ARTIKELVERWALTUNG

Ronnie Luft
2013/2014
TK – PHP/MySQL

1 Artikelverwaltung – Allgemeiner Arbeitsauftrag

Schüler bekommen zuerst einen Einblick, wie die Artikelverwaltung funktionieren soll:
c:/xampp/4DK/Artikel – kurze Besprechung

So soll die Artikelverwaltung aussehen:

- Layout: 2 Spalten – mit YAML erstellt (bzw. von einem bestehenden Beispiel kopiert)
- Linke Spalte: Menü mit einer unordered list und CSS entwickelt
- Rechte Spalte: Zeigt an, was aus dem Menü aufgerufen wurde
- Menü: (Artikel) **anlegen, suchen, löschen, ändern**
- In der Abb. 1 sieht man lediglich auf die technische Umsetzung – um Design/Layout habe ich mich hier wenig gekümmert, da es ja hauptsächlich um PHP mit Anbindung auf mySQL-Datenbank geht.

Diese Angabe soll euch Schülerinnen/Schülern dienen um unter Anleitung die PHP Anbindung an eine Datenbank selbständig zu realisieren. Ziel dieser Angabe ist, dass ihr euer eigenes Tempo findet und ich, als Lehrer als helfende Hand mitwirke und zur Seite stehe. Also liebe Schüler, findet euer Tempo, gruppiert euch (2 – 3 Schüler pro Gruppe) – helft euch, ergänzt euch. Es ist AUSDRÜCKLICH von meiner Seite NICHT gestattet in der Gruppenarbeit Themen, die NICHT zum Arbeitsauftrag passen, zu besprechen! Und nun zum Aufbau:

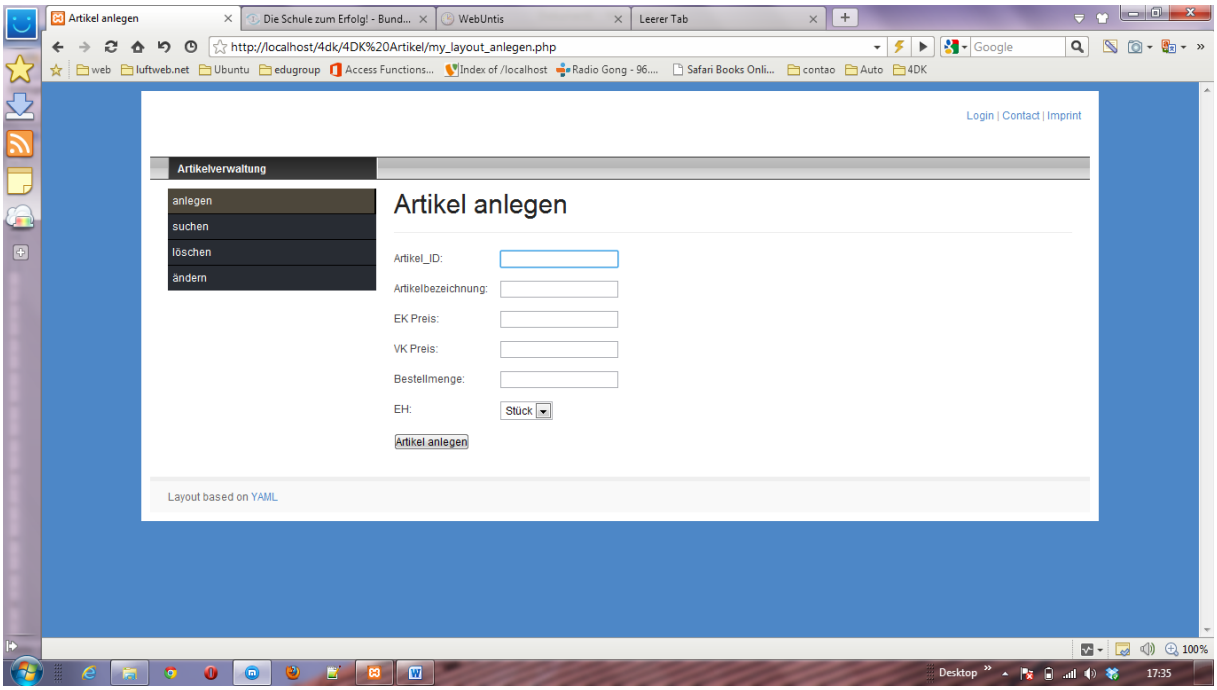


Abb. 1: Artikel anlegen

2 Layout

Beginnt euer Projekt mit dem Layout. Wie bereits oben beschrieben empfehle ich ein bestehendes YAML-Framework zu kopieren und einen neuen Ordner in den c:\xampp\htdocs\ Verzeichnis zu erstellen. Beispiel **c:\xampp\htdocs\Artikel.**

- Benennt das my_layout.html auf **my_layout_anlegen.php** und erstellt hier in der ersten Spalte mit einer unordered list ein vertikales Menü. Nehmt euch kurz Zeit dafür Zeit – maximal 10 min. – bitte nicht zu lange – wir dürfen unser Ziel – PHP/MySql nicht aus den Augen verlieren (CSS – schlage ich vor als HÜ zu erledigen)
- Verlinkt bitte nun im Menü zu den (noch nicht existierenden) Dateien –

```
<div id="col1">
  <div id="col1_content" class="clearfix">
    <!-- add your content here -->
    <div id="nav_left">
        <ul>
          <li class="active"><a>anlegen</a></li>
          <li><a href="my_layout_suchen.php">suchen</a></li>
          <li><a href="my_layout_loeschen.php">l&ouml;schen</a></li>
          <li><a href="my_layout_aendern.php">&auml;ndern</a></li>
        </ul>
    </div>
  </div>
</div>
```

  Erklärung:
  nav_left … ID zur CSS-Gestaltung für das vertikale Menü. Zuerst einmal mit list-style-type:none die Aufzählungszeichen ausblenden. Alle weiteren Eigenschaften für die Menü-Formatierung in gewohnter Weise umsetzen (wie gesagt – max. 10 min. – dann HÜ)

- Das Grundgerüst für unsere HP ist nun gelegt – wir brauchen aber auch weitere Dateien für *suchen, ändern und löschen*. Die bestehende Datei *my_layout_anlegen.php* bitte kopieren und einmal in **my_layout_suchen.php, my_layout_aendern.php** und **my_layout_loeschen.php** umbenennen.
- Ihr habt nun vier identische Dateien – jetzt müsst ihr jede einzelne Datei bearbeiten. Bitte beginnt mit dem Artikel anlegen – my_layout_anlegen.php

3 Artikel anlegen

In diesem Abschnitt geht es nun darum der Artikelverwaltung Leben einzuhauchen. Dazu benötigen wir die Datenbankstruktur, die letztendlich die Daten speichern wird. Danach werden wir mit einem Formular die Daten, die in die Datenbank geschrieben werden, erstellen.

3.1 Datenbank-Struktur

Zum Erstellen der Datenbank verwenden wir phpMyAdmin und nennen die Datenbank db_entw. Die erste Tabelle, die wir erstellen heißt tl_artikel mit folgender Struktur.

Hinweis:

- Die Installation von XAMPP, über die der Zugriff auf phpMyAdmin geht, habe ich auf www.schule.at Teil I: XAMPP http://www.schule.at/portale/wirtschaftsinformatik/detail/xampp.html bereits beschrieben.
- Ebenso die Verwendung bzw. das Anlegen einer Tabelle in phpMyAdmin: http://www.schule.at/portale/wirtschaftsinformatik/detail/teil-ii-mysql.html)
- sowie die Beschreibung der Datentypen http://www.schule.at/portale/wirtschaftsinformatik/detail/mysql-datentypen.html

Die Tabellenstruktur hat folgenden Aufbau:

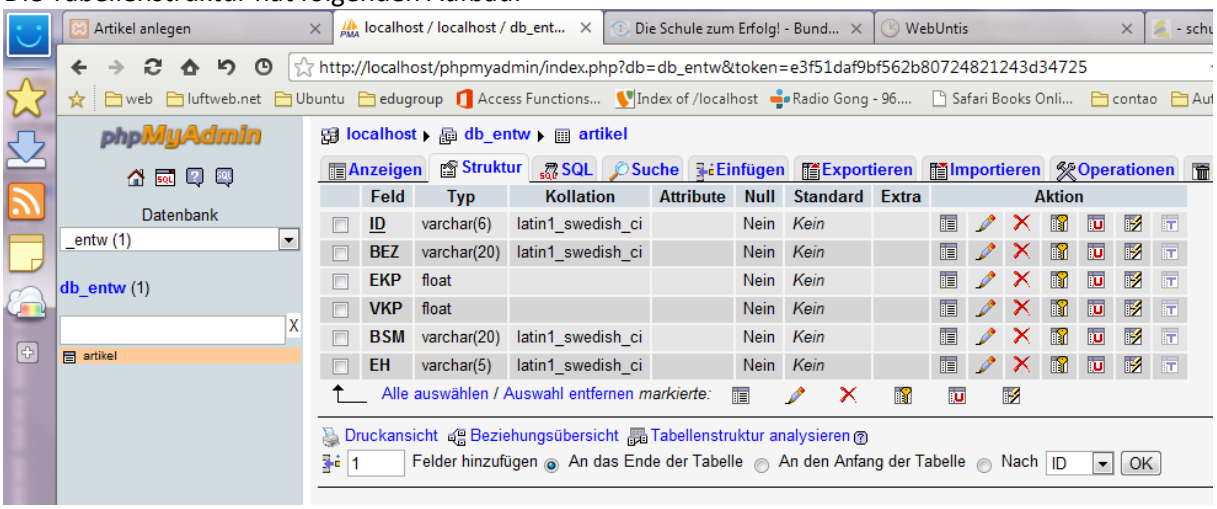


Abb. 2: Struktur der Tabelle tl_artikel

Nach dem Erstellen der Tabelle müssen noch Daten eingegeben werden. Die Eingabe der Artikel werden wir über unsere Homepage pflegen.

4 Artikel anlegen – my_layout_anlegen.php

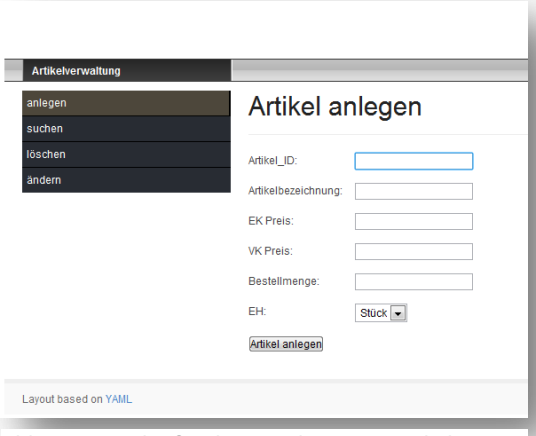


Abb. 3: Formular für die Eingabe eines Artikels

Wir müssen zwei Dinge beachten:

1. In der rechten Spalten (im Quellcode – col3) müssen wir ein Formular erstellen, das die Werte, die in der Datenbank gespeichert werden sollen (also die Artikel) übernimmt.
2. Unterhalb des Formulars müssen wir mit dem PHP-Teil beginnen und zuerst prüfen, ob der User Daten in das Formular eingegeben hat. Wenn ja, dann müssen wir in PHP die Datenbank ansprechen und die Daten mit einem INSERT INTO … einfügen.
3. Hinweis: Wir kümmern uns im gesamten Arbeitsauftrag NUR um die Basisfunktionen vom Eintragen (INSERT), Löschen (DELETE) und Ändern (UPDATE). Wir werden Sonderfälle vorerst ignorieren und diese erst nach der „erfolgreichen“ Implementierung näher unter die Lupe nehmen – zuerst soll die Basis funktionieren und wir werden als Test-User vorläufig nur „saubere“ Daten eingeben, löschen bzw. ändern.

4.1 Formular erstellen

Die Erstellung Feldbezeichnung (Artikel_ID) + Eingabefeld kann, so wie es in Abb. 2 dargestellt ist, nur mit CSS entwickelt werden. Wer Zeit sparen möchte, der schreibt zuerst die Feldbezeichnung und darunter das Eingabefeld (input type) – die CSS-Formatierung ist dann, wie gehabt, als HÜ.

Beim Formular müsst ihr im action="…" das Formular selbst aufrufen. (Wenn auf den Button-Submit gedrückt wird, soll in action="xyz.html/php" jene Datei angegeben werden, die nach dem Klick aufgerufen wird. In dieser Datei werden die Daten aus dem Formular mit $_POST oder $_GET verarbeitet – bei uns: in der Datenbank eingetragen).

- <form action="<?echo $_SERVER['PHP_SELF']; ?>" method="POST">
- Hinweis: Ihr müsst bei jedem Feld einen Namen vergeben, um das Feld später mit $_POST['NAME'] ansprechen zu können: Artikel_ID: <input type="text" name="**ID**">

4.2 PHP Coding INSERT INTO

Gleich nach dem Formular müsst ihr das PHP-Coding ansetzen. Prüft zuerst, ob im Feld Artikel_ID ein Wert eingegeben wurde. Diese Überprüfung sollten wir eigentlich auf alle eingegebenen Felder machen – werden das aber erst später vervollständigen. Fürs Erste sollte die Überprüfung auf Artikel_ID reichen.

```
if (!empty($_POST['ID'])
{
 dann wurde eine ID eingegeben und wir möchten die vom Formular übergebenen Werte
 in die Datenbank eintragen.
}
else
{
 Fehlerausgabe: „Eingabe unvollständig!"
}
```

4.2.1 Datenbank verbinden – Daten eintragen

In diesem IF-Zweig gestalten wir nun den Aufbau zur Datenbank. Folgende Punkte müssen wir beachten:

- Verbindung (connect) zur Datenbank aufbauen
- Einen SQL-String ($sql) aufbauen, der das nötige SQL als String speichert
- Den SQL-String als Query in der Datenbank absetzen
- Die Datenbank wieder schießen

4.2.2 Verbindung zur DB aufbauen

www.php.net – suchen nach mysqli, danach myslqi_fetch_assoc – das Coding kopieren und in den IF-Zweig eintragen:

Example #2 Procedural style

```
1 <?php
  $link = mysqli_connect("localhost", "my_user", "my_password", "world");

  /* check connection */
2 if (mysqli_connect_errno()) {
      printf("Connect failed: %s\n", mysqli_connect_error());
      exit();
  }

3 $query = "SELECT Name, CountryCode FROM City ORDER by ID DESC LIMIT 50,5";

4 if ($result = mysqli_query($link, $query)) {

      /* fetch associative array */
      while ($row = mysqli_fetch_assoc($result)) {
5         printf ("%s (%s)\n", $row["Name"], $row["CountryCode"]);
      }

      /* free result set */
6     mysqli_free_result($result);
  }

  /* close connection */
7 mysqli_close($link);
  ?>
```

The above examples will output:

```
Pueblo (USA)
Arvada (USA)
Cape Coral (USA)
Green Bay (USA)
Santa Clara (USA)
```

Abb. 4: Allgemeines Coding für den Datenbankzugriff aus php.net

4.2.2.1 *mysqli_connect ("localhost", "my_user", …)*

Ad 1 – Hier wird die Verbindung zur Datenbank in phpMyAdmin hergestellt.

- localhost: hier wird der Datenbank-Server eingetragen. Ihr braucht hier wirklich den Namen (unter Anführungszeichen) "localhost".
- my_user: Auf einer Datenbank können mehrere User arbeiten. Wir brauchen den User "root".
- my_password: einfach einen Leerstring unter zwei Anführungszeichen "" übergeben. (Für den User "root" ist kein Passwort in der Datenbank gespeichert.
- world: diesen Parameter einfach weglassen – brauchen wir nicht.

4.2.2.2 *mysqli_connect_error()*

Ad 2 – Sollte bei der Verbindung etwas schiefgegangen sein, dann wird die dazugehörige Fehlermeldung ausgegeben.

4.2.2.3 $query

Ad 3 – Mit der Query wird das SQL für die Datenbank beschrieben. Wir möchten einen Artikel in die Datenbank einfügen. Einfügen in die Datenbank geht mit dem Schlüsselwort INSERT. Am besten gebt ihr in phpMyAdmin einen Artikel beim Karteireiter "einfügen" einmal ein und kopiert dann den erzeugten SQL-String:

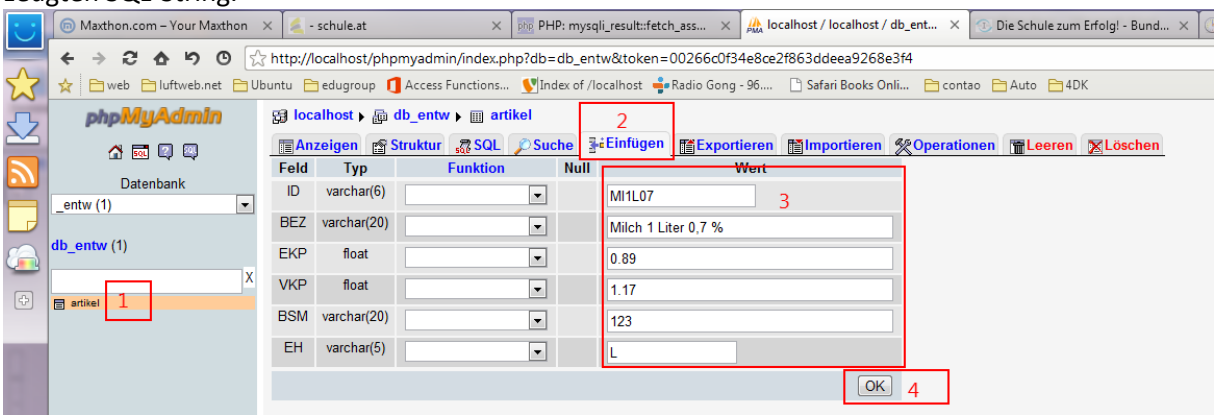


Abb. 5: Artikel in phpMyAdmin einfügen

1. Tabelle anklicken
2. Einfügen auswählen
3. Daten eingeben
4. OK

phpMyAdmin erstellt diesen Artikel und fügt die Daten mit SQL in die Datenbank ein. phpMyAdmin zeigt uns sogar, wie es das gemacht hat: Nach dem OK-Button bekommen wir folgende Ansicht:

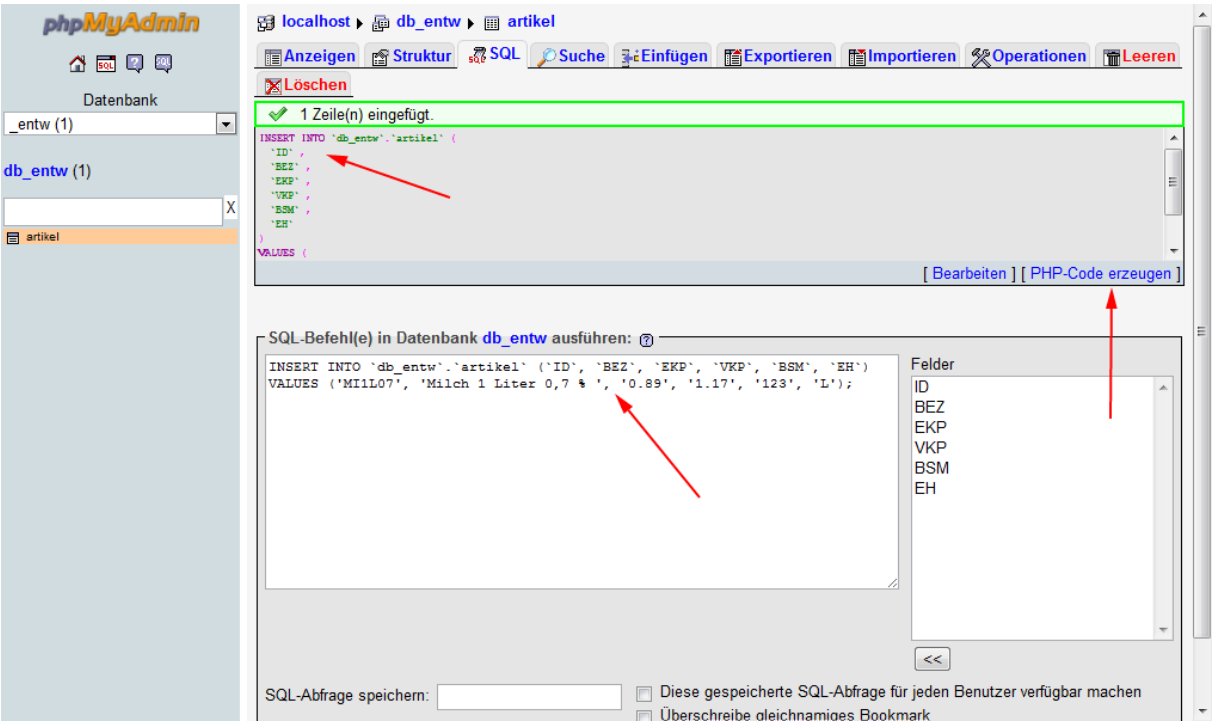


Abb. 6: SQL-Statement für INSERT INTO

Dieses INSERT INTO … kann man nun bequem kopieren und im PHP-Code als Basis für $query verwenden.

So schaut der fertige String für $query aus:

```
$query = "INSERT INTO `db_entw`.`artikel` (`ID`, `BEZ`, `EKP`, `VKP`, `BSM`, `EH`)
                    VALUES ('".$id."','".$bez."','".$ekp."','".$vkp."','".$bsm."','".$eh."');";
```

Achtung: die Hochkommata haben eine Bedeutung.

4.2.2.4 Einfache Hochkomma mit shift + Hochkomma + Leertaste

Bei `db.entw`.`artikel` sind dies einfache Hochkomma, die auf der PC-Tastatur über shift + Hochkomma (rechts oben, neben der Löschtaste) + Leertaste erzeugt werden. Sie zeigen phpMySql lediglich, dass es sich hier um eine Datenbank (db_entw) bzw. um eine Tabelle (artikel) bzw. um einen Feldnamen (z. B. ID) handelt.

4.2.2.5 Einfache Hochkomma mit shift + Raute-Taste

Diese Hochkomma schließen immer den Wert ein, der in die Tabelle eintragen werden soll.

4.2.2.6 Der String $query

Im $query fällt auf, dass im VALUES die Werte von $id, $bez, $ekp, … eingetragen wurden. Das sind lediglich Variablen, denen zuvor mit $id = $_POST['ID'] der Wert aus dem Formular zugewiesen wurde. Also bitte nicht vergessen – diese Werte für alle Formularfelder zuzuweisen.

Wenn ihr die $query fertiggebastelt habt, dann versucht, bevor ihr mit dem Coding weitermacht, diesen String auszugeben

echo $query;

und kontrolliert dann die Ausgabe mit dem String vom phpMyAdmin. Die Ausgabe muss gleich sein dem String vom phpMyAdmin. (Natürlich können andere Werte nach VALUES darinstehen, weil der User andere Werte im Formular eingegeben hat …)

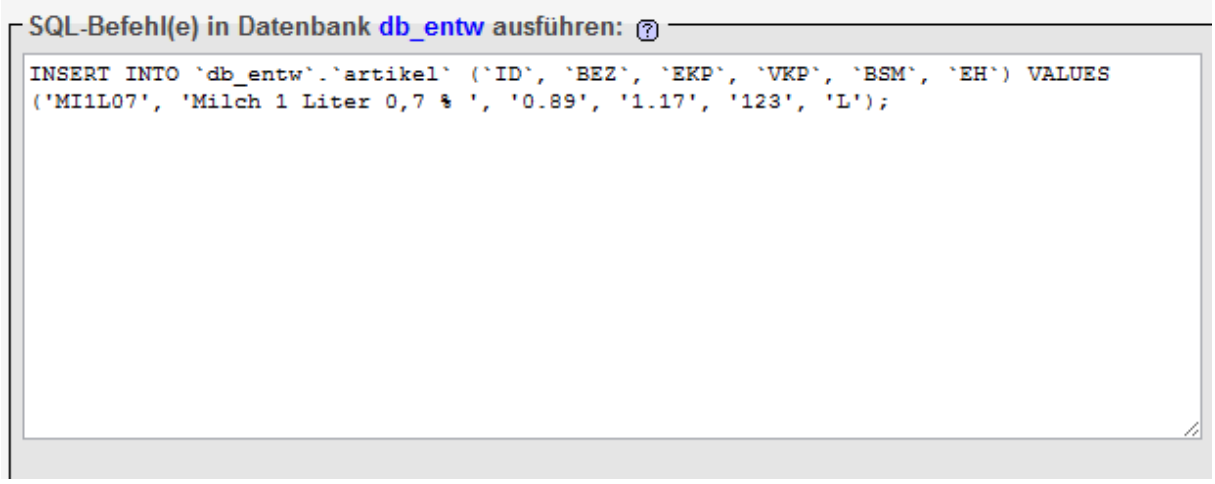


Abb. 7: SQL INSERT INTO phpMyAdmin

4.2.2.7 mysqli_query/mysqli_fetch

```
if ($result = mysqli_query($link, $query)) {

    /* fetch associative array */
    while ($row = mysqli_fetch_assoc($result)) {
        printf ("%s (%s)\n", $row["Name"], $row["CountryCode"]);
    }

    /* free result set */
    mysqli_free_result($result);
}
```

Abb. 8: mysqli_query aus php.net

Ad 4 und 5 – mysql_query braucht als Übergabeparameter den $link und die $query. Mit dem $link kann mysqli_query die Datenbank und die Tabelle finden und mit $query wird das INSERT INTO an diese Datenbank abgesetzt. Der Rückgabewert $result liefert dann zeilenweise das Ergebnis aus der Datenbankabfrage. Da wir hier ein INSERT absetzen, bekommen wir auch keine Datensätze zurück und die Abfrage auf $result (in der Angabe Punkt 6) kann somit wegfallen:

Für ein INSERT INTO reicht:

```
/* SQL-Statement ausführen */
if (mysqli_query($link, $sql) === TRUE) {
  echo "<div style=\"color:red;\">Artikel angelegt</div>";
}
else
{
  echo "mysqli_query failed ...".mysqli_error($link);
}
```

Abb. 9: SQL-Statement für die Artikelverwaltung

4.2.2.8 mysqli_close($link)

Ad 7 – Nachdem erfolgreichen INSERT wird die Datenbank wieder sauber geschlossen.

5 Artikel löschen – my_layout_loeschen.php

Nachdem ihr die Funktionalität zum Anlegen eines Artikels erledigt habt, werden wir die Funktionalität zum Löschen eines Artikels umsetzen.

Abb. 10:

Die Vorgehensweise zum Löschen ist (fast vollkommend) identisch zum Anlegen. Die Unterschiede liegen darin, dass das Coding im my_layout_loeschen.php implementiert wird und, dass beim SQL statt dem INSERT INTO ein DELETE FROM umgesetzt wird.

5.1 Vorgehensweise

- Änderungen in my_layout_loeschen.php vornehmen
- Formular mit nur einem Feld (Artikel_ID)
- PHP, das statt dem INSERT INTO ein DELETE FROM absetzt

5.2 Formular erstellen

Erstellt bitte ein Formular, in dem der USER die Artikel_ID vom Artikel, der gelöscht werden soll übergebt.

5.3 PHP Coding DELETE FROM

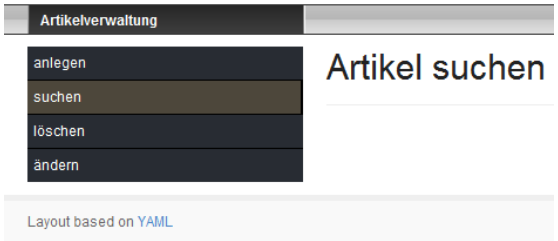
Wie im Beispiel zum Anlegen eines Artikels, wird PHP nach dem Formular im Coding umgesetzt. Ihr müsst, wie auch vorher schon, zur Datenbank connecten und ein neues $query zusammenbauen:

```
$query = "DELETE FROM `db_entw`.`artikel` WHERE `artikel`.`id` = '".$id."'";
echo $query;
```

Abb. 11: SQL – DELETE FROM

An die Datenbank soll die $query abgesetzt werden. Dabei wird jener Artikel gelöscht, der der $id entspricht. $id ist wieder eine Variable, die mit dem Wert aus $_POST['ID'] belegt wurde. ACHTUNG: hier ist eine besonders beachtenswerte Fehlerquelle: Wir müssen sicherstellen, dass der USER wirklich genau diesen Artikel löschen will. Ein Rückgängigmachen vom gelöschten Artikel ist nicht möglich! Diesen Spezialfall werden wir uns aber erst nach der Basisentwicklung genauer ansehen. Vorläufig gehen wir davon aus, dass der USER Herr seiner eigenen Geistigenkräften ist und sehr wohl weiß, welchen Artikel er löscht will.

6 Artikel suchen – my_layout_suchen.php



In Arbeit

7 Artikel ändern – my_layout_aendern.php

In Arbeit