

Inhalt

Inhalt	I
Abbildungsverzeichnis	III
Abkürzungsverzeichnis	IV
1 Aufgabenstellung	1
1.1 <i>Inhalte des Front-End Bereich</i>	<i>1</i>
1.2 <i>Inhalte des Back-End Bereich.....</i>	<i>1</i>
2 Problemanalyse	3
2.1 <i>Speichern und abrufen der Daten.....</i>	<i>3</i>
2.2 <i>Beitrag erstellen</i>	<i>3</i>
2.3 <i>Löschen von Beiträgen</i>	<i>3</i>
2.4 <i>Bildupload.....</i>	<i>3</i>
2.5 <i>Benutzer erstellen</i>	<i>3</i>
2.6 <i>Login und Logout.....</i>	<i>4</i>
2.7 <i>Menüpunkte erstellen.....</i>	<i>4</i>
3 Lösungskonzeption	5
3.1 <i>Speichern und abrufen der Daten.....</i>	<i>5</i>
3.2 <i>Beitrag erstellen</i>	<i>5</i>
3.3 <i>Löschen von Beiträgen</i>	<i>5</i>
3.4 <i>Bildupload.....</i>	<i>5</i>
3.5 <i>Benutzer erstellen</i>	<i>6</i>
3.6 <i>Login und Logout.....</i>	<i>6</i>
3.7 <i>Menüpunkte erstellen.....</i>	<i>6</i>
4 Programmentwurf	7
5 Details der Implementierung.....	8
5.1 <i>Anzeigen der Front-End Beiträge und Pagination.....</i>	<i>8</i>

II		Inhalt
5.2	<i>Löschen von Beiträgen.....</i>	9
5.3	<i>Bildupload.....</i>	10
5.4	<i>Benutzer erstellen.....</i>	10
5.5	<i>Sicherheit.....</i>	11
5.6	<i>Menü.....</i>	12
6	Fazit.....	14
7	Bedienungsanleitung.....	15
7.1	<i>Einleitung.....</i>	15
7.2	<i>Benötigte Software</i>	15
7.3	<i>Installation der Software.....</i>	15
7.4	<i>Login</i>	17
7.5	<i>Beiträge erstellen.....</i>	18
7.6	<i>Beitrag bearbeiten</i>	18
7.7	<i>Beitrag löschen.....</i>	20
7.8	<i>Benutzer registrieren</i>	20
7.9	<i>Abmelden.....</i>	21
	Literatur und verwendete Hilfsmittel	23
	Anlagen	
	XXV
	Selbstständigkeitserklärung.....	27

Abbildungsverzeichnis

Abbildung 1 – So sollte die Konfiguration aussehen.	16
Abbildung 2 - Login Seite	17
Abbildung 3 - Beitrag auf der Startseite	18
Abbildung 4 - Auflistung aller Beiträge	19
Abbildung 5 - Einzelne Bilder löschen.....	19
Abbildung 6 - Nutzer Registrierung	20

Abkürzungsverzeichnis

URL	Uniform Resource Locator
DBMS	Datenbankmanagementsystem
PDO	PHP Date Objects
WYSIWYG	What You see is what you get

1 Aufgabenstellung

Die vorgelegte Arbeit beschäftigt sich mit der Entwicklung eines Content-Management-Systems (CMS) Prototypen in einer objektorientierten Programmiersprache in Kombination mit einem Datenbankmanagementsystem.

Viele CMS wie z.B. „Typo3“ oder „Joomla“ sind für den Laien nur durch intensive Einarbeitung mit Hilfe besonderer Literatur oder Besuchen spezieller Kurse effektiv nutzbar.

Das in dieser Arbeit anvisierte Ziel dieses Content-Management-Systems ist es, dem Endnutzer eine einfache Möglichkeit zur Erstellung eines eigenen Web-Blogs zu bieten. Der Blog ist dabei in zwei Teile aufgeteilt. Den für jeden sichtbaren Front-End Bereich und den für den Administrator sichtbaren Back-End Bereich.

Das in diesem Projekt verwendete Design ist für den „Verein zur Erhaltung und Betreuung des Volksplatzes Borna e.V.“ angepasst.

1.1 Inhalte des Front-End Bereich

Der Front-End Bereich ist für jeden Nutzer sichtbar. Im Prototyp sollen die Kernfunktionen eines Blogs implementiert sein. Wichtig neben dem Design ist vor allem die Anzeige der Überschrift und Vorschau eines Beitrages sowie auch die komplette Ansicht des ausgewählten Beitrages inklusive hinterlegtem Bildmaterial. Die Hinterlegte „Menü“-Funktion im Kopfteil des Web-Blogs ist dabei lediglich ein Prototyp ohne relevante Funktion.

Der gesamte Inhalt des Blogs wird dabei aus einer Datenbank geladen.

1.2 Inhalte des Back-End Bereich

Der Back-End Bereich ist nur für den Administrator über die URL „/admin“ erreichbar. Eine Verlinkung über den Front-End Bereich existiert nicht. Wenn der Nutzer nicht eingeloggt ist, wird er auf die Login Seite weitergeleitet. Nach dem erfolgreichen Login befindet sich der Anwender im Back-End Bereich und kann über ein Menü Beiträge erstellen, bearbeiten und auch löschen sowie weitere Accounts für andere Nutzer erstellen.

2 Problemanalyse

Der englische Begriff Content-Management-System, kurz CMS, bezeichnet eine Software, die es mehreren Nutzern ermöglicht, gemeinsamen Content zu erstellen, zu bearbeiten und zu managen. [1]

2.1 Speichern und abrufen der Daten

Der Kern eines Content-Management-Systems ist der „Content“. Um Inhalte zu sichern und wieder abzurufen soll eine effiziente und dauerhafte Ablagemöglichkeit verwendet werden, diese Daten abzulegen. Diese soll unkompliziert in die Software implementiert werden. Abgerufene Daten sollen richtig formatiert wiedergegeben werden.

2.2 Beitrag erstellen

Um Inhalte eintragen zu können, soll eine unkomplizierte Möglichkeit implementiert werden Text einzufügen und zu bearbeiten ohne HTML Kenntnisse zu benötigen sowie Bilder in den Beitrag einzufügen.

2.3 Löschen von Beiträgen

Der Inhalt eines Beitrages inklusive Bilder soll von dem Datenspeicher gelöscht werden.

2.4 Bildupload

Bilder sollen über den Web Browser auf den Server hochgeladen und in den Beitrag eingefügt werden. Dabei muss überprüft werden, ob es sich bei der hochzuladenden Datei um ein Bild handelt. Eine Maximalgröße soll dabei berücksichtigt werden.

2.5 Benutzer erstellen

Es soll eine Möglichkeit implementiert werden, zusätzliche Benutzer zu registrieren die sich mit ihren eigenen Zugangsdaten Zugang zum Back-End Bereich verschaffen können. Das Passwort soll nicht im Klartext zum Server gesandt werden.

2.6 Login und Logout

Berechtigte Nutzer sollen Zugang zum Back-End Bereich erhalten. Passwörter sollen nicht im Klartext zum Server gesandt werden.

Bei einem Logout soll der Nutzer keinen Zugriff mehr auf den Back-End Bereich haben.

2.7 Menüpunkte erstellen

Es soll eine Möglichkeit für den Nutzer geben, im Back-End Bereich dem „Hauptmenü“ weitere Menüpunkte hinzuzufügen. Diese Menüpunkte sollen im Front-End Bereich sichtbar sein.

3 Lösungskonzeption

3.1 Speichern und abrufen der Daten

Um Daten, die der Nutzer sichern will auf dem Server zu speichern, wird in dieser Arbeit ein Datenbankmanagementsystem (DBMS) verwendet. Das verwendete System ist das Open-Source DBMS „MariaDB“ welches eine Abspaltung von MySQL ist. Damit das CMS sich mit der Datenbank verbinden kann wird zudem eine Softwarebibliothek benötigt. In dieser Arbeit wird die Bibliothek „PHP Data Objects“ (PDO) verwendet. Im Gegensatz zu „MySQLi“ kann das CMS auch mit anderen DBMS verwendet werden.

3.2 Beitrag erstellen

Damit der Nutzer Beiträge erstellen kann, benötigt er eine Eingabemöglichkeit. In diesem Projekt werden HTML Textareas verwendet. Diese wurden durch den WYSIWYG Editor „TinyMCE“ erweitert. Damit kann der eingegebene Text editiert und formatiert werden. Alle eingegebenen Daten werden dann in der Datenbank gespeichert. Die Tabelle ist dabei unterteilt in Überschrift, Vorschautext und der Beitragstext sowie einer Kategorie und einer Beitrags ID. Kategorien können in diesem Prototyp noch nicht vergeben werden. Bilder können im Anschluss hochgeladen werden. Diese werden in einer gesonderten Tabelle eingetragen.

3.3 Löschen von Beiträgen

Der Beitrag wird komplett aus der Datenbank entfernt und Bilder vom Server und in der dafür vorgesehen Datenbanktabelle gelöscht. Dabei wird die Zeile der Beitrags ID komplett entfernt und nachfolgende Beiträge hinsichtlich ihrer ID um -1 reduziert.

3.4 Bildupload

Bilder können ausschließlich nur während der Erstellung eines Beitrages auf den Server geladen werden. Die ausgewählten Bilder werden vor dem Hochladen geprüft. Dabei wird geprüft um was es sich für eine Datei handelt, ob diese Datei auf dem Server schon vorhanden ist und welche Größe die Datei hat. Zugelassene Dateitypen sind „JPG/JPEG“, „PNG“ und „GIF“. Als Maximalgröße sind 5 Megabyte vorgesehen und die Datei darf nicht schon vorher auf dem Server existieren. Wenn alle Bedingungen erfüllt sind wird das Bild an den Server geschickt und ein Datenbankeintrag in der Bildertabelle erfolgt. Dabei wird jedem Bild die ID des Beitrags zugeordnet.

3.5 Benutzer erstellen

Um Zugang zum Back-End Bereich zu erlangen muss ein Benutzer angelegt werden. Dieser besteht aus einer E-Mail Adresse und einem Passwort. Vor- und Nachnamen können in einer späteren Version implementiert werden. Das Passwort wird vor der Übermittlung an die Datenbank gehasht. Das Passwort muss zur Sicherheit des Nutzers zweimal eingegeben werden. Im Anschluss wird der Benutzer in einer gesonderten Tabelle angelegt.

3.6 Login und Logout

Damit keine Unerwünschten Nutzer zugriff auf Back-End Funktionen erlangen, muss sich jeder Nutzer vorher einloggen. Dazu werden E-Mail Adresse und Passwort abgefragt. Das Passwort wird dabei gehasht und an den Server zum Abgleich gesendet. Sind die eingegebenen Daten korrekt kann der Nutzer die Back-End Funktionen benutzen. Um nicht bei jedem Seitenwechsel die Nutzerdaten erneut abzufragen wird eine Session gestartet. Diese Session wird beim Logout wieder beendet.

3.7 Menüpunkte erstellen

Damit der Nutzer sich im Front-End orientieren kann, soll ein Menü implementiert werden. Dieses „Hauptmenü“ soll neben der Startseite, weitere Menüpunkte erhalten, die gleichzeitig als Kategorien für die Beitrag dienen soll. Dieses Menü soll in dieser Arbeit noch nicht komplett implementiert sein.

4 Programmentwurf

Siehe Anlagen

5 Details der Implementierung

Ein großer Teil der Arbeit besteht aus dem Analysieren von Einggegebenen Texten und deren Eintragung oder Löschung aus der Datenbank. Es gibt aber einige Stellen in diesem Projekt, die nicht ganz einfach zu erstellen waren, auf diese möchte ich näher eingehen.

5.1 Anzeigen der Front-End Beiträge und Pagination

Der Begriff Pagination kann mit "Seitennummerierung" ins Deutsche übersetzt werden und kommt zur Anwendung, wenn auf Websites Inhalte auf mehrere Seiten aufgeteilt werden, beispielsweise bei langen Auflistungen von Produkten in einem Online-Shop. [2]

Alle in der Datenbank gespeicherten Beiträge werden in der „show.php“ geladen und angezeigt. Weil mehrere hundert Beiträge in der Datenbank vorhanden sein und die Darstellung im Webbrowser überladen wirken könnte, wird der Inhalt auf mehreren Seiten dargestellt.

Dazu wird zuerst die höchste Beitrags ID in der Datenbank ermittelt.

```
1. $sql = $db->prepare("SELECT MAX(B_ID) from beitrags");
2. $sql->execute() or die("fehler");
3.
4.
5. while($row = $sql->fetch()){
6.     $sql_erg = $row['MAX(B_ID)'];
7.
8.     $total_post = $sql_erg;
9.
10. }
11.
12.
13. $result_per_site = 3;
```

Die Variable `$total_post` stellt die Anzahl Beiträge in der Datenbank dar. `$result_per_site` ist die Anzahl der anzuzeigenden Beiträge pro Seite.

```
1. $total_pages = ceil($total_post/$result_per_site);
```

Die Variable `$total_pages` ist die Anzahl der Seiten die am Ende angezeigt werden soll. Der numerische Wert wird aufgerundet, damit auch die korrekte Anzahl der Beiträge dargestellt wird.

```
1. $counter = $total_post - ($page * $result_per_site - ($result_per_site));
2. $view_post = $total_post - ($page * $result_per_site);
```

Die Darstellung der Beiträge erfolgt in umgekehrter Reihenfolge. Der Beitrag mit der höchsten ID wird zuerst dargestellt, der mit der niedrigsten zuletzt. Daraus erfolgen zwei Rechnungen. Die Variable `$counter` stellt den Beitrag dar der als erstes auf der Seite angezeigt werden soll, `$view_post` der Beitrag, der zuletzt auf der Seite erscheinen soll.

In einer Schleife werden die Beiträge geladen und angezeigt.

5.2 Löschen von Beiträgen

Es ergibt sich bei der Anzeige der Beiträge nun eine Frage. Was passiert, wenn ein Beitrag gelöscht wird?

Wenn ein Beitrag aus der Datenbank entfernt wird, so ändert sich die höchste Beitrags ID nicht, es sei denn, genau der zuletzt eingetragenen Tabelleneintrag wird gelöscht. Es werden weiterhin die Anzahl der anzuzeigenden Beiträge pro Seite dargestellt. Statt den gelöschten Beitrag zu überspringen und den darauffolgenden Beitrag anzuzeigen, wird der vorherige Beitrag doppelt dargestellt.

Um dieses Problem zu lösen, muss bei der Entfernung eines Beitrages, nicht einfach nur der komplette Tabelleneintrag gelöscht werden, es müssen die darauffolgenden Beiträge hinsichtlich ihrer Beitrags ID aktualisiert werden.

```
1. for($i = $B_ID + 1; $i<=$sql_erg; $i++){
2.     $sql = $db->prepare("UPDATE beitrags SET B_ID = ? WHERE B_ID = ?");
3.     $sql->execute(array($i-1, $i)) or die(" Fehler UPDATE<br>");
4.
5. }
```

`$sql_erg` ist dabei die höchste Beitrags ID in der Datenbanktabelle.

Auch die Bilder die einem Beitrag zugeordnet wurden müssen vom Server entfernt und die entsprechenden Einträge aus der Datenbank gelöscht werden. Hier muss die Bilder-tabelle aktualisieren.

```
1. for($i = $B_ID + 1; $i<=$sql_erg; $i++){
2.
3.     $sql = $db->prepare("SELECT * FROM pictures WHERE B_ID = ?");
4.     $sql->execute(array($i)) or die("Fehler Abfrage Datenbank<br>");
5.
6.     if($sql->fetch()==true){
7.         echo "Bild existiert<br>";
8.
9.         $sql = $db->prepare("UPDATE pictures SET B_ID = ? WHERE B_ID = ?");
```

```
10.     $sql->execute(array($i - 1, $i)) or die("Bilder ID aktuali-
    sieren Fehlgeschlagen<br>");
11.     echo "Bilder Aktualisieren erfolgreich<br>";
12.
13. }
14. else {
15.     echo "Bild existiert nicht<br>";
16. }
17. }
```

Vor der Aktualisierung muss man prüfen, ob überhaupt Bilder dem darauffolgenden Beitrag zugeordnet wurden. Besitzt der darauffolgende Beitrag ein Bild, so wird die Beitrags ID aktualisiert. Sonst wird zum nächsten Beitrag übergegangen.

5.3 Bildupload

Bilder werden während der Beitragserstellung hochgeladen, sofern sie während der Erstellung ausgewählt wurden. Für die Verarbeitung der hochzuladenden Datei ist die Klasse „upload.php“ zuständig.

```
1. $check = getimagesize($picture_tmp);
2. if($check !== false) {
3.     echo "Datei ist ein Bild - " . $check["mime"] . ".";
4.     $uploadOk = 1;
5. }
6. else{
7.     echo "Datei ist kein Bild.";
8.     $uploadOk = 0;
9. }
```

Über die Variable \$check wird mittels der PHP Imagefunktion getimagesize(\$file) geprüft, ob die hochzuladende Datei ein Bild ist. Ist dies der Fall so wird die Variable \$uploadOK auf 1 gesetzt. Diese wird am Ende der Methode picUpload(\$pic,\$pic_tmp) abgefragt. Zuvor werden weitere Abfragen hinlänglich der Name, Größe der Datei und dem Dateityp abgefragt. Es werden nur Dateitypen vom Typ JPG/JPEG, PNG und GIF mit einer maximalen Größe von 5MB zugelassen. Wenn die Datei den gleichen Namen wie eine Datei dem Bilderordner hat, wird sie nicht hochgeladen. Wenn die Variable \$uploadOK den Wert 0 besitzt, wird die Datei ebenso nicht hochgeladen.

5.4 Benutzer erstellen

Das erstellen eines neuen Benutzers wird über die Datei „register.php“ und die Klasse „User.php“ realisiert. Über das Formular werden E-Mail-Adresse und das Passwort eingegeben.

```
1. if(!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
2.     echo 'Bitte eine gültige E-Mail-Adresse eingeben<br>';  
3.     $error = true;  
4. }
```

Über der Funktion `filter_var($var,$filter)` wird mithilfe des Filters „FILTER_VALIDATE_EMAIL“ geprüft, ob es sich bei den eingegeben Text auch um eine E-Mail-Adresse handelt. Wenn dies nicht der Fall ist wird die Variable `$error` auf `true` gesetzt und die Registrierung kann nicht abgeschlossen werden.

Wenn E-Mail-Adresse und Passwort erfolgreich geprüft wurden, wird ein neues Objekt der Klasse „User“ angelegt. Dem werden in seinem Konstruktor die E-Mail-Adresse und das Passwort übergeben. Es wird geprüft ob die E-Mail-Adresse bereits in der Datenbank vorhanden ist. Ist dies der Fall so wird die Registrierung abgebrochen.

```
1. function register( $db){  
2.     $password_hash = password_hash($this->password, PASSWORD_DEFAULT);  
3.  
4.     $statement = $db->prepare("INSERT INTO users (email, password) VALUES (:email, :password)");  
5.     $result = $statement->execute(array('email' => $this->email, 'password' => $password_hash));  
6.  
7.     if($result) {  
8.         echo 'Du wurdest erfolgreich registriert. <a href="login.php">Zum Login</a>';  
9.     } else {  
10.        echo 'Beim Abspeichern ist leider ein Fehler aufgetreten<br>';  
11.    }  
12. }
```

Mithilfe der Methode `register($db)` wird der Nutzer in der Datenbank angelegt. Dazu wird das Passwort mit der PHP Funktion `password_hash($password, $algo)` gehasht. Damit wird das Kennwort nicht in der Datenbank im Klartext gespeichert. Im Anschluss werden E-Mail-Adresse und Passwort in die Tabelle eingetragen.

5.5 Sicherheit

Die Sicherheit bei Webseiten ist wichtig, weil es viele offensichtliche und auch nicht offensichtliche Schlupflöcher gibt um Schadsoftware zu implementieren. Wenn sich der Nutzer im Back-End Bereich einloggt, wird eine Session gestartet.

```
1. $_SESSION['userid'] = $user['id'];
```

```
1. session_start();  
2. if(!isset($_SESSION['userid'])) {  
3.     header("HTTP/1.1 301 Moved Permanently");
```

```
4.     header('Location:login.php');
5.     die('Bitte zuerst <a href="login.php">einloggen</a>');
6.
7. }
```

Wenn keine Session gestartet wurde, wird der Nutzer immer wieder auf die Login Seite geleitet um sich dort einzuloggen.

Ein weiteres Sicherheitsrisiko ist die SQL Injection.

Direkt SQL Command Injection ist eine Technik, wo ein Angreifer SQL Kommandos erstellt oder existierende verändert, um versteckte Daten sichtbar zu machen, wertvolle Daten zu überschreiben, oder sogar gefährliche Kommandos auf Systemebene des Datenbank-Hosts auszuführen. [3]

Diese Probleme treten häufig im Front-End Bereich auf und müssen verhindert werden. Im Front-End Bereich dieser Arbeit befinden sich zwei offensichtliche Möglichkeiten SQL Befehle über die Superglobale Variable `$_GET` anzuwenden. Diese Schwachstellen befinden sich in der „show.php“ und „showpost.php“.

```
1. if(isset($_GET["b_id"])){
2.     $B_ID = $_GET["b_id"];
3.
4.     if(is_numeric($B_ID) == false){
5.         die("ERROR: Kein Beitrag angegeben");
6.     }
7.
8. }
```

Da es sich nur um Zahlen handelt, die übergeben werden, kann man mit der PHP Funktion `is_numeric($var)` prüfen, ob es sich bei der Variablen um eine Zahl oder einen numerischen String handelt. Wenn es sich bei der übergebenen Variablen nicht um eine Zahl handelt, wird der Vorgang abgebrochen und es kommt zu keiner Datenbankabfrage.

Eine andere häufig genutzte Schwachstelle ist das Cross-Site-Scripting. Dabei wird versucht JavaScript über verschiedene Eingabemöglichkeiten in die Webseite einzubetten. Da keinerlei Textfelder, Textboxen oder ähnliche Elemente vorhanden sind, kann kein Cross-Site-Scripting angewandt werden. Elemente wie Textfelder oder Textboxen zum suchen oder kommentieren von Beiträgen sollen aber in Zukunft hinzugefügt werden, daher ist es notwendig dieses Thema weiter zu verfolgen und Präventivmaßnahmen zu einem späteren Zeitpunkt zu implementieren.

5.6 Menü

Das Menü wird nur im Front-End Bereich der Arbeit dargestellt. Dazu wird das in der Datenbank hinterlegte Menü geladen. Erstellt werden kann das Menü im Back-End Bereich. In diesem Status können bisher nur der Name und der Rang des Menüpunktes eingetra-

gen werden. Ausführbar ist das Menü im Front-End Bereich für den Anwender noch nicht. Dies soll zu einem anderen Zeitpunkt fertiggestellt werden.

Die Klasse „menuItem.php“ erstellt die einzelnen Menüpunkte und trägt diese in die Datenbank ein.

```
1. public function checkRank($menuItem1, $menuItem2){
2.     if ($menuItem1 == $menuItem2){
3.         die("Fehler: Rank zwischen $menuItem1 und $menuItem2 ist gleich!");
4.     }
5. }
6. function rank($database, $rankItem, $rank){
7.     $sql_abfrage = $database->prepare("SELECT M_SubMenu, M_Rank
8.     FROM menu");
9.     $sql_abfrage->execute() or die("Fehler: rank()");
10. while($row = $sql_abfrage->fetch()){
11.     $serg = $row['M_Rank'];
12.     $rankItem->checkRank($serg, $rank);
13. }
14.
15. }
```

Bevor ein Menüpunkt in die Datenbank eingetragen werden darf, wird vorher geprüft ob ein vorhandener Menüpunkt denselben Rang wie der einzutragende Menüpunkt besitzt. Ist dies der Fall, wird der Vorgang abgebrochen.

Das Menü wird durch die Funktion loadMenu(\$database) in der „menu.php“ im Front-End Bereich geladen.

```
1. function loadMenu($database){
2. $sql_abfrage = $database->prepare("SELECT M_Name, M_SubMenu, M_Rank
3. FROM menu ORDER BY M_Rank");
4. $sql_abfrage->execute() or die("Fehler: loadMenu()");
5. while($row = $sql_abfrage->fetch()){
6.     $serg = $row['M_Name'];
7.     $serg2 = $row['M_Rank'];
8.
9.     echo '<li class="nav-item">';
10.     echo '<a class="nav-link" href="index.php">'. $serg. '</a>';
11.     echo '</li>';
12. }
13. }
14. }
```

Dabei wird die Datenbankabfrage nach der Rangfolge der Menüpunkte sortiert. Zu diesem Zeitpunkt leiten aber alle Menüpunkte auf die Startseite „index.php“.

6 Fazit

Als Mitglied des „Verein zur Erhaltung und Betreuung des Volksplatzes Borna e.V.“ wurde ich vom Vorstand gefragt, ob ich nicht Interesse hätte, eine neue Webseite für den Verein zu gestalten, da diese nicht mehr zeitgemäß sei. Genaue Angaben zum Aussehen und Funktionen wurden keine gemacht, die Webseite solle aber auch für Mobilgeräte angepasst werden.

Die zum Zeitpunkt der Fertigstellung der Arbeit vorhandene Webseite basiert auf dem Content-Management-System „Typo3“. Da dieses CMS sehr komplex aufgebaut ist und der jetzige Administrator der Seite auch immer wieder Schwierigkeiten hatte und auch alle anderen Personen des Vorstandes und Vereines nicht gerade technikaffin sind, sollte die Seite für jede Person ohne viel Einarbeitung nutzbar sein.

Ursprünglich sollte das Projekt in Java oder C# entwickelt werden, da aber der vorhandene Webseiten Host keine Server für .NET Webapplikationen oder Java EE anbietet, wurde dieses Projekt in PHP umgesetzt. In diese Programmiersprache musste ich mich erst einmal einarbeiten und ich stellte fest, dass viele Elemente aus Java oder C# auch in PHP existieren, aber die Bibliotheken und auch die Verarbeitung der eingetragenen Daten anders funktioniert. Gerade die Verarbeitung der Daten hat mich manchmal an den Rand der Verzweiflung gebracht. Das führte auch dazu, dass Dateien, Klassen und auch Methoden und Funktionen manchmal sehr umständlich aufgebaut sind und ich diese jetzt im Nachhinein anders gestaltet hätte. Gerade die Angabe der Datenbank in vielen Funktionen und Methoden hätten anders implementiert werden können. Auch Fehlermeldungen müssen für den Nutzer besser dargestellt werden. Zudem muss noch eine Möglichkeit implementiert werden, den Datenschutzbestimmungen zuzustimmen.

Diese Arbeit hat im jetzigen Status die wichtigsten Funktionen eines Content-Management-Systems. Beiträge können erstellt, bearbeitet und gelöscht werden. Beiträge werden im Front-End Bereich wiedergegeben. In Zukunft soll das Erstellen, Bearbeiten und Löschen von Menüpunkten, sowie das Einrichten von Kategorien und das Erstellen von Einzelseiten erfolgen. Bis dahin ist das CMS dennoch nutzbar. Weiter soll es in Zukunft auch eine Möglichkeit geben, ein anderes Design zu implementieren. Das in diesem Projekt vorhandene Design basiert auf Bootstrap 4 und wurde mithilfe des Builders „Pingendo“ erstellt.

7 Bedienungsanleitung

7.1 Einleitung

Die hier vorgelegte Software wird zur Erstellung und Bearbeitung eines Weblogs verwendet.

7.2 Benötigte Software

Um diese Software verwenden zu können, benötigen Sie Software von dritt Anbietern. Sie benötigen:

- Webserver (z.B. Apache, Nginx o.a.)
- MySQL Server (z.B. MariaDB)

Die meisten Webhosting Anbieter haben diese Software bereits vorinstalliert.

7.3 Installation der Software

Sofern Sie die Software noch nicht vorliegen haben, können Sie die aktuellste Version auf

<https://github.com/Maxpur12/Volksplatz>

herunterladen.

Verbinden Sie sich über FTP oder einem anderen Dienst mit ihrem Webserver und laden Sie den Inhalt des Verzeichnisses auf den Server. Richten Sie nun eine neue Datenbank auf Ihrem Datenbank Server ein. Weitere Informationen dazu finden Sie bei ihrem Webhosting Anbieter. Alternativ können Sie manchmal auch über den Dienst „phpMyAdmin“ selber eine neue Datenbank erstellen.

Verbinden Sie sich mit dem Verwaltungsdienst des Datenbankservers. Dies ist meistens die Software „phpMyAdmin“. Wählen Sie nun Ihre Datenbank aus. Klicken Sie im Oberen Menü auf „Importieren“. Wählen Sie Datei „Volksplatz.sql“ aus.

In den aktuell ausgewählten Server importieren

Zu importierende Datei:

Datei kann komprimiert (gzip, bzip2, zip) oder unkomprimiert sein.

Der Dateiname einer komprimierten Datei muss mit `.[Format].[Komprimierung]` enden. Beispiel: `.sql.zip`

Durchsuchen Sie Ihren Computer: volksplatz.sql (Maximal: 2.048KiB)

Sie können auch per Drag & Drop eine Datei auf einer beliebigen Seite legen.

Zeichencodierung der Datei:

Teilweiser Import:

☒ Import abbrechen, wenn die maximale PHP-Scriptlaufzeit erreicht wird. (Damit ist es möglich, große Dateien zu importieren, allerdings kann es Transaktionen zerstören.)

Diese Anzahl Abfragen (für SQL) überspringen, beginnend von der ersten:

Andere Optionen:

☒ Fremdschlüsselüberprüfung aktivieren

Format:

Formatspezifische Optionen:

SQL-Kompatibilitätsmodus:

☒ AUTO_INCREMENT nicht für Nullwerte verwenden

Abbildung 1 – So sollte die Konfiguration aussehen.

Klicken Sie nun auf „OK“.

Gehen Sie nun auf Ihrem Webserver in das Verzeichnis „db“ und bearbeiten Sie die Datei „db.php“. Ändern Sie die Werte folgendermaßen:

<code>\$db_ip_address</code>	Die IP Adresse Ihres Datenbankservers
<code>\$db_user</code>	Den Nutzernamen Ihren Datenbankaccounts
<code>\$db_password</code>	Das Passwort Ihres Datenbankaccounts

\$database

Der Name Ihrer Datenbank

Die meisten Informationen zu den einzutragenden Werten erhalten sie von Ihrem Webhosting Anbieter.

Sollte alles funktioniert haben, können Sie die Startseite „index.php“ Fehlerfrei in Ihrem Webbrowser darstellen lassen.

7.4 Login

The image shows a login interface on a light gray background. At the top, the text "Bitte melden Sie sich an!" is centered in a dark gray font. Below this, there is a login form consisting of two stacked input fields. The top field is light blue and contains the text "admin@localhost.de". The bottom field is white and contains four dots, indicating a password. Below the input fields is a prominent blue button with the white text "Anmelden". At the bottom center of the form area, the copyright notice "© 2018-2019" is displayed in a small, gray font.

Abbildung 2 - Login Seite

Um sich in den Back-End Bereich Ihres Blogs einzuloggen, geben Sie in Ihrem Webbrowser hinter Ihrer Domain das Verzeichnis „/admin“ an. Sie werden auf die Login Seite weitergeleitet. Melden Sie sich nun an.

E-Mail-Adresse: admin@localhost.de

Passwort: toor

Sollte der Vorgang erfolgreich abgeschlossen sein, befinden Sie sich nun nicht mehr auf der Login Seite, sondern im Back-End Bereich Ihres Blogs.

7.5 Beiträge erstellen

Um Beiträge zu erstellen, klicken Sie im Menü Links in ihrem Browserfenster auf „Beitrag erstellen“.

In der Textbox „Überschrift“ können Sie die Überschrift ihres Beitrages setzen. Unter „Vorschau“ können Sie einen Vorschautext einfügen. Dieser wird auf der Startseite ihres Weblogs unter der Überschrift ihres Beitrages angezeigt. Unter „Beitragstext“ fügen Sie den Hauptteil ihres Beitrages ein. Den Text können sie mithilfe des Texteditors beliebig bearbeiten. Sofern Sie Bilder in ihren Beitrag einfügen möchten, so können Sie diese mit einem Klick auf den Button „Dateien auswählen“ einfügen. Beachten Sie jedoch, zugelassene Bilddateien sind nur JPG/JPEG, GIF oder PNG. Alle anderen Bilddateien können nicht hochgeladen werden. Auch die Datei darf auf dem Server nicht existieren, sonst wird sie nicht hochgeladen.

Haben Sie den Beitrag an den Server gesendet, so wird dieser sofort auf der Startseite mit Überschrift und Vorschautext angezeigt.



Abbildung 3 - Beitrag auf der Startseite

Wenn Sie auf „Weiterlesen“ klicken, können Sie den kompletten Beitrag ansehen. Dort befinden sich nur die Überschrift und der volle Beitragstext, jedoch nicht den Vorschautext.

7.6 Beitrag bearbeiten

Wenn Sie einen Beitrag bearbeiten möchten, klicken Sie im Back-End Menü auf den Punkt „Beitrag bearbeiten“. Dort finden Sie eine Auflistung der Beiträge. Wählen Sie nun den gewünschten Beitrag durch den Button „Bearbeiten“ aus.

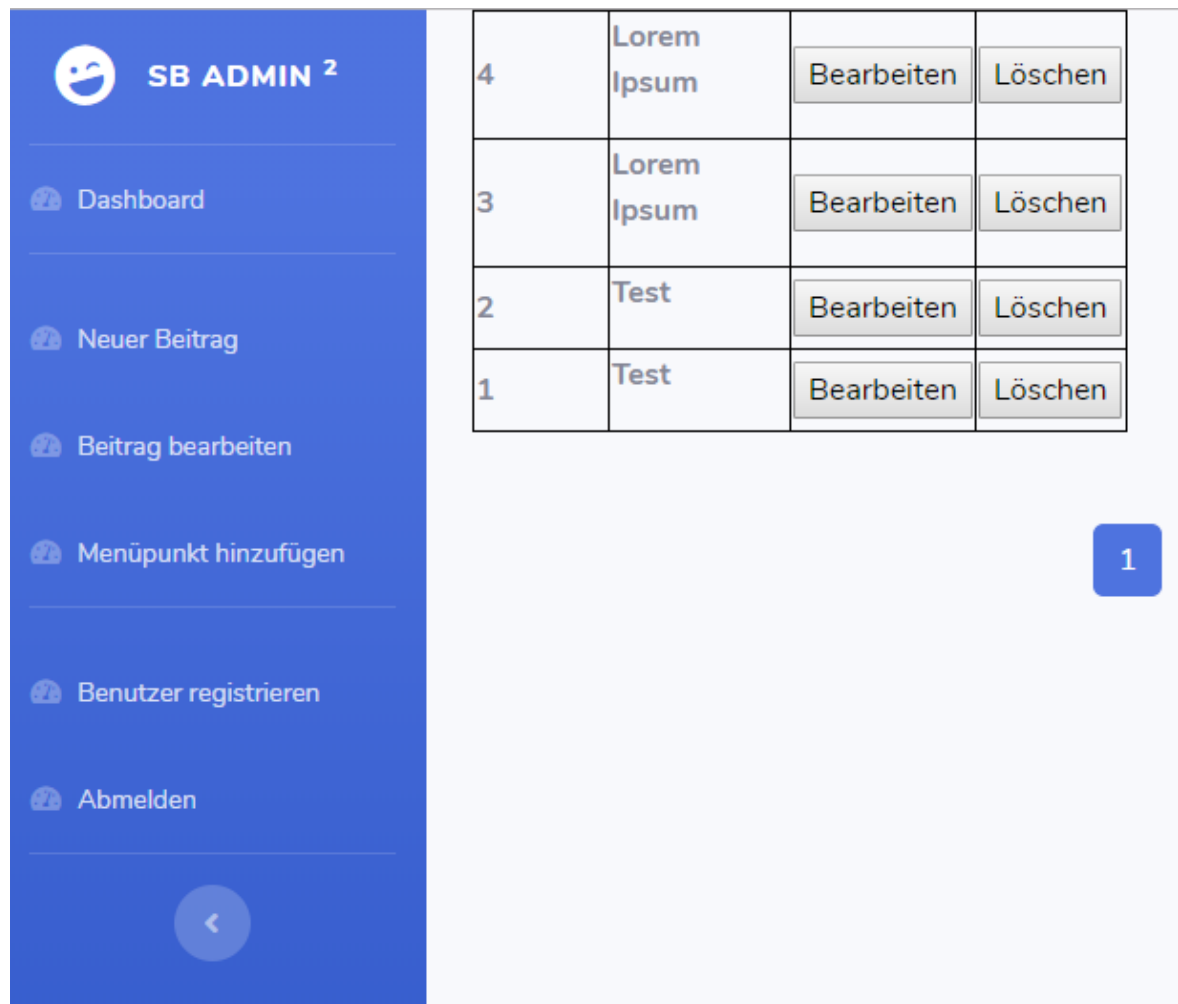


Abbildung 4 - Auflistung aller Beiträge

Es werden nun Überschrift, Vorschautext und Beitragstext in den einzelnen Textboxen geladen und können diese Texte nun bearbeiten. Wenn Sie ein Bild hinzufügen wollen, können Sie dies unter dem Punkt „Bilder einfügen“ machen. Wenn Sie ein Bild löschen wollen, wählen Sie das Bild in der Tabelle aus und klicken Sie auf den dazugehörigen „Löschen“ Button.

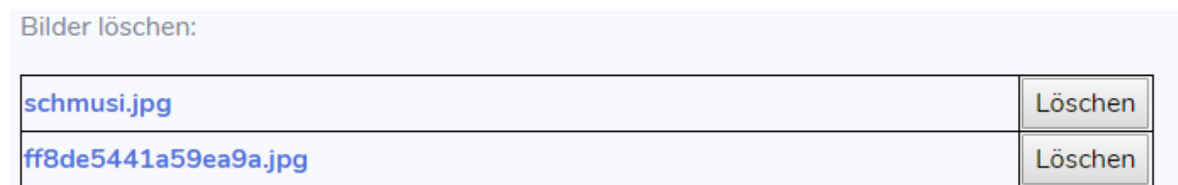


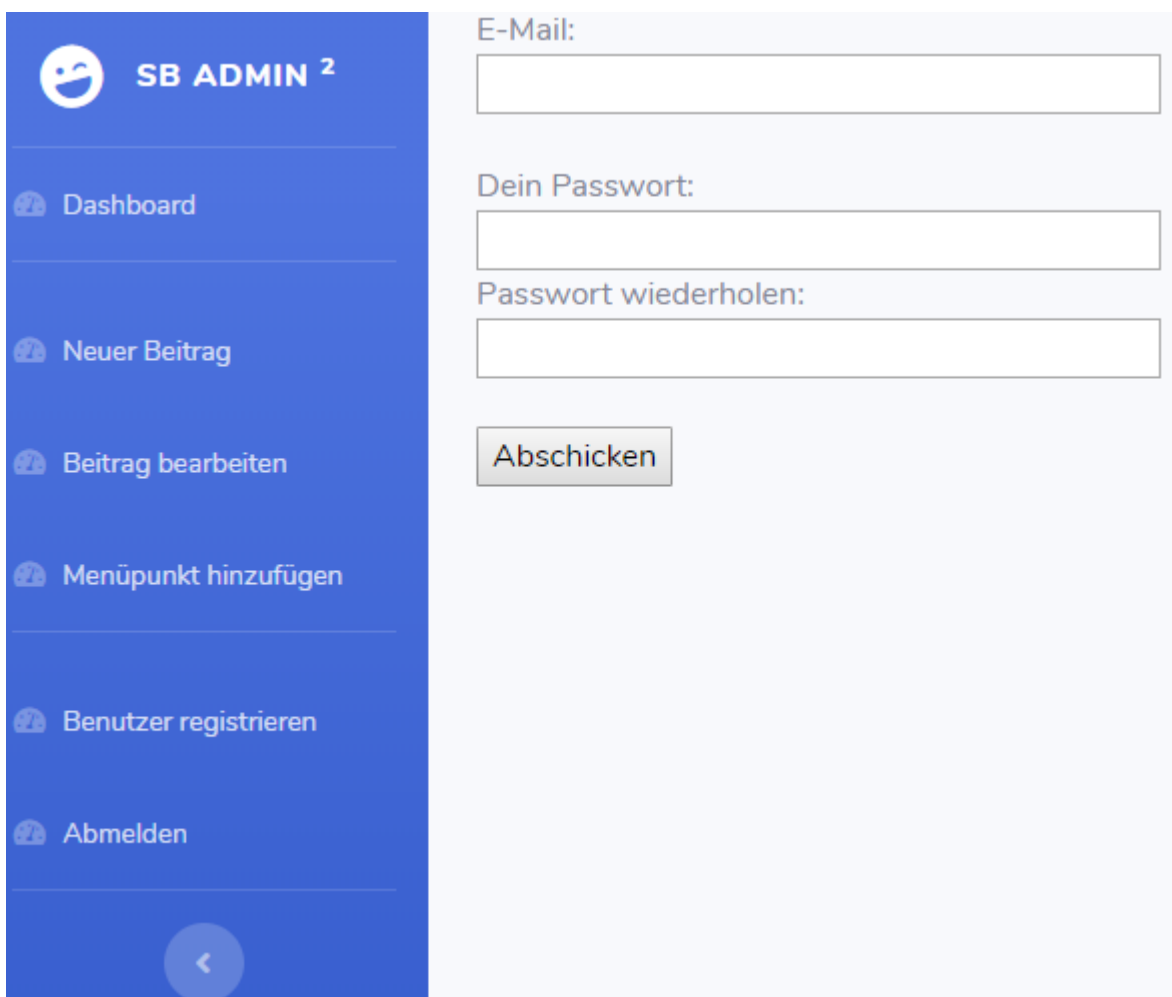
Abbildung 5 - Einzelne Bilder löschen

7.7 Beitrag löschen

Wenn Sie einen Beitrag löschen wollen, so wählen Sie diesen unter „Beitrag bearbeiten“ aus und klicken Sie auf den „Löschen“ Button in der jeweiligen Tabellenzeile. Der Beitrag und die Bilder werden unwiderruflich vom Server gelöscht.

7.8 Benutzer registrieren

Wenn sie einen neuen Benutzer erstellen wollen, gehen Sie dazu auf „Benutzer registrieren“ im Back-End Bereich Ihres Weblogs. Geben Sie nun die E-Mail-Adresse und das Passwort des neuen Benutzers ein. Bestätigen Sie das Passwort und klicken Sie auf „Abschicken“.



The screenshot shows the 'SB ADMIN 2' interface. On the left is a blue sidebar with a menu containing: Dashboard, Neuer Beitrag, Beitrag bearbeiten, Menüpunkt hinzufügen, Benutzer registrieren, and Abmelden. The 'Benutzer registrieren' option is highlighted. The main content area is light blue and contains the registration form with the following elements:

- E-Mail:** A text input field.
- Dein Passwort:** A text input field.
- Passwort wiederholen:** A text input field.
- Abschicken**: A button to submit the form.

Abbildung 6 - Nutzer Registrierung

7.9 Abmelden

Wenn Sie sich aus dem Back-End abmelden wollen, klicken Sie einfach „Abmelden“. Sie werden aus dem Back-End Bereich abgemeldet und werden auf die Login Seite weitergeleitet.

Literatur und verwendete Hilfsmittel

PHP-Handbuch: <https://www.php.net/manual/de/index.php>

Quelltext-Editor: Visual Studio Code <https://code.visualstudio.com/>

Software-Dokumentationswerkzeug: Doxygen <http://www.doxygen.nl/>

Apache und MariaDB mithilfe der Entwicklungsumgebung XAMPP
<https://www.apachefriends.org/de/index.html>

Texteditor: TinyMCE <https://www.tiny.cloud/>

Design:

- Bootstrap 4 <https://getbootstrap.com/>
- Back-End Design ,SB-Admin 2': <https://startbootstrap.com/themes/sb-admin-2/>
- Front-End Design erstellt mit ,Pingendo': <https://pingendo.com/>
- Login Seite: <https://getbootstrap.com/docs/4.1/examples/sign-in/>

Programmmentwurf: ArgoUML <http://argouml.tigris.org/>

References

- [1] "Content Management System (CMS) Definition,"
https://www.gruenderszene.de/lexikon/begriffe/content-management-system-cms?interstitial_click.
- [2] "Pagination - Was ist das? - Seobility Wiki," 5/24/2019,
<https://www.seobility.net/de/wiki/Pagination>.
- [3] "PHP: SQL Injection - Manual," 6/16/2019,
<https://www.php.net/manual/de/security.database.sql-injection.php>.

Anlagen

Programmentwurf

DVD:

- Content-Management-System
- Datenbank
- Schriftliche Ausarbeitung (PDF)
- Programmentwurf
- Dokumentation

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Borna, den 19.06.2019

Max Stötzner