE_NAME = 'phpcrud';
    try {
        return new PDO('mysql:host=' . $DATABASE_HOST . ';dbname=' . $DATABASE_NAME .
    } catch (PDOException $exception) {
        // If there is an error with the connection, stop the script and display the e
        exit('Failed to connect to database!');
    }
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>$title</title>
        <link href="style.css" rel="stylesheet" type="text/css">
        <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/a
    </head>
    <body>
    <nav class="navtop">
        <div>
            <h1>Website Title</h1>
            <a href="index.php"><i class="fas fa-home"></i>Home</a>
            <a href="read.php"><i class="fas fa-address-book"></i>Contacts</a>
        </div>
    </nav>
EOT;
}
function template_footer() {
echo <<<EOT
    </body>
</html>
EOT;
}
?>
```

Make sure to change the MySQL connection details to reflect your details. We're utilizing the PDO interface to connect to our MySQL database. PDO will make it easier for us to interact with our MySQL database and secure our queries.

Also, indenting the template code may cause parsing errors.

4.2. Creating the Home Page

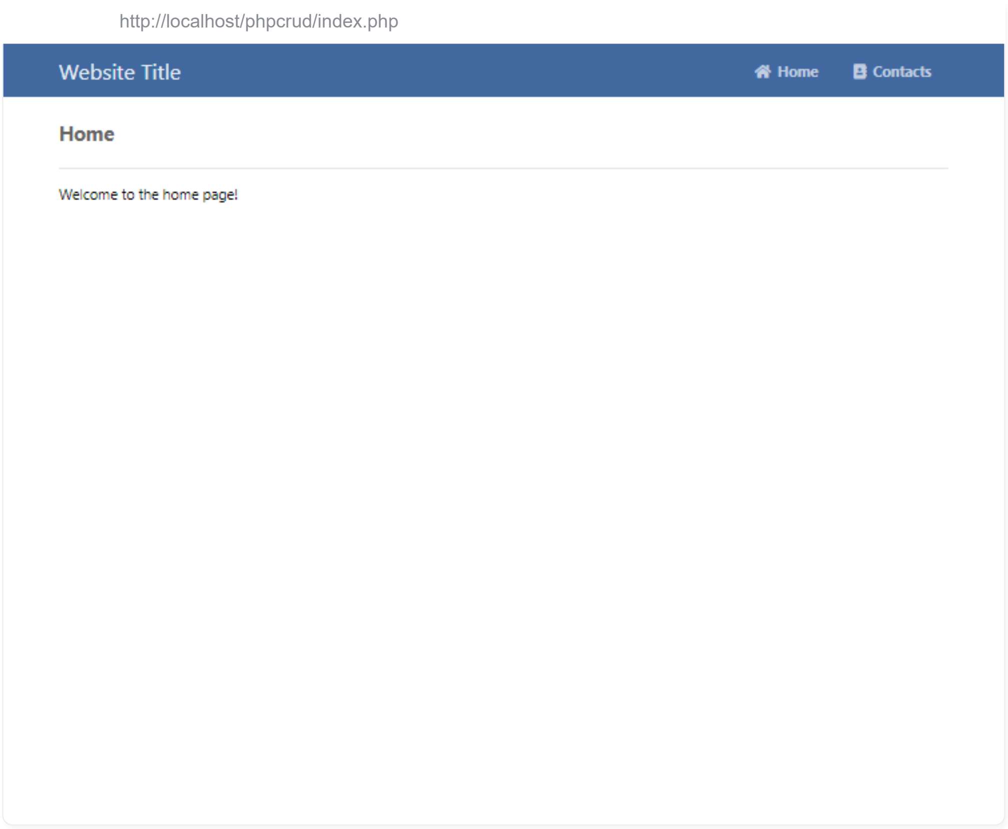
When you navigate to **http://localhost/phpcrud/** it will serve the **index.php** file, this page will be our home page.

Edit the **index.php** file and add the following code:

```
<?php
include 'functions.php';
// Your PHP code here.

// Home Page template below.
?>

<?=template_header('Home')?>

<div class="content">
    <h2>Home</h2>
    <p>Welcome to the home page!</p>
</div>

<?=template_footer()?>
```

This will create a basic home page. We can use this page to navigate to the other pages. As you can see, we include the **functions.php** file and execute the template functions that we created.

And now if we navigate to **http://localhost/phpcrud/** we'll see the following:



Home

Welcome to the home page!

That's basically it for the home page. Feel free to add your own content, as this page is basically a portal to our other pages that we'll create in the next section.

4.3. Creating the Read Page

In this section, we'll develop the read page that'll be responsible for populating records from our **contacts** table in an HTML table.

Edit the **read.php** file and add the following code:

```
<?php
include 'functions.php';
// Connect to MySQL database
$pdo = pdo_connect_mysql();
// Get the page via GET request (URL param: page), if non exists default the page to 1
$page = isset($_GET['page']) && is_numeric($_GET['page']) ? (int)$_GET['page'] : 1;
// Number of records to show on each page
$records_per_page = 5;
```

Once again, we include the functions file, but this time we connect to our MySQL database by executing the function: `pdo_connect_mysql`. If the connection is successful, we can use the `$pdo` variable to execute queries.

We also create two more variables — the `$page` variable will determine the page that the user is currently on, the `$records_per_page` will be used to limit the number of records to display on each page, for example, if we limit the number of records to 5 and we have 10 records in our

Add the following code to the **read.php** file:

```
// Prepare the SQL statement and get records from our contacts table, LIMIT will deter
$stmt = $pdo->prepare('SELECT * FROM contacts ORDER BY id LIMIT :current_page, :record
$stmt->bindValue(':current_page', ($page-1)*$records_per_page, PDO::PARAM_INT);
$stmt->bindValue(':record_per_page', $records_per_page, PDO::PARAM_INT);
$stmt->execute();
// Fetch the records so we can display them in our template.
$contacts = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

The above code will select records from the **contacts** table. This will be determined by the current page the user is on, the records will be ordered by the **id** column, we can easily change the order by column if we wanted to, for example, if we change it to **created** then it will sort the records by the create date instead.

Security Tip

Utilizing prepared PDO statements will prevent SQL injection and therefore protect the database from exposure to suspicious users.

Add the following code to the **read.php** file:

```
// Get the total number of contacts, this is so we can determine whether there should
$num_contacts = $pdo->query('SELECT COUNT(*) FROM contacts')->fetchColumn();
?>
```

The above SQL query will get the total number of records in the **contacts** table. We don't need to use a prepared statement here because the query doesn't include user input variables.

Add the following code to the **read.php** file:

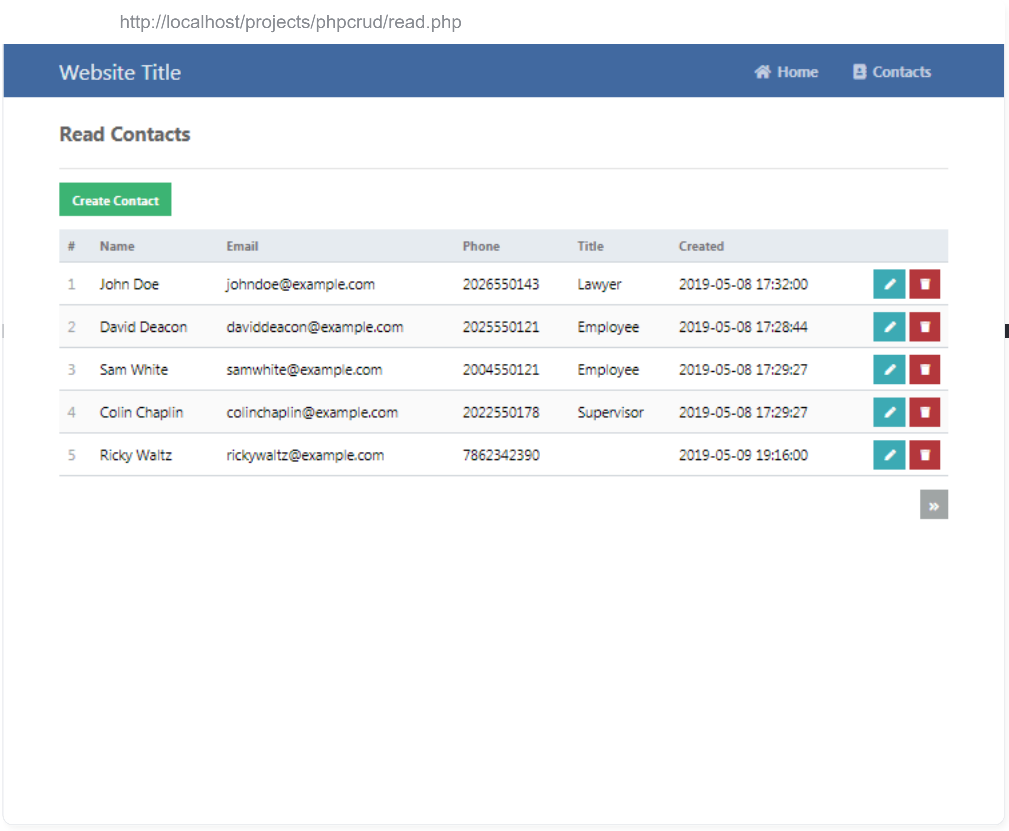
```
<?=template_header('Read')?>

<div class="content read">
    <h2>Read Contacts</h2>
    <a href="create.php" class="create-contact">Create Contact</a>
    <table>
        <thead>
            <tr>
                <td>#</td>
                <td>Name</td>
                <td>Email</td>
                <td>Phone</td>
                <td>Title</td>
                <td>Created</td>
                <td></td>
            </tr>
        </thead>
        <tbody>
            <?php foreach ($contacts as $contact): ?>
            <tr>
                <td><?=$contact['id']?></td>
                <td><?=$contact['name']?></td>
                <td><?=$contact['email']?></td>
                <td class="actions">
                    <a href="update.php?id=<?=$contact['id']?>" class="edit"><i class=
                    <a href="delete.php?id=<?=$contact['id']?>" class="trash"><i class
                </td>
            </tr>
            <?php endforeach; ?>
        </tbody>
    </table>
    <div class="pagination">
        <?php if ($page > 1): ?>
        <a href="read.php?page=<?=$page-1?>"><i class="fas fa-angle-double-left fa-sm"
        <?php endif; ?>
        <?php if ($page*$records_per_page < $num_contacts): ?>
        <a href="read.php?page=<?=$page+1?>"><i class="fas fa-angle-double-right fa-sm
        <?php endif; ?>
    </div>
</div>

<?=template_footer()?>
```

This is the template for the read page, which iterates the contacts array and adds them to the HTML table. We'll be able to read the records in a table format when we navigate to the read page.

Pagination is added so we can navigate between pages on the read page (page 1, page 2, etc.).

For the icons we're using Font Awesome, make sure that it's included in the header template function, or the icons will not appear.

And now if we navigate to **http://localhost/phpcrud/read.php**, we'll see the following:



#	Name	Email	Phone	Title	Created	
1	John Doe	johndoe@example.com	2026550143	Lawyer	2019-05-08 17:32:00	
2	David Deacon	daviddeacon@example.com	2025550121	Employee	2019-05-08 17:28:44	
3	Sam White	samwhite@example.com	2004550121	Employee	2019-05-08 17:29:27	
4	Colin Chaplin	colinchaplin@example.com	2022550178	Supervisor	2019-05-08 17:29:27	
5	Ricky Waltz	rickywaltz@example.com	7862342390		2019-05-09 19:16:00	

created those pages yet.

You can also click the **Contacts** link in the header bar, which will subsequently navigate to the read page.

4.4. Creating the Create Page

The create page will be used to create new records and insert them into our **Contacts** table.

Edit the **create.php** file and add:

```
<?php
include 'functions.php';
$pdo = pdo_connect_mysql();
$msg = '';
// Check if POST data is not empty
if (!empty($_POST)) {
    // Post data not empty insert a new record
    // Set-up the variables that are going to be inserted, we must check if the POST v
    $id = isset($_POST['id']) && !empty($_POST['id']) && $_POST['id'] != 'auto' ? $_PC
    // Check if POST variable "name" exists, if not default the value to blank, basica
    $name = isset($_POST['name']) ? $_POST['name'] : '';
    $email = isset($_POST['email']) ? $_POST['email'] : '';
    $phone = isset($_POST['phone']) ? $_POST['phone'] : '';
    $title = isset($_POST['title']) ? $_POST['title'] : '';
    $created = isset($_POST['created']) ? $_POST['created'] : date('Y-m-d H:i:s');
    // Insert new record into the contacts table
    $stmt = $pdo->prepare('INSERT INTO contacts VALUES (?, ?, ?, ?, ?, ?)');
    $stmt->execute([$id, $name, $email, $phone, $title, $created]);
    // Output message
    $msg = 'Created Successfully!';
}
?>
```

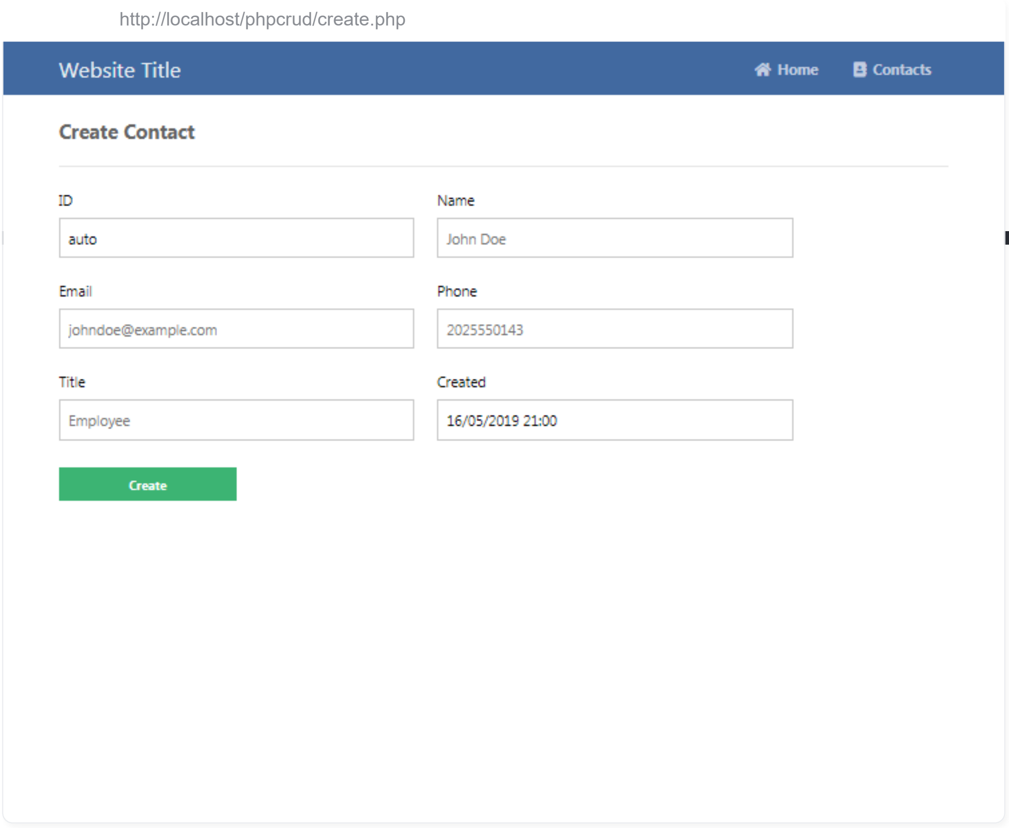
The above code will check if the **POST** array (form data) is not empty. If it's not, then it basically means the user has filled out the form and clicked the submit button, which will then insert a new record into our **Contacts** table.

Add after:

```
<?=template_header('Create')?>

<div class="content update">
    <h2>Create Contact</h2>
    <form action="create.php" method="post">
        <label for="id">ID</label>
        <label for="name">Name</label>
        <input type="text" name="id" placeholder="26" value="auto" id="id">
        <input type="text" name="name" placeholder="John Doe" id="name">
        <label for="email">Email</label>
        <label for="phone">Phone</label>
        <input type="text" name="email" placeholder="johndoe@example.com" id="email">
        <input type="text" name="phone" placeholder="2025550143" id="phone">
        <label for="title">Title</label>
        <label for="created">Created</label>
        <input type="text" name="title" placeholder="Employee" id="title">
        <input type="datetime-local" name="created" value="<?=date('Y-m-d\TH:i')?>" id
    <p><?=$msg?></p>
    <?php endif; ?>
</div>

<?=template_footer()?>
```

This is the template for our create page. As you can see, we have created a form and named each input field accordingly. The name of the input field is how we'll retrieve the POST variable in our PHP code. For example, if we name an input field "zip_code", we can get the value of that input field with `$_POST['zip_code']` in PHP (assuming that the form's method is set to **post**).

Finally, if we navigate to **http://localhost/phpcrud/create.php** or click the **Create** button on the read page, we'll see the following:



Create Contact

ID	Name
auto	John Doe
Email	Phone
johndoe@example.com	2025550143
Title	Created
Employee	16/05/2019 21:00

4.5. Creating the Update Page

The update page will be used to update records in our **Contacts** table. This page is similar to the create page, but instead of inserting a new record into the database, we'll be updating existing records. We'll be able to get the record ID with a GET request.

Edit the **update.php** file and add:

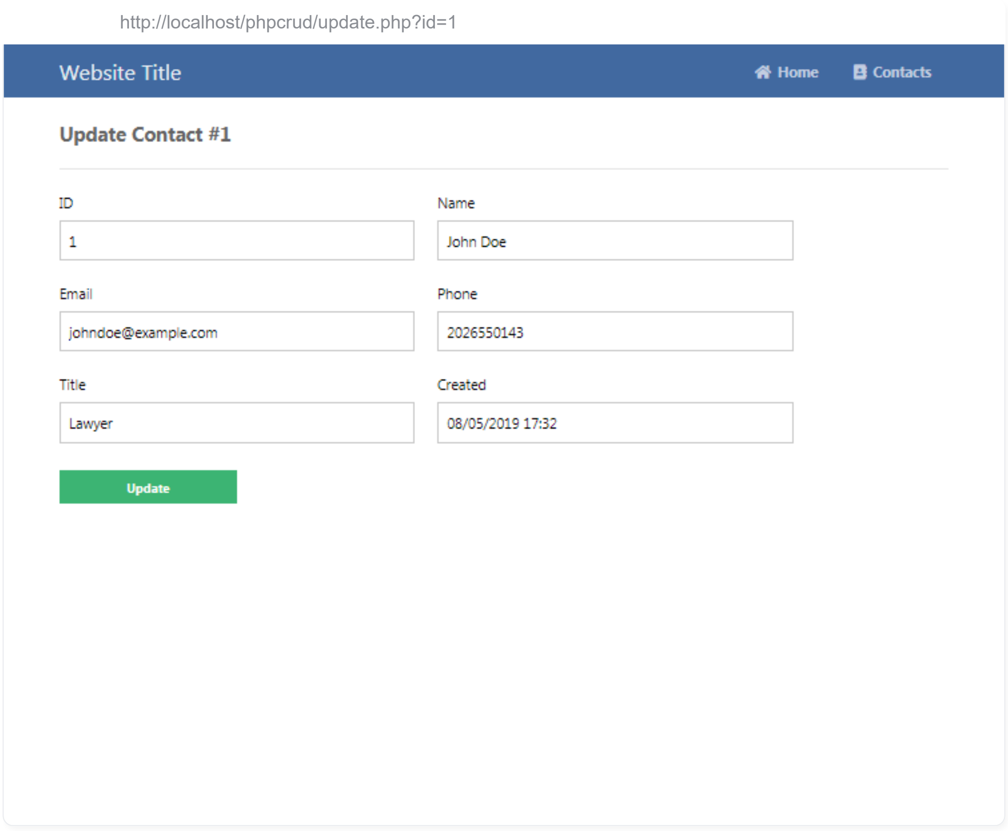
```
<?php
include 'functions.php';
$pdo = pdo_connect_mysql();
$msg = '';
// Check if the contact id exists, for example update.php?id=1 will get the contact wi
if (isset($_GET['id'])) {
    if (!empty($_POST)) {
        // This part is similar to the create.php, but instead we update a record and
        $phone = isset($_POST['phone']) ? $_POST['phone'] : '';
        $title = isset($_POST['title']) ? $_POST['title'] : '';
        $created = isset($_POST['created']) ? $_POST['created'] : date('Y-m-d H:i:s');
        // Update the record
        $stmt = $pdo->prepare('UPDATE contacts SET id = ?, name = ?, email = ?, phone
        $stmt->execute([$id, $name, $email, $phone, $title, $created, $_GET['id']]);
        $msg = 'Updated Successfully!';
    }
    // Get the contact from the contacts table
    $stmt = $pdo->prepare('SELECT * FROM contacts WHERE id = ?');
    $stmt->execute([$_GET['id']]);
    $contact = $stmt->fetch(PDO::FETCH_ASSOC);
    if (!$contact) {
        exit('Contact doesn\'t exist with that ID!');
    }
} else {
    exit('No ID specified!');
}
?>
```

The above code will check for the contact ID, which will be a parameter in the URL, for example, **http://localhost/phpcrud/update.php?id=1** will get the contact with the ID of 1, and then we can handle the request with the GET method and execute a MySQL query that will get the contact by the specified ID.

Add after:

```
<?=template_header('Read')?>

<div class="content update">
    <h2>Update Contact #<?=$contact['id']?></h2>
    <form action="update.php?id=<?=$contact['id']?>" method="post">
        <label for="id">ID</label>
        <label for="name">Name</label>
        <input type="text" name="id" placeholder="1" value="<?=$contact['id']?>" id="i
        <input type="text" name="name" placeholder="John Doe" value="<?=$contact['name
        <label for="email">Email</label>
        <label for="phone">Phone</label>
        <input type="text" name="email" placeholder="johndoe@example.com" value="<?=$c
        <input type="text" name="phone" placeholder="2025550143" value="<?=$contact['p
        <label for="title">Title</label>
        <label for="created">Created</label>
        <input type="text" name="title" placeholder="Employee" value="<?=$contact['tit
        <input type="datetime-local" name="created" value="<?=date('Y-m-d\TH:i', strtc
        <input type="submit" value="Update">
    </form>
    <?php if ($msg): ?>
    <p><?=$msg?></p>
    <?php endif; ?>
</div>

<?=template_footer()?>
```

This is the template for the update page. The input values are already specified with the contact columns. The MySQL query we implemented previously will retrieve those values.



4.6. Creating the Delete Page

The delete page will be used to delete records from the **Contacts** table. Before a user can delete a record, they will need to confirm it, as it will prevent accidental deletion.

Edit the **delete.php** file and add:

```php
<?php
include 'functions.php';
$pdo = pdo_connect_mysql();
$msg = '';
// Check that the contact ID exists
if (isset($_GET['id'])) {
    // Select the record that is going to be deleted
    $stmt = $pdo->prepare('SELECT * FROM contacts WHERE id = ?');
    $stmt->execute([$_GET['id']]);
    $contact = $stmt->fetch(PDO::FETCH_ASSOC);
    if (!$contact) {
        exit('Contact doesn\'t exist with that ID!');
    }
    // Make sure the user confirms beore deletion
    if (isset($_GET['confirm'])) {
        if ($_GET['confirm'] == 'yes') {
            // User clicked the "Yes" button, delete record
            $stmt = $pdo->prepare('DELETE FROM contacts WHERE id = ?');
            $stmt->execute([$_GET['id']]);
            $msg = 'You have deleted the contact!';
        } else {
            // User clicked the "No" button, redirect them back to the read page
            header('Location: read.php');
            exit;
        }
}
?>
```

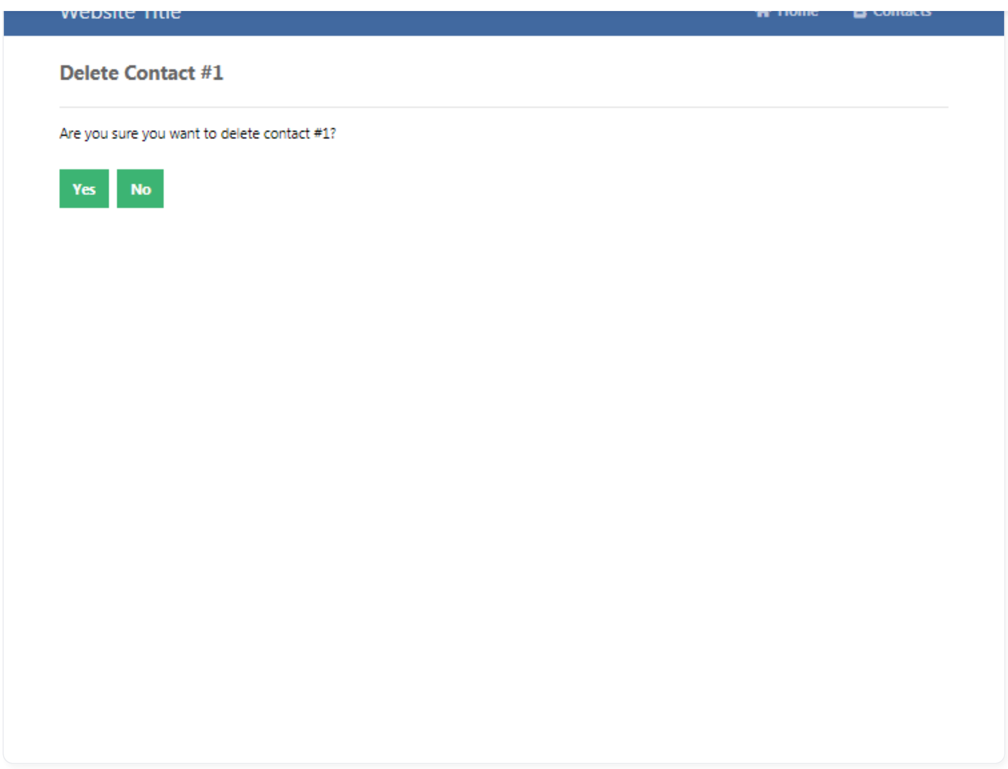
To delete a record, the code will check if the GET request variable **"id"** exists. If it does, then check if the record exists in the **Contacts** table and prompt the user whether they would like to delete the contact or not. A simple GET request will determine which button the user clicked (Yes or No).

Add after:

```
<?=template_header('Delete')?>

<div class="content delete">
    <h2>Delete Contact #<?=$contact['id']?></h2>
    <?php if ($msg): ?>
    <p><?=$msg?></p>
    <?php else: ?>
    <p>Are you sure you want to delete contact #<?=$contact['id']?>?</p>
    <div class="yesno">
        <a href="delete.php?id=<?=$contact['id']?>&confirm=yes">Yes</a>
        <a href="delete.php?id=<?=$contact['id']?>&confirm=no">No</a>
    </div>
    <?php endif; ?>
</div>

<?=template_footer()?>
```

The above code is the template for the delete page, which includes the **Yes** and **No** buttons (delete confirmation) and the output message. The **Yes** and **No** buttons will initiate a new GET request that will confirm the user's choice.

On the read page (Contacts), click the delete button on one of the records, and we should see something like the following:



Delete Contact #1

Are you sure you want to delete contact #1?

Please understand that if you uploaded the code to a production server, anyone with access could interact with your database (delete, update, and edit records) and therefore I highly advise you integrate it with a secure login system.

Conclusion

Congratulations! You have successfully created a basic CRUD app with PHP and MySQL. What next? Consider adding your own columns to the **Contacts** table and to the code.

If you've enjoyed this tutorial, don't forget to share using the social links below and check out our many more tutorials on our website.

Enjoy coding!

If you would like to support us, consider the advanced crud application below. It will greatly help us create more tutorials and keep our website up and running. The advanced package includes improved code and more features.

	Basic	Advanced
Source Code	✓	✓
Database SQL File	✓	✓
Secure CRUD System	✓	✓
Create, Read, Update & Delete Functions	✓	✓

Pagination Feature	✓	✓
Limit Records Per Page	✓	✓
Enhanced Pagination	✗	✓
Sortable Table Columns	✗	✓
Export & Import (CSV)	✗	✓
Bulk Actions	✗	✓
Elegant UI	✗	✓
Dynamic Version — Convert a table to a CRUD system in minutes	✗	✓
Responsive Design (mobile-friendly)	✗	✓
SCSS File	✗	✓
Commented Code	✓	✓
Free updates & support (bugs and minor issues)	✓	✓
NO Code Restrictions	✓	✓
User Guide	✓	✓
* Payments are processed with PayPal/Stripe. * Advanced package also includes the tutorial source and basic package. * Instant download after payment.	**$20.00** **Card & More (Stripe)** Download **PayPal** Download	

To learn more about the advanced package, please visit the Advanced CRUD Application page.

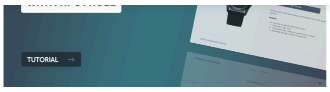
About the Author: David Adams

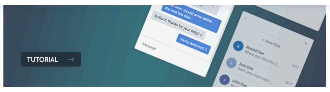
I'm an enthusiastic full-stack engineer who's been in the web development scene for over a decade. I enjoy the creativity I put into my projects and what others bring to the awesome web. My goal is to share my knowledge and help newcomers develop their skills.

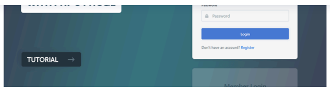
crud mysql pdo php programming tutorials

← **Login System with Python Flask and MySQL** **EU Cookie Consent Popup with JavaScript** →

RELATED POSTS


Shopping Cart System with PHP and MySQL


Live Support Chat with AJAX, PHP and MySQL


Secure Login System with PHP and MySQL

219 Comments Sort by **Newest**

Add to the discussion...

B Barton Jetter 2 months ago

Hello David,

I have been working on learning PHP and purchased the advanced package. I have learned so much just from working with it and modifying it as a personal project. My question is this, in the advanced dynamic you have a config.php that allows me to declare all the fields as well as things like lables, type, sortable, etc... i was wanting to make this setup where only certain fields show up in the table on the read page, but all the fields are able to show up in the create and update pages using the setup in the config.php file and i have been struggling with this. could you recommend some resources that explain how this works, or at least what this is setup is called so i can do more research on how this works.

Thanks,
Bart

0 Reply

D David Adams Mod 2 months ago

Hello Barton,

Thanks a bunch!

Find the following line in the read.php file (should be two instances):

```
<?php foreach ($columns as $column_key => $column): ?>
```

And replace it with:

```
<?php foreach ($columns as $column_key => $column): ?>
<?php if (!$column['visible']) continue; ?>
```

And then update the columns in the config.php file to include the new "visible" property, like so:

```
'phone' => [
'label' => 'Phone',
'sortable' => false,
'type' => 'string',
'visible' => false,
'input' => [
'placeholder' => 'Phone Number',
'type' => 'tel',
'required' => true,
'validate_msg' => 'Please enter a valid phone number!',
'validate_regex' => '/^[0-9]{8,15}$/',
]
]
```

Make sure to include it in all your columns and set it to either true or false.

0 Reply

B Barton Jetter 2 months ago

thank you for the assistance and the quick response.

one suggestion I would make is to make a short tutorial of how to integrate your scripts together, like the CRUD with the advanced login system. I figured out a way

Bart

0 Reply

D David Adams Mod 2 months ago

You're welcome! With the login and registration system (PDO version), you can add the following to any page you need to restrict access to:

```
<?php
// Include the main.php file
include 'main.php';
// Check if the user is logged in, if not then redirect to login
page
check_loggedin($pdo);
// Template code below
?>
```

Assuming the main.php is in the same folder as your CRUD system or any other script.

And for role based restrictions, you can simply do:

```
if ($_SESSION['account_role'] == 'Admin') {
// Execute if admin
} else {
// Execute if non-admin
}
```

Or in the template code:

```
<?php if ($_SESSION['account_role'] == 'Admin'): ?>
<p>Only admin can see this!</p>
<?php else: ?>
<p>Only non-admins can see this!</p>
<?php endif; ?>
```

0 Reply

S Sven Schulte 9 months ago

Hello David, is it possible to open the update, create and delete forms in a modal view?

BR
Sven

0 Reply

D David Adams Mod 9 months ago

Sure thing!

If you follow the interactive modal guide: https://codeshack.io/interactive-modals-javascript/

You can add the form content to the modal and utilize the JS fetch() API to process the form in the background (AJAX).

0 Reply

L Luca 1 year ago

How can I link a search engine to this that filters all data?

0 Reply

L Luca 1 year ago

How can I ensure that as soon as I create a new contact the ID immediately retains a consecutive number?

can you show me the code?

0 Reply

D David Adams Mod 1 year ago

The MySQL AUTO_INCREMENT option and PRIMARY KEY will ensure a consecutive

G Greg McNamara 1 year ago

Hi David,

I'm using the dynamic... and it's amazing! I'm learning so much.

I added a column (Local) in the config.php file and when I'm on the update page, I would like it to be a select from a row in a separate table rather than static input values.

So I created this to grab the data from the table to create the select list:

$array = $stmt1 -> fetchAll(PDO::FETCH_ASSOC);

foreach($array as $row) {

$LocalOptions[] = $row['Local_Number'];
}
$LocalOptions = implode("', '", $LocalOptions);
$LocalOptions = "'".$LocalOptions."'";

Then under my 'Local' column, where it says options, I placed this:

'options' => [$LocalOptions],

However, it's not catching my array. The resulting html output is:

```
<select id="Local" name="Local" required="">
<option value="'00034', '00041', 'Y0104' , 'Y0104'">'00034', '00041', 'Y0104'
</option>
</select>
```

I'm thinking it's my syntax?

Or is there a smarter way/example to pull values from a table to populate the options array?

Thanks! Greg

0 Reply

D David Adams Mod 1 year ago

Thanks, Greg! It's great to hear you're satisified with the advanced package!

You can add the array returned from your query directly to the options:
'options' => $array,

And then iterate that array with a foreach to populate your option elements:

```
<?php foreach ($array['options'] as $opt): ?>
<option value="<?=$opt['Local_Number']?>"><?=$opt['Local_Number']?></opti
<?php endforeach; ?>
```

0 Reply

N Networking Disqus 1 year ago

David,

I began reviewing the CRUD application purchased as part of the bundle. It seems like the app is older since I do not see PDO statements like the advanced registration package. Additionally, I would expect to see the use of POST verbs and not GET. To me, the GET verb just opens the system to data leakage. I'll further look at integrating the advanced registration system to authorize the usage (saw you made some previous suggestions).

Can we tie in the CSRF and XSS functionality?

A feature request would be to further present related table grid views. The classic example would be an invoice header then the secondary line items grid based on the selected grid header value.

Thanks.

0 Reply

D David Adams Mod 1 year ago

Thanks a bunch, Networking Disqus!

The GET request variables are required for functionalities such as column sorting, pagination, etc. It's the most convenient and optimal solution for such purposes. I've added function to prevent XSS.

It's recommended to use the CRUD system behind a secure authentication system like the secure login + registration system that we have on here, as it will prevent unauthorised users from using it.

CSRF is relatively easy to add to the system - you can basically copy the code from the login + registration CSRF protection user guide and place the code inside the create, edit, delete forms, etc.

Thanks for the suggestions! I'll consider them when working on a new update!

0 Reply

M Max 1 year ago

How can we add <select> create a drop-down list, for "title" column ? I know there is filter for everything, but what is the possibility of creating a list from column "title" and filter records by value only from this column ?

0 Reply

D David Adams Mod 1 year ago

Hello Max, Which version are you using?

0 Reply

M Max 1 year ago

Bought 2 days ago, so should be the newest.
I see in log that there should be:

> [Added] dropdown filter list with from and to date filters.

But in download files there isn't any dropdown filter.

0 Reply

M Max 1 year ago

Ohh you think about dropdown filter when i update or create new record. But I mean about dropdown filter on main screen, that I could search all record with specific "title". Dropdown menu for search only one specific column, example "title"

0 Reply

D David Adams Mod 1 year ago

The filters are on the read.php page. If you click the filter icon next to the search element, you'll see the dropdown box containing the date range filters.

I believe the "title" is already a dropdown select element. If you want to add a new filter, you can add:

```
<label for="title">title</label>
<select id="title" name="title">
<option value="">Select Title</option>
<option value="Employee">Employee</option>
<option value="Manager">Manager</option>
</select>
```

To the dropdown element and declare a variable to retrieve the value:

```
$title = isset($_GET['title']) ? $_GET['title'] : '';
```

And append it the $where_sql variable if a value is specified:

```
if (!empty($title)) {
$where_sql .= ($where_sql ? ' AND ' : ' WHERE ') .  ' title <= :ti
}
```

And finally bind the value to both queries:

```
}
```

If you look for the $to_date and $from_date variables, you'll see where to declare the code.

0 Reply

F

Fred Bonani 1 year ago

David, is there any code that I can add to the create, update and delete files that would prevent anyone other than me from accessing those pages? I realize I don't have to display those links, but would prefer to. Thanks in advance.

0 Reply

D

David Adams Mod 1 year ago

You could integrate the login + registration system and add the following:

```
check_loggedin($con)
```

To prevent direct access.

Or implement a secret code param to the URI:

```
if (!isset($_GET['code']) || $_GET['code'] != 'secret_code') {
    exit('Invalid code!');
}
```

But that means you'll have to append it to every URL (?code=secret_code).

0 Reply

F

Fred Bonani 1 year ago

Where would the

```
if (!isset($_GET['code']) || $_GET['code'] != 'secret_code') {
exit('Invalid code!');
}
```

go in the update file as an example. I would rather not use a login if I don't have to and I expect that the code param would only have to be entered in three files.

0 Reply

D

David Adams Mod 1 year ago

Preferably after the opening tag (<?php).

0 Reply

F

Fred Bonani 1 year ago

Thanks, I think I got it working. I just have to beautify it.

0 Reply

R

Ray 2 years ago

Hello, i'm having some trouble with editing the script. Why does nothing changes when i edit the style.css, even after i delete everything inside style.css, it's still look the same.
This is most likely a pretty stupid question but i still would like to know why?
Thanks.

0 Reply

D

David Adams Mod 2 years ago

need to clear the cache for your development site for the changes to take effect. Holding down SHIFT while pressing F5 should clear the cache. If that doesn't work, refer to your browser documentation on how to clear the cache for a particular website.

It's a common question, so I wouldn't worry about it.

0 Reply

C CarlosR 2 years ago

Hello David!
I bought the advanced version and I have to say its simple and great! Nice job!
But I need some more options and functions for a project. Perhaps you can help me...

1. How can I create fixed options to select in the create-form/update-form and limit them?
2. Is it possible to put icons/png in the table and select them in the create-form or update-form?
3. How can I calculate a sum from a column of the table and display it below the table?
4. The csv upload example didn't work for me. Could you describe it a little bit more?

Thanks.

0 Reply

D David Adams Mod 2 years ago

Hi,

Thank you for purchasing the advanced package!

Regarding your questions, please see below.

1) Are you referring to radio buttons?

2) You would have to implement a new form element to upload files:

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

And the code to upload the file:

```
if(isset($_FILES['image'])){
  $file_name = $_FILES['image']['name'];
  $file_size = $_FILES['image']['size'];
  $file_tmp = $_FILES['image']['tmp_name'];
  $file_type = $_FILES['image']['type'];
  $file_ext = strtolower(end(explode('.',$_FILES['image']['name'])));
  $extensions = ['png'];
  if(in_array($file_ext,$extensions)=== false){
 exit('Please choose a PNG file!');
  }
  $file_dest = 'uploads/' . $file_name;
  move_uploaded_file($file_tmp, $file_dest);
}
```

You can then bind the $file_dest variable to your SQL query, so you store the location of the file in your MySQL table.

Also, to upload files, you need to add:

```
enctype="multipart/form-data"
```

To your form element.

3) You can simply do:

```
$value = $pdo->query('SELECT SUM(column_name) FROM contacts')->fetchColum
echo $value;
```

4) The CSV file must reflect the number of columns in your "contacts" table. Also, the values have to be comma-separated.

0 Reply

Justin Leif 1 year ago

I know i still need a ext check - But if i comment out:
//$file_ext = strtolower(end(explode('.',$_FILES['image']['name'])));
//$extensions = ['png'];
//if(in_array($file_ext,$extensions)=== false){
// exit('Please choose a PNG file!');
//}

I do get a successful file_dest write to the DB :)

I'm trying to integrate the image upload / edit / delete function into this CRUD project. I'm running into a number of issues.

Would you mind showing Create / Update / Delete (.php)
with the image upload feature integrated?
I'll echo out the image on a different page on the website. so, no need to go through the effort of altering read.php, if you don't want to ;)

0 Reply

D DoubleDutch 2 years ago

Hi, i'm having a little trouble with editing the script to my needs. I have added a couple of text fields. I can create a new contact but when I try to update a contact I get the following error. "Integrity constraint violation: 1048 Column 'id' cannot be null". Any idea what this could be? Thank you

0 Reply

D David Adams Mod 2 years ago

If you don't plan on dynamically changing the ID column, you can remove it from the SQL query. See the updated code below.

```
$stmt = $pdo->prepare('UPDATE contacts SET name = ?, email = ?, phone = ?
```

0 Reply

B Bas Martens 2 years ago

Hi, I'm using this script in combination with the "Secure Login System". is it possible to only show the "edit" & "trash" buttons when a certain user is logged in? e.g. only when user "Office" is logged in the buttons are shown. or do you have an other idea?

0 Reply

D David Adams Mod 2 years ago

Sure! Is it based on the username? If so, a simple IF statement will suffice:

```
<?php if (isset($_SESSION['loggedin']) && $_SESSION['username'] == 'Offic
 // html buttons here
<?php endif; ?>
```

0 Reply

B Bas Martens 2 years ago

Thank you, but it throws an error. Undefined array key "username"

0 Reply

D David Adams Mod 2 years ago

Try $_SESSION['name']

0 Reply

B Bas Martens 2 years ago

Yep, that did the trick. Thank you very much

0 Reply

B Bas Martens 2 years ago

First of all, great little application.
I was wondering if you could help me with a minor issue.
On the "read" page the date is formatted as 2019-05-08 17:32:00

D David Adams Mod 2 years ago

You can convert the string to date and remove the seconds:

```
<?=date('Y-m-d H:i', strtotime($contact['created']))?>
```

0 Reply

B Bas Martens 2 years ago

Thanks for the quick reply!
Working as intended :)

0 Reply

F Fred Bonani 2 years ago

In looking at the features, it says that the basic version has a search feature. I can't seem to find it or my eyes are worse than I thought. Can you point out where in the code it is located.

0 Reply

D David Adams Mod 2 years ago

Hi @fredbonani,

The search functionality is on the "read.php" page, located in the top right corner. Perhaps you've mistaken the tutorial version for the basic version?

0 Reply

F Fred Bonani 2 years ago

Evidently I have mistaken the tutorial version for the basic version. Where/how do I get the basic version and is there a fee for it?

0 Reply

D David Adams Mod 2 years ago

The basic version is included with the advanced package.

0 Reply

A Adrian M 2 years ago

Hello, good job! Can you add a filter like - for users to be able to read/update/delete only their own records?
Or mixing to something bigger - add some RBAC - or login/permission system where user have roles and permissions? Or you can split in multiple scripts that can be added togheter for paid version.

regards,
Adrian

0 Reply

D David Adams Mod 2 years ago

Hello,,

Thank you for your suggestions! I'll consider them when working on the next update. I'd suggest you take a look at the advanced login & registration system as it includes roles which you can leverage to achieve what you ask.

0 Reply

A Adrian M 2 years ago

Ok, thank you!

0 Reply

A Adrian M 2 years ago

Hello, the script is updated to PHP 8?

D

It should work with PHP 8.

0 Reply

M

Merle 2 years ago

What is the benefit of python vs php? both integrate with mysql, ajax, etc. I wanted to download your login system but not sure which is better. Can I use php and python together? i.e. python login and php pages?

0 Reply

D

David Adams Mod 2 years ago

Both have their own advantages you can search up. However, if you want to know which I recommend then I'd say the PHP version as I tend to update packages more often and offer more content compared to Python.

I don't recommend mixing Python and PHP pages. I'd choose one over the other to prevent your application from being exposed to vulnerabilities.

0 Reply

C

Chris 3 years ago

Hey David, thank you for the hard work you have done. I went ahead and bought the advanced and edit the code to fit my needs. I was wondering how can I expand the table in read.php as I have added more columes to display things, etc. The Edit/Delete buttons are on top of each other, instead of side by side now. Thank you!

0 Reply

Show More

Search...

Become a Subscriber

Stay up to date and join our newsletter to receive the latest updates.

Email Address

Subscribe

FOLLOW US

RECENT POSTS

20 Essential Python Code Snippets for Every Beginner

How to Convert HTML Tables to CSV Using JavaScript

TOPICS

PHP	29
MySQL	19
JavaScript	15
HTML	5
CSS	6
Python	2

SHARE ON:

CODESHACK.IO

RESOURCES	TOOLS	COMPANY	PACKAGES
Tutorials	Live Code Editor	About Us	PHP
Tools	JSON Sorter	Contact Us	Python
Snippets	CSS Sprite Generator	Resend a Receipt	Node.js
References	Images to Sprite Sheet Generator	License Agreement	PHP Scripts Bundle