



Projektdokumentation

Programmierung einer App zur Verwaltung von Kunstkursen

Anja Kraus

Fachinformatikerin Anwendungsentwicklung

Ausbildungsbetrieb:

Berufsförderungswerk Schöenberg gGmbH

Bühlhof 6, 75328 Schöenberg

Projektübersicht

Thema:	Erstellung einer Web-Applikation zur Verwaltung von Kunstkursen
Bearbeiter:	Anja Kraus
Beruf:	Fachinformatikerin Anwendungsentwicklung
Ausbildungsbetrieb:	Berufsförderungswerk Schömburg gGmbH Bühlhof 6 75328 Schömburg
Betreuer:	Dieter Göttisheim Fachbereichsleiter IT- und Elektroberufe Berufsförderungswerk Schömburg
Zeitraum:	01.04.2019 – 12.04.2019
Beschreibung:	Web-Applikation zur Erstellung von Beiträgen und zur Verwaltung von Veranstaltungen und deren Teilnehmern, inbegriffen ist auch ein Anmeldeformular für Website-Besucher. Projekt-Basis ist die bestehende Firmenwebsite und eine angebundene Datenbank.

Inhaltsverzeichnis

1 Auftrag.....	6
1.1 Auftraggeber und Projektumfeld	6
1.2 Kundenwünsche.....	6
1.3 Projektziel	6
1.4 Ist-Zustand	7
1.5 Soll-Zustand	7
1.6 Abgrenzung und Rahmenbedingungen	8
1.7 Prozessschnittstellen	8
2 Planung	9
2.1 Meilensteine	9
2.2 Projektablauf und Teilaufgaben.....	10
2.3 Zeitplanung	10
2.4 Projektablaufplan: EPK-Diagramm.....	11
2.5 Kosten- und Sachmittelplanung	12
2.6 Personalplanung	12
3 Durchführung des Projektes.....	13
3.1 Entwicklungsumgebung einrichten	13
3.2 Visualisierungsskizzen.....	13
3.3 UML-Anwendungsfall-Diagramm	14
3.4 Benutzeroberflächen anlegen.....	15
3.5 Gestaltung mit Hilfe von Bootstrap	17

3.6	ERM – Entity-Relationship-Modell	19
3.7	Datenbank-Design erstellen.....	20
3.8	Datenbank mit Testdaten füllen.....	21
3.9	Relationale Datenbank	22
3.10	Anpassungen	23
3.11	Datenbankzugriff per PDO	25
3.11.1	Verbindungsaufbau	25
3.11.2	Verbindungsfehler.....	26
4	Projektarchitektur	27
4.1	MVC-Pattern.....	27
4.2	Das Model	27
4.3	View	29
4.4	Controller.....	30
5	Projektabschluss	31
5.1	Qualitätssicherung	31
5.2	Soll-Ist-Vergleich – erreichte Ziele	33
5.3	Erweiterbarkeit.....	33
5.4	Abweichungen.....	33
5.5	Benötigte Projektzeiten.....	34
5.6	Fazit	34

Abbildungsverzeichnis

Abb. 1	EPK-Projektablaufplan	11
Abb. 2	Vorentwurf der Oberfläche Kurs-Design	13
Abb. 3	UML –Anwendungsfalldiagramm	14
Abb. 4	Beiträge auf der Seite Kunstkurse	15
Abb. 5	Benutzeroberfläche Anmeldung	16
Abb. 6	Bootstrap: CSS und Klassen-Beispiel	17
Abb. 7	Kundentabelle im Bootstrap-Grid	18
Abb. 8	ERM – Entity-Relationship-Modell	19
Abb. 9	Datenbank Kunstkiste	20
Abb. 10	Tabelle Kunde in phpMyAdmin	21
Abb. 11	Tabelle Kunde im Backend der WebsiteDatenbank-Übersicht.....	21
Abb. 12	Bildpfad in der Bild-Tabelle	22
Abb. 13	Bilder-Ordner und img-Ordner	22
Abb. 14	Skizze Benutzeroberfläche Kunde	23
Abb. 15	Benutzeroberfläche Kunden endgültige Seite	24
Abb. 16	Verbindungsaufbau mit PDO	25
Abb. 17	PDO mit Try-Catch-Blog	26
Abb. 18	Model/Resource/Kunde.php	27
Abb. 19	Model/Kunde.php	28
Abb. 20	Namespace View	29
Abb. 21	Controller/Index.php	30

Tabellenverzeichnis

Tab. 1	Termine im Projekt.....	9
Tab. 2	Zeitplanung.....	10
Tab. 3	Soll-Ist-Zeiten-Vergleich	34

1 Auftrag

1.1 Auftraggeber und Projektumfeld

Bei einem kleinen fiktiven Fachgeschäft für Künstler-Bedarf auf dem Lande steht ein Generationswechsel an. Die Tochter der Eigentümer möchte den Betrieb übernehmen. Allerdings braucht sie neue Wege um Kunden zu gewinnen, damit sie von dem Geschäft auch leben kann. Aus diesem Grund möchte sie als weitere Einnahmequelle Kunstkurse veranstalten und diese auf der Firmenwebsite bewerben. Dort sollen sie auch gebucht und verwaltet werden können. Aktuell gibt es nur eine einfache, statische Website.

1.2 Kundenwünsche

Die Kundin möchte selbstständig Artikel zur Bewerbung Ihrer Kunstkurse zusammenstellen und in ihre Homepage einfügen können. Besucher ihrer Website sollen sich direkt online für diese Kurse anmelden können. Ebenso möchte sie die Kurse und deren Teilnehmer auch verwalten können und braucht dazu eine angebundene Datenbank. Zunächst soll alles recht einfach und unkompliziert sein ohne Benutzerregistrierung für die Websitebesucher.

1.3 Projektziel

Die Firmen-Website erhält eine zusätzliche Seite auf der Kunstkurse angeboten werden. Besucher der Website sollen sich dort für Kurse anmelden können. Im Backend möchte die Kundin eine Applikation. Mit dieser sollen die Kunstkurse gestaltet und hochgeladen werden. Kunstkurse, Teilnehmer, Bezahlung und Anwesenheit sollen im Backend verwaltet werden können und in einer angebotenen MySQL-Datenbank gespeichert werden. Integriert sein soll auch das Auslösen einer Rundmail an alle Teilnehmer eines Kurse.

1.4 Ist-Zustand

Bisher leben die Inhaber der Kunstkiste von Kunden die ins Geschäft kommen und vor Ort Künstlerbedarf-Artikel einkaufen. Das ist bisher die einzige Einnahmequelle und bis jetzt kommen kaum jüngere Kunden in das Geschäft. Es gibt eine statische Website, auf der einige Informationen über die Unternehmer und das Geschäft zu finden sind, ebenso Kontaktdaten und Anfahrtswege. Diese Seite können die Unternehmer bisher nicht selbst aktualisieren und für Besucher der Seite gibt es keine Möglichkeit zur Interaktion.

1.5 Soll-Zustand

Durch das Projekt sollen folgende Punkte realisiert werden:

- Bewerbung von Kunstkursen auf der Website
- Einfacher Backendzugang für die Besitzer der Kunstkiste
- Erstellung und Hochladen von Kunstkursbeschreibungen
- Zugriff auf einen Bilderordner zum Einfügen von Bildern für Kurse
- Anmeldemöglichkeit für Kurse durch Besucher der Website
- Automatische Mailbestätigung für die Anmeldungen
- Automatische Info-Mail wenn eine Anmeldung eingeht.
- Archivierung, Auslesung, Änderung und Löschung von Kundendaten in einer MySQL-Datenbank mit PHP-myAdmin
- Zusammenführung von Teilnehmerdaten und Kursdaten
- Abfrage der Teilnehmer pro Kurs
- Abfrage der Kurse die ein Teilnehmer bereits absolviert hat.
- Rundmailfunktion an alle Teilnehmer eines Kurses
- Speichern von Zahlungseingängen oder Mahnungsausgängen in der Datenbank

1.6 Abgrenzung und Rahmenbedingungen

Ziel dieses Projektes ist die Entwicklung eines kleinen Web-Programms, welches mit einer MySQL-Datenbank interagieren kann. Es wird implementiert als Browseranwendung, die in die Firmenwebsite integriert ist. Aktualisierungen und Updates sind nicht vorgesehen, da diese Software auch unter neueren Browserversionen laufen wird, die in der Regel abwärts kompatibel sind.

Datensicherung erfolgt im Rahmen der regelmäßigen Sicherungen der kompletten Firmenwebsite welche auch die Datenbank miteinbezieht.

Die Website selbst und erforderliche Änderungen zur Umwandlung in eine dynamische Website, beispielsweise das neue Menu und die Anpassungen der übrigen Seiten an SASS, gehört nicht zum Projekt, es wird lediglich die Seite Kunsturse inklusive Anmeldeformular erstellt und die Backendapplikation entwickelt.

1.7 Prozessschnittstellen

Prozessschnittstellen gibt es zwischen der fiktiven Chefin des Künstlerbedarf-Fachgeschäfts Kunstiste als Auftraggeberin und der Entwicklerin als Auftragnehmerin. Des weiteren werden durch das Berufsförderungswerk Schömburg der Arbeitsplatz zur Entwicklung des Projektes gestellt und durch den Bereichsleiter IT- und Elektroberufe, Herrn Dieter Göttisheim die Betreuung der Durchführung des Projektes gewährleistet. Der BWL-Anteil an Projekt und Doku wird von der BWL Ausbilderin Beatrice Stokelj betreut.

2 Planung

2.1 Meilensteine

Folgende Meilensteine sind im Verlauf des Projektes geplant

- Einrichtung Entwicklungsumgebung
- Vorabskizzen zur Visualisierung der Anforderungen
- Erstellen der Benutzeroberflächen mit HTML5, Bootstrap und SASS
- Anlegen der Datenbank inkl. Testdaten
- Entwicklung der Softwareprozesse mit PHP und MySQL
- Lokale Tests und Fehlermanagement
- Dokumentationen erstellen

Bezeichnung	Termin
Projektzeitraum	01.04. – 12.04.2019 (70 h)
<u>Meilenstein</u> : Entwicklungsumgebung	01.04.2019
<u>Meilenstein</u> : Vorabskizze	02.04.2019
<u>Meilenstein</u> : Benutzeroberflächen	04.04.2018
<u>Meilenstein</u> : Datenbank u. Testdaten	05.04.2019
<u>Meilenstein</u> : PHP + MySQL	09.04.2019
<u>Meilenstein</u> : Tests-Fehlermanagement	10.04.2019
<u>Meilenstein</u> : Kundendokumentationen	12.04.2019

Tab. 1 Termine im Projekt

2.2 Projektablauf und Teilaufgaben

Durch folgende Teilschritte soll das Projekt realisiert werden:

- Bedarfsanalyse anhand Kundenwunsch
- Installation der Entwicklungsumgebung: XAMPP und Visual Studio Code
- Skizzen für die Benutzeroberflächen in Frontend und Backend
- UML-Anwendungsfall-Diagramm für Benutzeroberfläche
- Entwickeln der Benutzeroberflächen mit HTML5, Bootstrap und SASS
- Software entwickeln mit PHP und MYSQL nach dem MVC-Modell
- ERM-Diagramm für die Datenbank
- Anlegen der Datenbanken mit Testdaten in phpmyAdmin
- Funktionen und Datenbankabfragen testen
- Projektdokumentation + Kundendokumentation erstellen

2.3 Zeitplanung

Bezeichnung	Geplante Zeit
Bedarfsanalyse gemäß Anforderung	2 Std
Planungsdiagramme + Skizzen Benutzeroberfläche	10 Std
Entwicklungsumgebung einrichten	2 Std
Datenbank mit Testdaten	4 Std
Benutzeroberflächen entwickeln	16 Std
Entwickeln + Implementieren der PHP + SQL-Software	24 Std
Testläufe und Fehlerkorrektur	4 Std
Kundendokumentation	8 Std
Gesamtzeit	70 Std

Tab. 2 Zeitplanung

2.4 Projektablaufplan: EPK-Diagramm

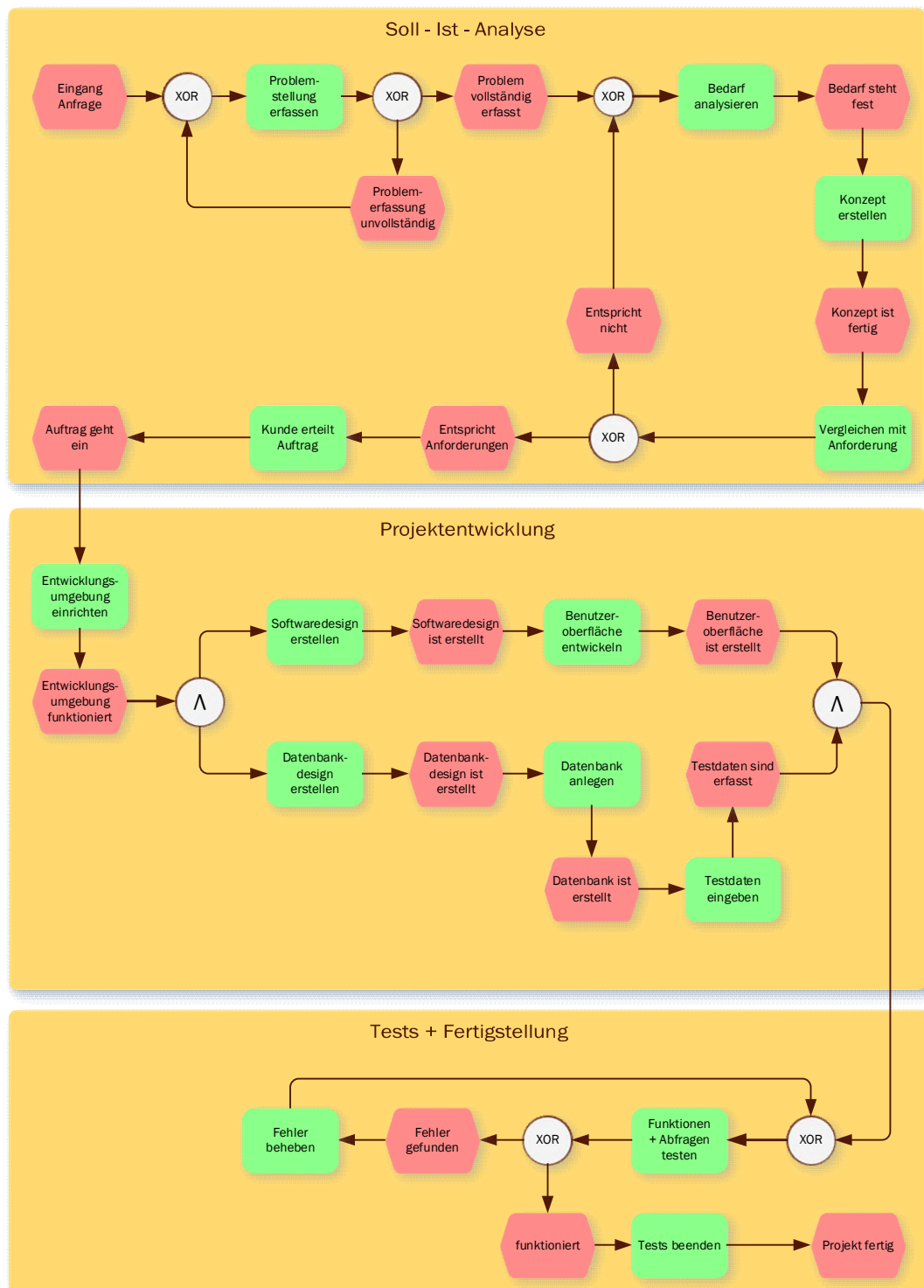


Abb. 1 EPK-Projektablaufplan

2.5 Kosten- und Sachmittelplanung



Für das Projekt kann die im Berufsförderungswerk Schöenberg vorhandene Infrastruktur kostenlos verwendet werden. Es fallen somit keine Gemeinkosten an.

Hardware muss nicht beschafft werden, ein PC mit 4 GHz - Intel Core i7-4790K Prozessor, 24 GB RAM und Windows 10 Education ausgestattet steht zur Verfügung. Für die Planungsdiagramme kann Microsoft Visio benutzt werden. Für die Entwicklungsumgebung wird



und



mit  APACHE Server und  phpMyAdmin

verwendet, welches OpenSource Programme unter MIT- bzw. GNU-Lizenzen sind, die kostenlos heruntergeladen und benutzt werden können.

2.6 Personalplanung

Dieses Projekt wird durch die Entwicklerin alleine erstellt, es ist keine Personalplanung erforderlich und es fallen keine weiteren Personalkosten an.

Die für das Projekt benötigte Arbeitszeit wird mit 75,- € pro Stunde abgerechnet. Bei 70 Arbeitsstunden entstehen Gesamtkosten von insgesamt 5.250,- €

3 Durchführung des Projektes

3.1 Entwicklungsumgebung einrichten

Für die Entwicklungsumgebung Visual Studio Code als benutzt. Wurden bereits viele nützliche Erweiterungen programmiert, die man einfach installieren kann. So gibt es zum Beispiel eine automatische Code-Vervollständigung und viele voreingestellte Tastenkombinationen damit das Programmieren schneller geht. PHP braucht einen lokalen Server, damit es ausgeführt bzw. interpretiert wird. Dafür wird XAMPP installiert welches mit phpMyAdmin auch gleich ein Datenbankmanagementsystem mitbringt.

3.2 Visualisierungsskizzen

Um die Anforderungen an das Projekt leichter konkretisieren zu können, werden mit Microsoft Visio, Skizzen der Benutzeroberflächen gefertigt. So bekommt man schnell eine Vorstellung wie die Benutzeroberfläche gestaltet sein sollte und welche Programm-Funktionen dazu implementiert werden müssen.

The wireframe sketch shows a web interface for course management. At the top is a header box labeled 'Kursfassung'. Below it, the main content area is divided into two columns. The left column has a sub-header 'Kursbezeichnung' above a large rectangular area containing a simple line drawing of a person climbing a mountain. Below this drawing is a text input field labeled 'Bild aus Ordner einfügen'. The right column contains a search bar labeled 'Suche', followed by a large text area for 'Beschreibung der Kursinhalte', and then three smaller text input fields labeled 'Kurstermin u. -Dauer', 'Beschreibung des Ablaufs (Zeit)', and 'Kursgebühren/Kosten'. At the bottom of the interface, there is a row of five buttons: 'Kursanlegen', 'ändern', 'speichern', 'löschen', and 'abbrechen'.

Abb. 2 Vorentwurf der Oberfläche Kurs-Design

3.3 UML-Anwendungsfall-Diagramm

Hier wird dargestellt wer Zugriff auf welche Komponenten haben muss und welche Funktionalität die App tatsächlich braucht.

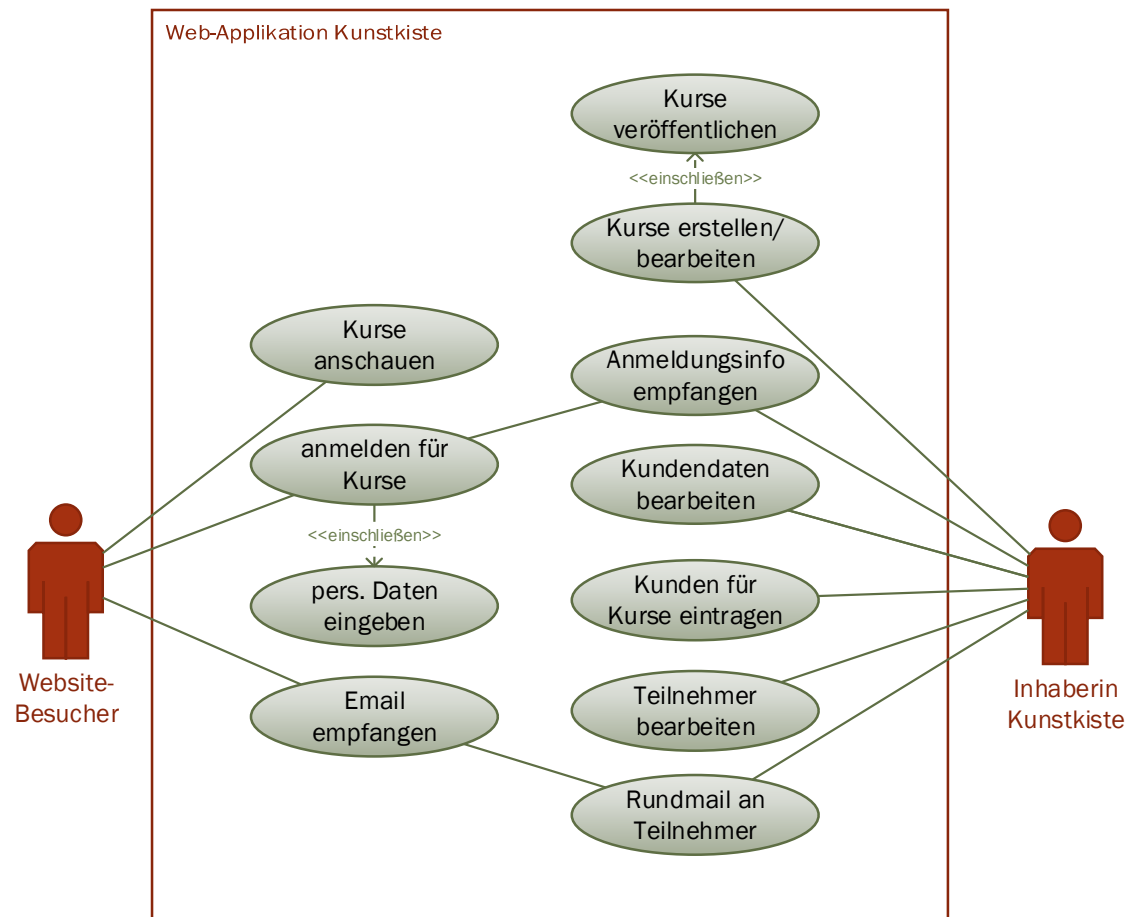


Abb. 3 UML –Anwendungsfalldiagramm

3.4 Benutzeroberflächen anlegen

Als Start für das Projekt wird die zusätzliche Seite Kunstkurse für die bestehende Firmenwebsite angelegt. Auf der Seite werden Beiträge zu den geplanten Kunstkursen entworfen mit jeweils einem Button „Hier für den Kurs anmelden“. Durch Drücken des Buttons öffnet sich die Benutzeroberfläche „Anmelden“ gleichzeitig wird die Bezeichnung des Kurses in das Anmeldeformular eingefügt.

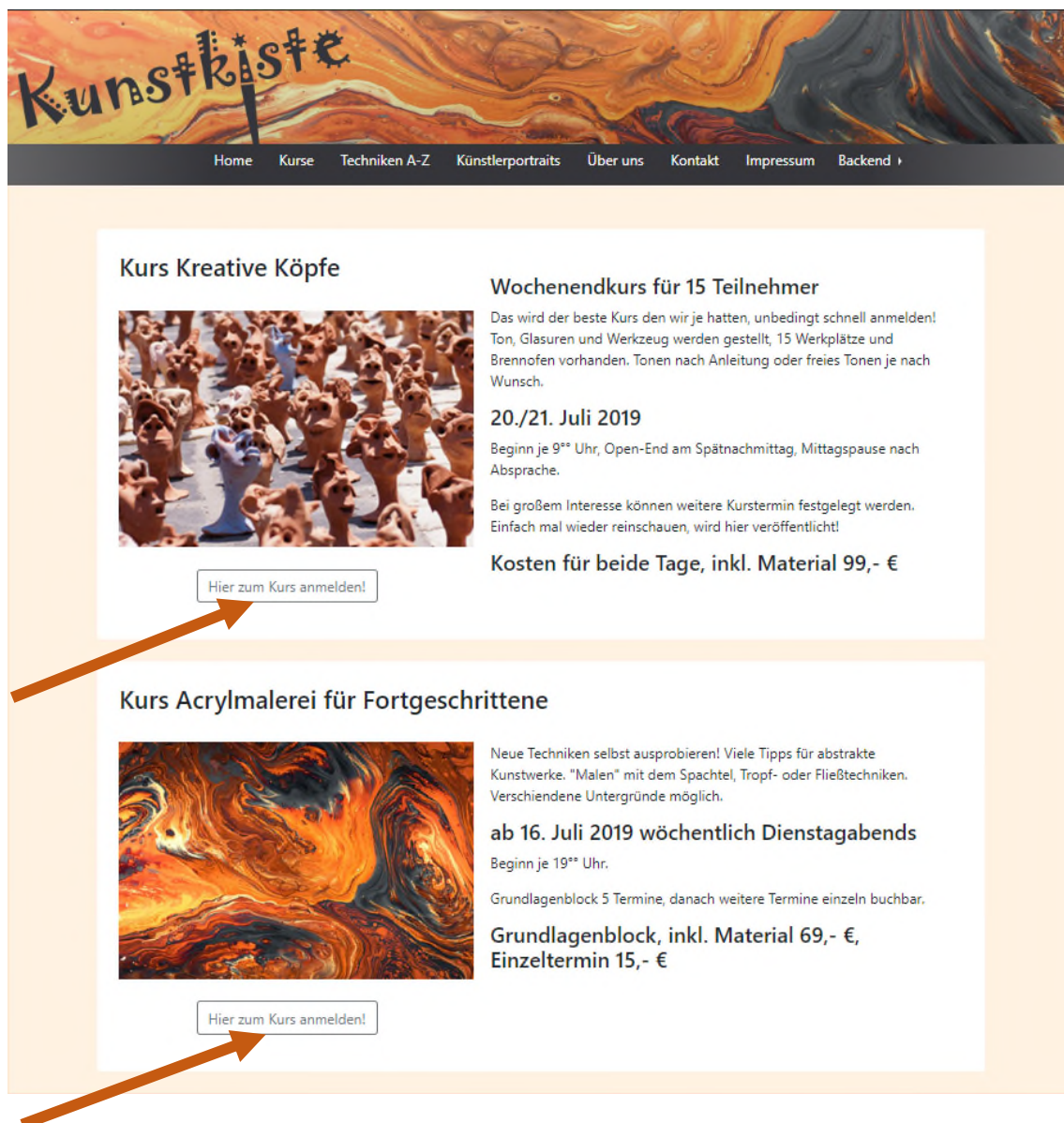


Abb. 4 Beiträge auf der Seite Kunstkurse

In das Anmeldeformular kann der Website-Besucher seine Daten eintragen und bei Aktivieren des Senden-Buttons werden diese Daten direkt in die Tabelle „Kunden“ der Datenbank Kunstkiste gespeichert.

The screenshot shows a web browser window with the URL `localhost/Kunstkiste/Frontend/Anmeldung.php`. The page features a header with the 'Kunstkiste' logo and a navigation menu with links: Home, Kurse, Techniken A-Z, Künstlerportraits, Über uns, Kontakt, Impressum, and Backend. The main content area is titled 'Anmeldung für den Kurs Kreative Köpfe' and contains the following text:

Wir freuen uns sehr über Ihr Interesse an unserem Kurs Kreative Köpfe! Sie können sich hier für den Kurs vormerken lassen.

Anschließend bekommen Sie eine Bestätigungsmail, dass Ihre Anfrage eingegangen ist. Zeitnah werden Sie benachrichtigt ob noch Plätze frei sind und ob der Kurs stattfinden kann.

Bitte geben Sie im Formular Ihre Daten ein und drücken dann auf "senden".

The form consists of the following fields:

- Vorname
- Name
- Straße
- Hausnummer
- Postleitzahl
- Ort
- Telefon
- E-Mail
- Bemerkungen (with placeholder text: Ihre Mitteilung an uns:)

A 'senden' button is located at the bottom of the form.

Abb. 5 Benutzeroberfläche Anmeldung

Der Senden-Button löst gleichzeitig eine Bestätigungsmail an die E-Mail-Adresse des Interessenten aus und eine Mail an die Inhaberin der Kunstkiste, die sie über die Anmeldung eines Interessenten informiert. Auch hier wird die Bezeichnung des Kurses weitergegeben, damit der Interessent für den richtigen Kurs eingetragen werden kann.

Die Eintragung in die Tabelle Teilnehmer muss die Kundin dann im Backend selbst vornehmen, wenn tatsächlich ein Kurs gebucht wurde.

3.5 Gestaltung mit Hilfe von Bootstrap

Um schnell eine übersichtliche Seite zu erhalten, wird Bootstrap 4 verwendet. Durch benutzen der in Bootstrap implementierten Klassen und Einbinden der frei verfügbaren CSS-Datei von Bootstrap, kann auf fertige Elemente wie zum Beispiel Buttons oder Formulare zurückgegriffen werden. Auch die Anpassung der Darstellung auf unterschiedliche Bildschirmgrößen ist schon angelegt und steht automatisch zur Verfügung, wenn das Metatag <viewport> verwendet wird.

Beispiel: der Button „Hier zum Kurs anmelden!“ bekommt die Klasse: „btn btn-outline-secondary“ und erhält durch die eingebundene Datei:

```
<head>
  <title>Kunstkiste</title>
  <!-- Latest compiled and minified CSS -->
  <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

<div class="row justify-content-center row-book">
  <form action="process.php" method="POST">
    <div class="form-group">
      <a class="btn btn-outline-secondary" href="Frontend/Anmeldung.php" role="button">Hier zum Kurs anmelden!</a>
    </div>
  </form>
</div>
```

Abb. 6 Bootstrap: CSS und Klassen-Beispiel

automatisch die für Bootstrap typischen abgerundeten Ecken und den grauen Rand. Auch für das Anmeldeformular wird das Bootstrap-Grid verwendet:

Die Klasse „row“ gibt den äußeren Rahmen als „Reihe“ bzw. Container über die festgelegte Breite des „body“. Darin ist mit der Klasse „col“ eine Spalte angelegt, mit „sm, md, lg...“ usw. wird festgelegt für welche Bildschirmgrößen das gelten soll und die Zahl am Ende gibt die Breite der Spalten von 1-12 an. Wobei 12 über die volle Breite geht 6 über die Hälfte usw. Wenn das Label die Klasse „col-sm-2“ hat und das Inputfeld die Klasse „col-sm-9“ belegen sie zusammen 11 der 12 Spalten und 1 Spalte ist noch Abstand nach Außen (Code S. 21, Abb. 10).

```

<div class="row" id="anlegen">
  <form class="form-inline" action="../../process.php" method="POST">
    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2"></label>
      <h3>Neuer Kunde</h3>
    </div>
    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2">Vorname</label>
      <input type="text" name="vorname" class="form-control col-sm-9"
        value="<?php echo $vorname; ?>" placeholder="Vorname" required>
    </div>
    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2">Name</label>
      <input type="text" name="nachname" class="form-control col-sm-9"
        value="<?php echo $name; ?>" placeholder="Name" required>
    </div>
    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2">Straße</label>
      <input type="text" name="str" class="form-control col-sm-9"
        value="<?php echo $str; ?>" placeholder="Straße">
    </div>
    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2">Hausnummer</label>

```

(Mitte der Datei ausgeschnitten).....

```

    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2" for="email">E-Mail</label>
      <input type="email" class="form-control col-sm-9" name="email" id="email"
        value="<?php echo $email; ?>" placeholder="E-Mail-Adresse" required>
    </div>
    <div class="form-group col-sm-12 mb-3">
      <label class="col-sm-2">Bemerkungen</label>
      <input type="text" name="bemerkung" class="form-control col-sm-9"
        value="<?php echo $text; ?>" placeholder="Ihre Mitteilung an uns:">
    </div>
    <div class="form-group col-sm-12">
      <label class="col-sm-2"></label>
      <?php
        if ($update == true) :
          ?>
          <button type="submit" class="btn btn-secondary
            col-sm-4" name="update">ändern</button>
        <?php else : ?>
          <button type="submit" class="btn btn-secondary
            col-sm-4" name="save">speichern</button>
        <?php endif; ?>
      </div>
    </form>
  </div>

```

Abb. 7 Kundentabelle im Bootstrap-Grid

3.6 ERM – Entity-Relationship-Modell

Mit einem ERM-Diagramm wird die benötigte Datenbank-Struktur ermittelt. Jedes dargestellte Objekt, spiegelt eine Tabelle der Datenbank wieder. Die Attribute entsprechen den Spalten, die Verbindungen/Beziehungen zwischen den Tabellen werden durch die Rauten dargestellt.

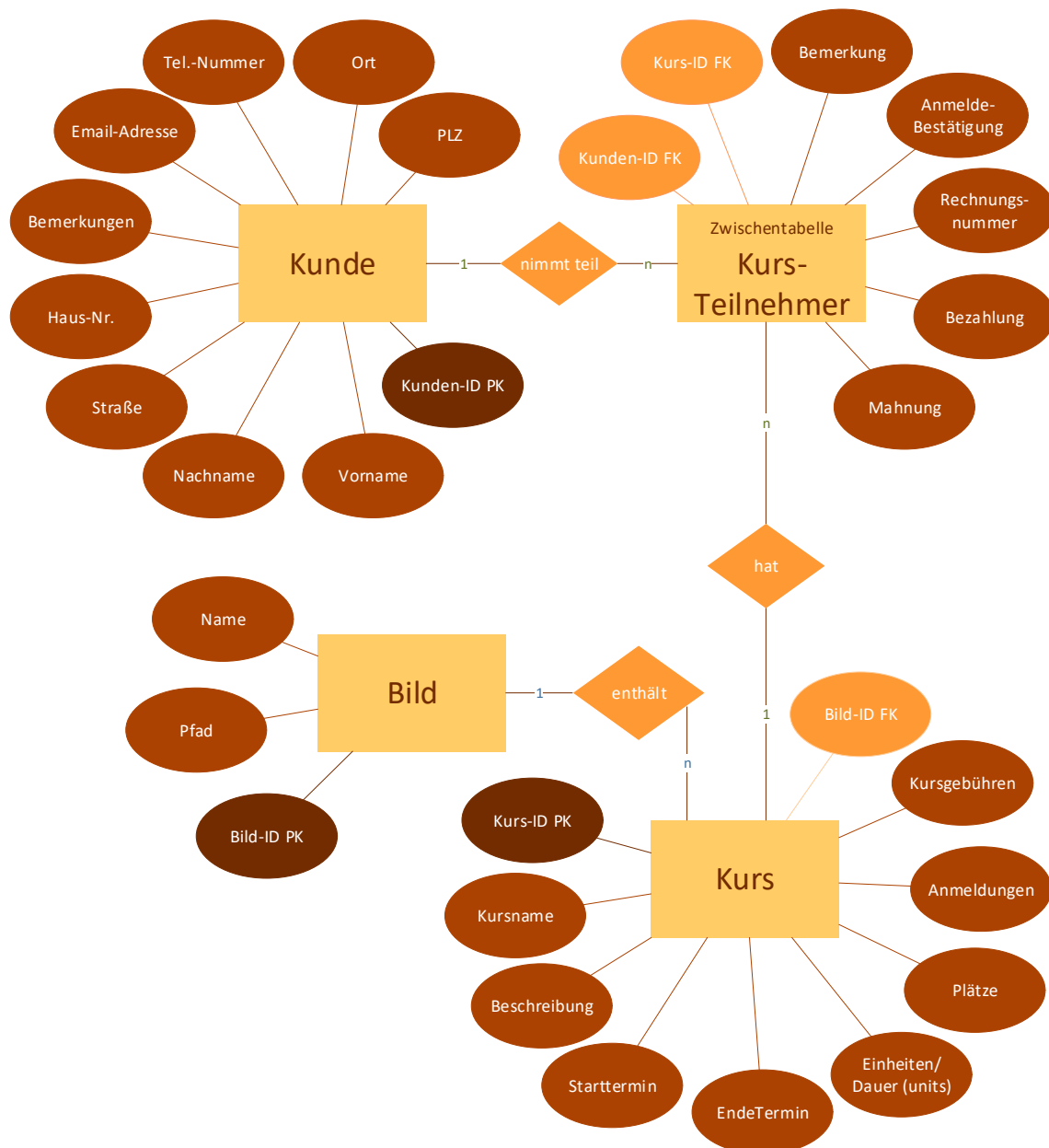


Abb. 8 ERM – Entity-Relationship-Modell

3.7 Datenbank-Design erstellen

Zur Speicherung der Daten wird die Datenbank Kunstkiste mit den erforderlichen Tabellen, entsprechend des ERM-Diagramms in phpMyAdmin angelegt.

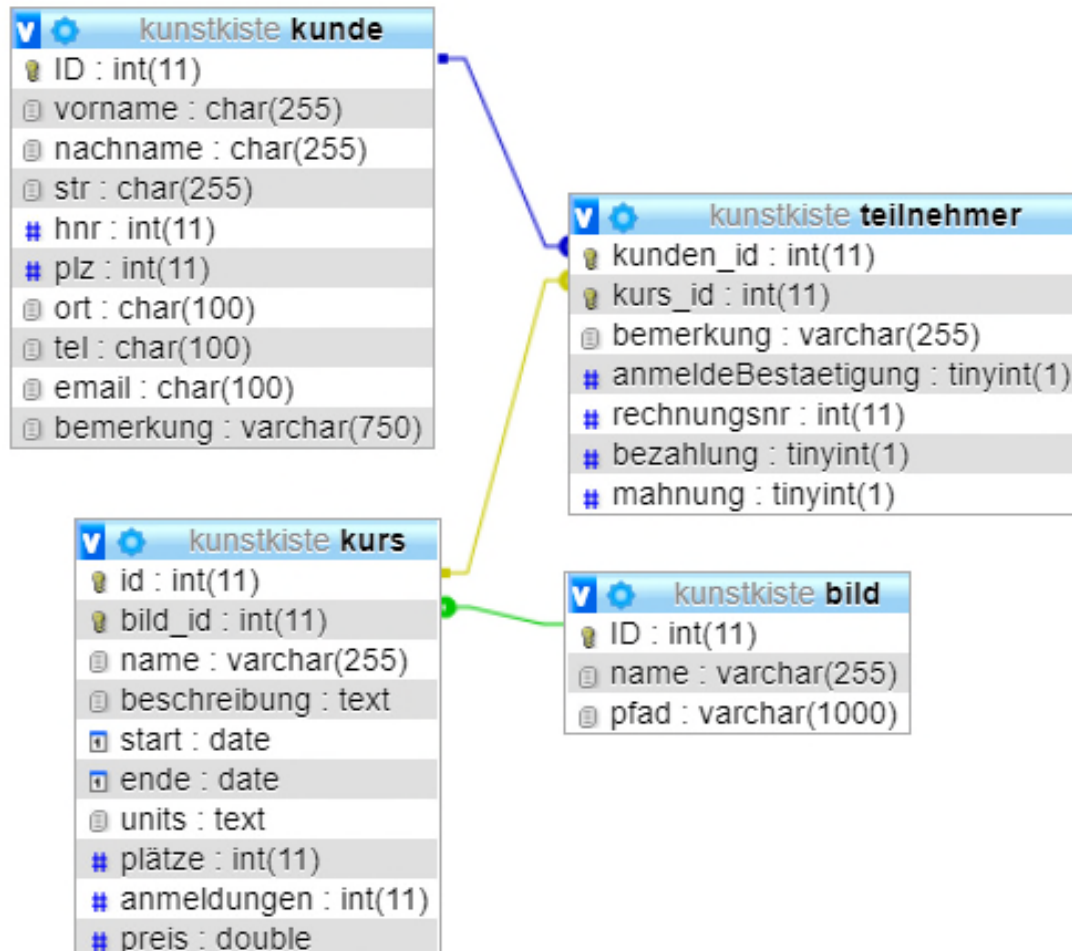


Abb. 9 Datenbank Kunstkiste

3.8 Datenbank mit Testdaten füllen

Zunächst werden die Tabellen noch mit einigen Testdaten gefüllt. Dadurch kann leichter überprüft werden, ob die Verbindung zur Datenbank korrekt eingerichtet wurde und man kann sofort die Darstellung der Tabelle auf der Seite anpassen.

ID	vorname	nachname	str	hnr	plz	ort	tel	email	bemerkung
1	Otto	Normalverbraucher	Hauptstr.	10	12345	Musterhausen	01234/789654	otto.normalverbraucher@email.de	Das ist die erste Bemerkung beim ersten Ku
2	Klara	Extravagant	Superweg	1	98765	Spezialstadt	0121/445566	klara@extravagant.de	
3	Klaus	Normalverbraucher	Hauptstr.	10	12345	Musterhausen	01234/789654	klaus.normalverbraucher@email.de	Sohn von Otto Normalverbraucher
4	Silvia	Extravagant	Superweg	1	98765	Spezialstadt	0121/445566	Silvia123@extravagant.de	Tochter von Klara Extravagant
5	Hans	GuckindieLuft	Hochweg	3	223366	Himmel	1236654778899	GuckindieLuft@email.de	
8	Erster	Neuer	weg	3	1235654	Hier	123654	erster@neuer.de	
10	Martin	Mustermann	Schaustr.	3	55892	Stadt	1236654778899	martin55@email.de	
13	Lisa	Neuer	Alterweg	5	1235654	Hier	123654	LisaMaria@neuer.de	Montag
15	Rainer	Kleinert	Neue Str.	15	98745	Kleiner Ort	12368/789954	Kleinert.Rainer@web.de	Hallo!
19	Lisa	Logisch	Neuweg	3	223366	Großstadt	1236654778899	vor.name@email.de	nochmal ein angelegter Kunde
20	Karin	Kleinert	Neue Str.	15	98745	Kleiner Ort	12368789954	Kleinert.Karin@web.de	

Abb. 10 Tabelle Kunde in phpMyAdmin

The screenshot shows the 'KunstKiste' website backend. At the top, there is a navigation bar with links: Home, Kurse, Techniken A-Z, Künstlerportraits, Über uns, Kontakt, Impressum, and Backend. Below the navigation bar is a table with the following columns: Vorname, Nachname, Straße, HNr, PLZ, Ort, Telefon, Bemerkung, and Action. The table contains 10 rows of customer data. Below the table is a section titled 'Neuer Kunde' with a form to add a new customer. The form has three input fields: Vorname, Name, and Straße.

Vorname	Nachname	Straße	HNr	PLZ	Ort	Telefon	Bemerkung	Action
Otto	Normalverbraucher	Hauptstr.	10	12345	Musterhausen	01234/789654	Das ist die erste Bemerkung beim ersten Kunden	bearbeiten ▶
Klara	Extravagant	Superweg	1	98765	Spezialstadt	0121/445566		bearbeiten ▶
Klaus	Normalverbraucher	Hauptstr.	10	12345	Musterhausen	01234/789654	Sohn von Otto Normalverbraucher	bearbeiten ▶
Silvia	Extravagant	Superweg	1	98765	Spezialstadt	0121/445566	Tochter von Klara Extravagant	bearbeiten ▶
Hans	GuckindieLuft	Hochweg	3	223366	Himmel	1236654778899		bearbeiten ▶
Erster	Neuer	weg	3	1235654	Hier	123654		bearbeiten ▶
Martin	Mustermann	Schaustr.	3	55892	Stadt	1236654778899		bearbeiten ▶
Lisa	Neuer	Alterweg	5	1235654	Hier	123654	Montag	bearbeiten ▶
Rainer	Kleinert	Neue Str.	15	98745	Kleiner Ort	12368/789954	Hallo!	bearbeiten ▶
Lisa	Logisch	Neuweg	3	223366	Großstadt	1236654778899	nochmal ein angelegter Kunde	bearbeiten ▶
Karin	Kleinert	Neue Str.	15	98745	Kleiner Ort	12368789954		bearbeiten ▶

Neuer Kunde

Vorname:

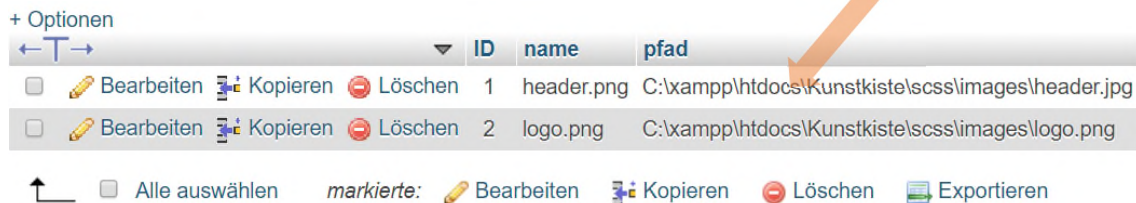
Name:

Straße:

Abb. 11 Tabelle Kunde im Backend der WebsiteDatenbank-Übersicht

3.9 Relationale Datenbank

Für dieses Projekt wurde eine relationale Datenbank angelegt. Da die benötigten Daten für diese Applikation, alle festen DatenTypen (wie z.B. „integer“ für die IDs, oder



+ Optionen

	ID	name	pfad
<input type="checkbox"/> Bearbeiten <input type="checkbox"/> Kopieren <input type="checkbox"/> Löschen	1	header.png	C:\xampp\htdocs\Kunstliste\scss\images\header.jpg
<input type="checkbox"/> Bearbeiten <input type="checkbox"/> Kopieren <input type="checkbox"/> Löschen	2	logo.png	C:\xampp\htdocs\Kunstliste\scss\images\logo.png

↑ ☐ Alle auswählen markierte: ☐ Bearbeiten ☐ Kopieren ☐ Löschen ☐ Exportieren

Abb. 12 Bildpfad in der Bild-Tabelle

„date“ für ein Datum) zugeordnet werden können. Für eingefügte Bilder im KursDesigner z.B. wird nur der Pfad zum Bild und der Bildname in der Datenbank gespeichert. Hierfür wurde eine Hilfstabelle „Bild“ angelegt in der nur diese beiden Datentypen gespeichert und einer ID zugeordnet werden. Die ID wird als Verbindung zur Tabelle Kurs und zur eindeutigen Zuordnung eines Bildes Im KursDesigner benötigt. Für dieses Projekt hätte auch eine Objektrelationale Datenbank angelegt werden können, hierauf wurde jedoch verzichtet, weil nicht vorgesehen ist die kompletten Bilder oder Audio-Dateien und Textdateien in der Datenbank zu speichern.

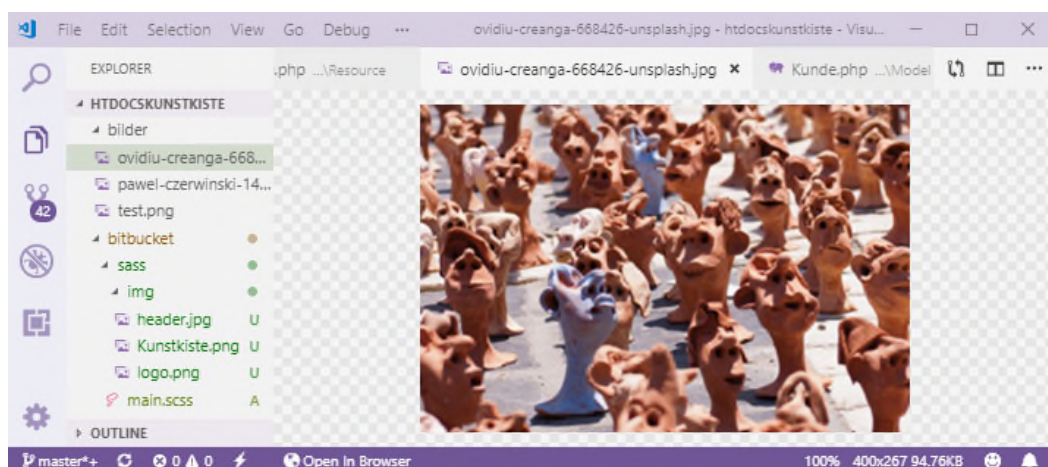


Abb. 13 Bilder-Ordner und img-Ordner

Die Bilder werden direkt im Projekt gespeichert. Laut Konvention wird dafür ein separater Ordner „Bilder“, für Bilder die in Beiträgen verwendet, im Projekt angelegt. Bilder die zur Websitegestaltung verwendet werden wie z.B. header.jpg werden im SASS-Ordner, Unter-Ordner „img“ abgelegt.

3.10 Anpassungen

Im Verlauf des Projektes haben sich teilweise die ursprünglich geplanten Tabellenspalten oder die erforderlichen Labels und Eingabefelder geändert, weil sie sich als nicht ideal erwiesen haben. Auch Layouts mussten immer wieder angepasst werden.

Die größte Anpassung waren die „Speichern-, Ändern- und Löschen-Buttons“, sie waren ursprünglich 1x pro Benutzeroberfläche geplant und mussten am Ende aber in jeder Tabellenzeile eingefügt werden, weil sonst die Zuordnung der Funktion zu den Daten schwieriger gewesen wäre, man muss ja wissen, was gelöscht oder geändert werden soll. Die erste Idee zur Kundenerfassung war so:

The sketch shows a form for managing customer data. It features a title 'Kunde' and a 'Kunde anlegen' button. The form includes several input fields: 'Vorname' and 'Name' for the customer's name, 'Email-Adresse' for the email, 'Bemerkungen' for notes, 'Adresse' for the address (with sub-labels 'Straße, Haus-Nr., PLZ, Ort'), 'Telefonnummer' for the phone number (with a placeholder 'XXX XXXXXXXXX'), and 'Kunde-ID PK' for the customer ID. At the bottom of the form are four buttons: 'ändern', 'speichern', 'löschen', and 'abbrechen'.

Abb. 14 Skizze Benutzeroberfläche Kunde

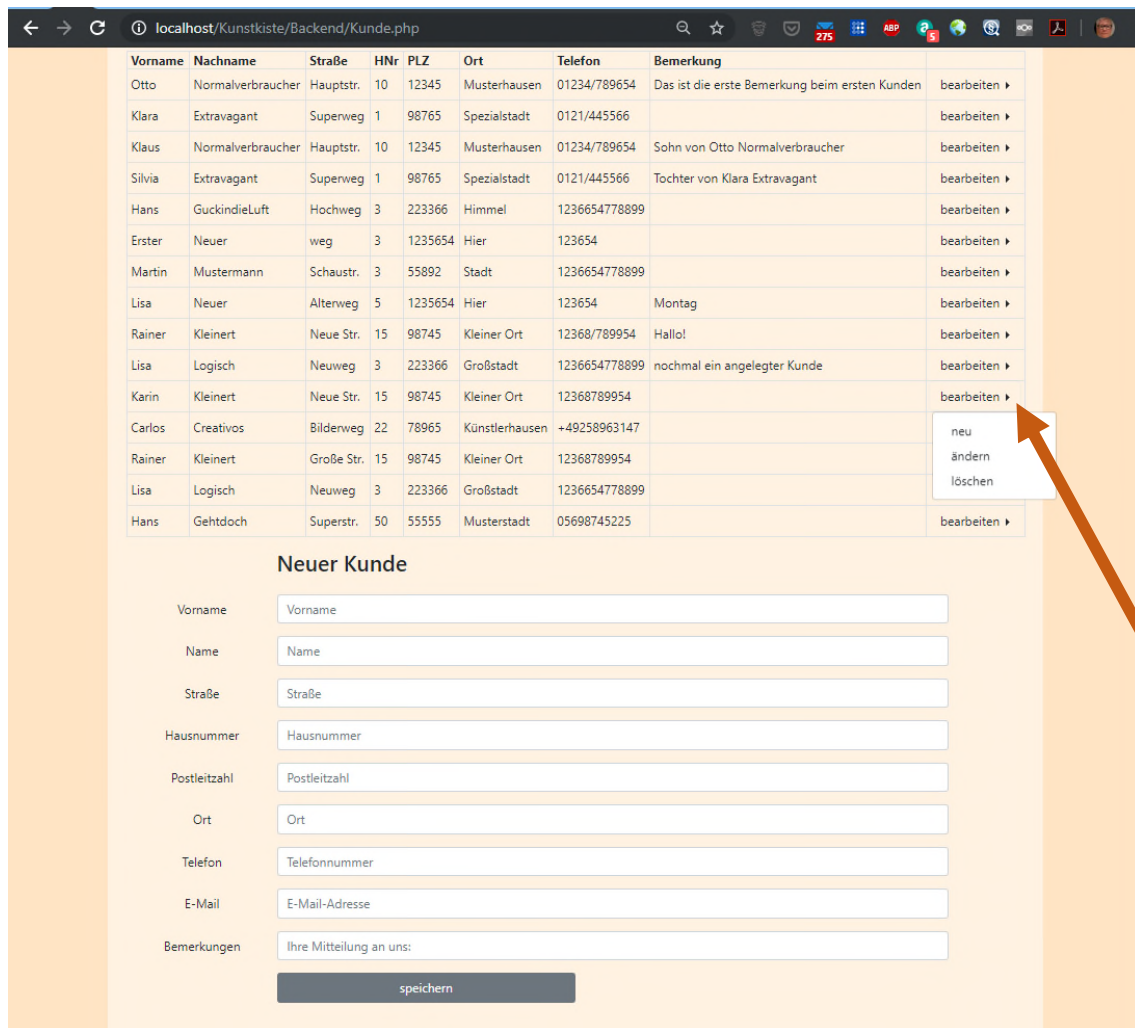


Abb. 15 Benutzeroberfläche Kunden endgültige Seite

Am Ende sah das Formular so aus und jede Tabellenzeile bekam ein Dropdown mit Namen „Bearbeiten“ durch das die entsprechenden Funktionen dann in jeder Zeile zur Verfügung stehen.

3.11 Datenbankzugriff per PDO

Um Benutzeroberflächen mit der Datenbank zu verbinden muss man eine Datenverbindung aufbauen. Die aktuell gebräuchteste Methode dafür ist PDO.

3.11.1 Verbindungsaufbau

Für das Verbinden mit der Datenbank gibt es in PHP eine Basisklasse PDO. Der Konstruktor der Klasse erwartet die Parameter: Datenbank-Quelle, Benutzer und Passwort. Für die Datenbank-Quelle wird der Hostname und der Datenbankname erwartet, evtl. kann man noch die Codierung mit aufnehmen.

Deswegen wurde in der Klasse Base die Methode connect() mit einer Instanz von PDO und den entsprechenden Parametern implementiert.

```
class Base
{
    protected function connect()
    {
        $dataSource = new \PDO(
            'mysql:host=localhost;dbname=kunstkiste;charset=utf8',
            'root', ''
        );
        return $dataSource;
    }
}
```

Abb. 16 Verbindungsaufbau mit PDO

Wann immer im Projekt eine Datenbankverbindung benötigt wird kann diese Methode verwendet werden.

3.11.2 Verbindungsfehler

Wenn PDO keine Verbindung zur Datenbank aufbauen kann, wird eine *PDOException* geworfen, diese beendet das Script und gibt die zum Verbindungsversuch mitgegebenen Parameter aus.

Da zu diesen Parametern auch der Benutzername und das Passwort gehört, muss diese Ausgabe auf jeden Fall verhindert werden. Hier wurde das mit einen Try-Catch-Block realisiert. Dieser versucht eine Verbindung aufzubauen (try) und fängt (catch) wenn das nicht klappt, den Fehler mit einer allgemeinen Fehlermeldung ab.

```
3 namespace Model\Resource;
4
5 class Base
6 {
7     protected function connect()
8     {
9         try {
10             $dataSource = new \PDO(
11                 'mysql:host=localhost;dbname=kunstkiste;charset=utf8',
12                 'root', ''
13             );
14             return $dataSource;
15         }
16         catch(\PDOException $Exception) {
17             throw new MyDatabaseException($Exception->getMessage(),
18                 (int)$Exception->getCode());
19         }
20     }
21 }
22
```

Abb. 17 PDO mit Try-Catch-Block

4 Projektarchitektur

4.1 MVC-Pattern

Dieses Projektes wurde nach dem Model-View-Controller Schema implementiert.

4.2 Das Model

```
namespace Model\Resource;

class Kunde extends Base
{
    public function insert(string $vorname, string $nachname, string $str, string $hnr,
    int $plz, string $ort, string $tel, string $email, string $text)
    {
        $sql = "
        INSERT INTO kunde (vorname, nachname, str, hnr, plz, ort, email, bemerkung)
        VALUES (:vorname, :nachname, :str, :hnr, :plz, :ort, :tel, :email, :text)";

        $connection = $this->connect();
        $statement = $connection->prepare($sql);

        $statement->bindValue(':vorname', $vorname);
        $statement->bindValue(':nachname', $nachname);
        $statement->bindValue(':str', $str);
        $statement->bindValue(':hnr', $hnr);
        $statement->bindValue(':plz', $plz);
        $statement->bindValue(':ort', $ort);
        $statement->bindValue(':tel', $tel);
        $statement->bindValue(':email', $email);
        $statement->bindValue(':bemerkung', $text);

        $statement->execute();

        return $connection->lastInsertId();
    }

    public function getKunde()
    {
        $connection = $this->connect();
        $Sql = sprintf("
        SELECT id, vorname, nachname, str, hnr, plz, ort, tel, email, bemerkung
        FROM kunde");
        $dbResult = $this->connect()->query($Sql);

        $kunde = [];

        while ($row = $dbResult->fetch(\PDO::FETCH_ASSOC))
        {
            /**@var \Model\Kunde $kunde */
            $kunde = \App::getModel('Kunde');
            $kunde->setId($row['id']);
            $kunde->setVorname($row['vorname']);
            $kunde->setNachname($row['nachname']);
            $kunde->setStr($row['str']);
            $kunde->setHnr($row['hnr']);
        }
    }
}
```

Abb. 18 Model/Resource/Kunde.php

In der Datei Kunde.php in Model/Resource (S. 26 Abb. 17) ist die Klasse Kunde implementiert. Diese Klasse erbt von der Klasse Base die Funktion connect() und kann diese somit einfach aufrufen und verwenden. Damit in den dort implementierten Funktionen „getKunde“ oder „insert“ auch Daten ankommen, bzw. weitergegeben werden können, braucht es die „Getter“ und „Setter“ aus Model/Kunde.php - siehe Abb. 18.

```
namespace Model;

class Kunde {
    private $_id = 0;
    private $_vorname = "";
    private $_nachname = "";
    private $_str = "";
    private $_hnr = "";
    private $_plz = "";
    private $_ort = "";
    private $_tel = "";
    private $_email = "";
    private $_text = "";
    private $_benutzer_id = "";

    /**
     * @return int
     */
    public function getId()
    {
        return $this->_id;
    }

    /**
     * @param int $id
     */
    public function setId($id)
    {
        $this->_id = $id;
    }

    /**
     * @return string
     */
    public function getVorname()
    {
        return $this->_vorname;
    }

    /**
     * @param string $name
     */
}
```

Abb. 19 Model/Kunde.php

4.3 View

Der Konstruktor in der Klasse Template (Namespace View) lädt die Variable „\$tplFile“ mit dem Seiteninhalt der sich zur Laufzeit im aktuellen Moment auf der Seite befindet.

Die Methode „renderTemplate“ holt sich dazu noch die vom Controller registrierten Benutzereingaben oder Benutzerabfragen zur Laufzeit und die aus SASS zu diesem Moment compilierte CSS-Datei. Vom so generierten Quellcode wird dann die Seite im Browser ausgegeben.

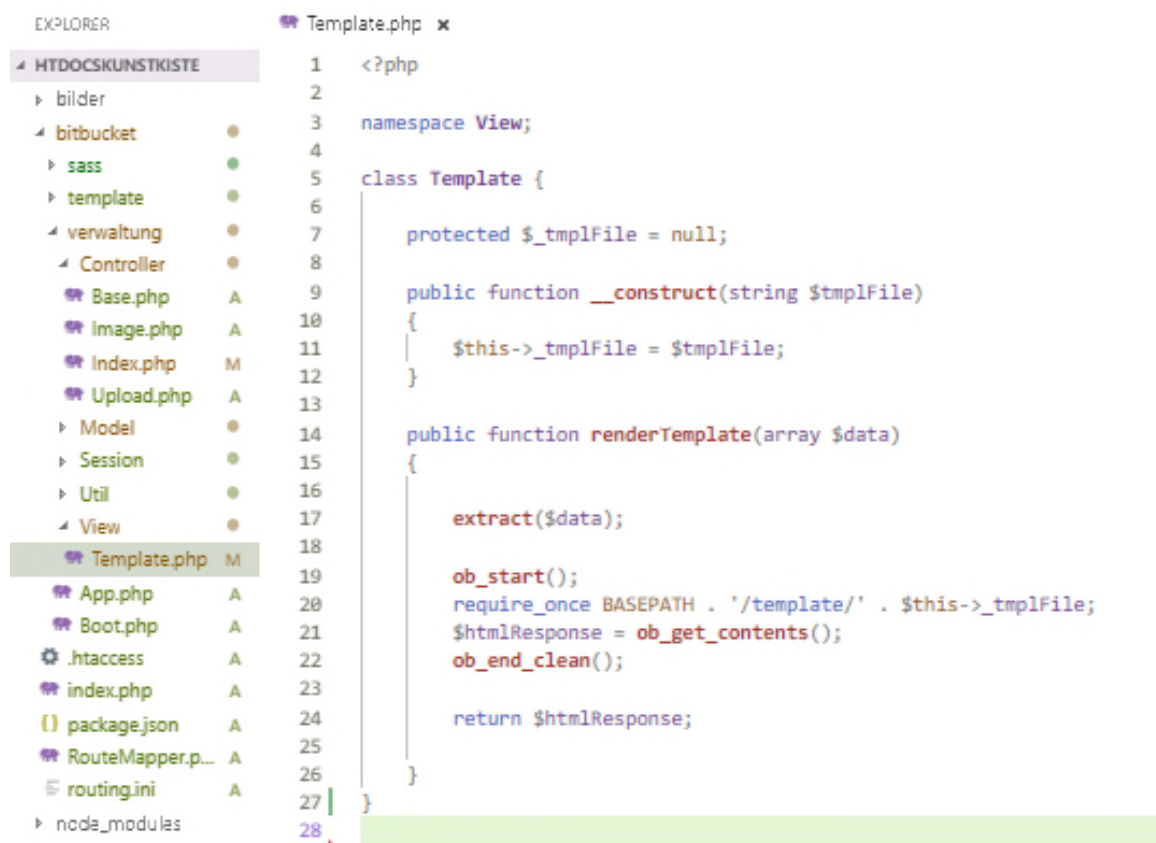


Abb. 20 Namespace View

4.4 Controller

Der Controller reagiert auf das was der Benutzer auf der Web-Oberfläche eingibt, abfragt oder klickt.

```
3 namespace Controller;
4
5 use Model\Benutzer;
6 use Model\Bild;
7 use Model\Kunde;
8 use Session\User;
9
10 class Index extends Base {
11
12     public function indexAction($params)
13     {
14         // resource model instanzieren
15         /** @var \Model\Resource\Bild $model */
16         $model = \App::getResourceModel('Bild');
17         // bilder abrufen
18         $bilder = $model->getBilder();
19         //bilder darstellen / template
20         echo $this->render('bilder.php', array('bilder' => $bilder));
21     }
22
23     public function kundeAction($params)
24     {
25         /** @var \Model\Resource\Kunde $kunde */
26         $model = \App::getResourceModel('Kunde');
27
28         $kunde = $model->getKunde();
29
30         echo($this->render('kunde.php', array('kunde' => $kunde)));
31     }
32
33     public function kurseAction($params)
34     {
35         /** @var \Model\Resource\Kurse $kurse */
36         $model = \App::getResourceModel('Kurse');
37
38         $kunde = $model->getKurse();
39
40         echo($this->render('kurse.php', array('kurse' => $kurse)));
41     }
42 }
```

Abb. 21 Controller/Index.php

5 Projektabschluss

5.1 Qualitätssicherung

Alle Komponenten, Links und Funktionen wurden während der Entwicklung und zum Abschluss laufend getestet. Hier die Abschlussdokumentation:

1. index.php, automatischer Aufruf _____ o.k.
2. header.php _____ o.k.
 - a. Einbindung in index.php _____ o.k.
 - b. Einbindung in Anmeldung.php _____ o.k.
 - c. Einbindung in Kunde.php _____ o.k.
 - d. Einbindung in KursDesign.php _____ o.k.
 - e. Einbindung in Kurse.php _____ o.k.
 - f. Einbindung in Teilnehmer.php _____ o.k.
3. Link zu process.php
 - a. Buttons in index.php _____ o.k.
 - b. Buttons in Anmeldung.php _____ o.k.
 - c. Buttons in Kunde.php _____ o.k.
 - d. Buttons in KursDesign.php _____ o.k.
 - e. Buttons in Kurse.php _____ o.k.
 - f. Buttons in Teilnehmer.php _____ o.k.
4. Rück-Link von process.php zu den Dateien
 - a. zu index.php _____ o.k.
 - b. zu Anmeldung.php _____ o.k.
 - c. zu Kunde.php _____ o.k.
 - d. zu KursDesign.php _____ o.k.
 - e. zu Kurse.php _____ o.k.
 - f. zu Teilnehmer.php _____ o.k.

5. Anmelden-Link von index.php
 - a. zu Anmeldung.php _____ o.k.
6. Auslesen der Datenbanktabellen
 - a. Select zu Kunde.php _____ o.k.
 - b. Select zu Kurse.php _____ o.k.
 - c. Select zu Teilnehmer.php _____ o.k.
7. Schreiben in Datenbanktabellen
 - a. Post von Anmeldung.php zu Tabelle _____ o.k.
 - b. Post und Update von Kunde.php zu Tabelle _____ o.k.
 - c. Post und Update von KursDesign.php zu Tabelle _____ o.k.
 - d. Post und Update von Kurse.php zu Tabelle _____ o.k.
 - e. Post und Update von Teilnehmer.php zu Tabelle _____ o.k.
8. Zeile löschen aus Datenbanktabellen
 - a. Delete von Kunde.php zu Tabelle _____ o.k.
 - b. Delete von KursDesign.php zu Tabelle _____ o.k.
 - c. Delete von Kurse.php zu Tabelle _____ o.k.
 - d. Delete von Teilnehmer.php zu Tabelle _____ o.k.

5.2 Soll-Ist-Vergleich – erreichte Ziele

Die Website Kunstkiste hat nun eine zusätzliche Seite zur Veröffentlichung von Kunstkursen. Diese können vom Besucher dieser Seite auch direkt gebucht werden. Es wurde, wie gefordert, ein Backend zur Verwaltung einer Datenbank, in der Kurse, Kunden und deren Teilnahme in den Kursen gespeichert werden können, entwickelt. Die Inhaber der Kunstkiste haben nun gute Chancen mit diesem neuen Geschäftszweig neue Kunden und mehr Umsatz zu erzielen.

Das Projekt konnte gut im geplanten Rahmen umgesetzt werden. Allerdings erscheint im Nachhinein die Implementierung eines MVC-Patterns für eine Web-Applikation mit einem relativ geringen Funktionsumfang ein bisschen überdimensioniert. Wenn keine Erweiterungen geplant sind, hätte es wahrscheinlich ausgereicht, eine Prozess-Datei als Funktionslogik für die benötigten Benutzeroberflächen zu implementieren.

5.3 Erweiterbarkeit

Da dieses Projekt bereits objektorientiert angelegt ist. Kann diese Konstellation schon als Basis für eine Shop-Implementierung dienen, falls irgendwann die Erweiterung zu einem Onlineshop geplant würde. Auch eine Erweiterung der Datenbank als Basis für ein Email-Marketing sollte kein Problem sein. Es sind also durchaus noch Folgeaufträge zu erwarten.

5.4 Abweichungen

Da dies ein fiktives Projekt ist, wurde das Backend über einen Menü-Punkt zugänglich gemacht. Dies würde man bei einem reellen Projekt natürlich nicht machen. Da müsste dann die Backend-Adresse über den Browser eingegeben und mit Passwort geschützt werden.

5.5 Benötigte Projektzeiten

Die tatsächlich benötigte Zeit weicht teilweise von der Planung ab. Da das Projekt jedoch innerhalb der Planzeit realisiert werden konnte, hat das für den Kunden keine Auswirkungen.

Bezeichnung	Soll	Ist
Bedarfsanalyse gemäß Anforderung	2 Std	3 Std
Planungsdiagramme + Skizzen Benutzeroberfläche	10 Std	8 Std
Entwicklungsumgebung einrichten	2 Std	1 Std
Datenbank mit Testdaten	4 Std	2 Std
Benutzeroberflächen entwickeln	16 Std	18 Std
Entwickeln + Implementieren der Software	24 Std	26 Std
Testläufe und Fehlerkorrektur	4 Std	4 Std
Kundendokumentation	8 Std	8 Std
Gesamtzeit	70 Std	70 Std

Tab. 3 Soll-Ist-Zeiten-Vergleich

5.6 Fazit

Im Rückblick war es eine gute Entscheidung, die Erweiterung für die Website der Kunstkiste als selbst entwickeltes Projekt zu planen. Die Umsetzung war am Ende wesentlich einfacher als zunächst gedacht.

Sicher hätte man auch mit Contentmanagement-System diese Funktionalität erreichen können. Vermutlich wäre die Benutzeroberfläche zur Erstellung von Beiträgen für Kunstkurse schon verfügbar gewesen, dafür hätte man aber alle anderen Inhalte der Website übertragen müssen. Die Datenbank hätte man auch hierfür anlegen und anbinden müssen.

Quellen und Lernmittel

<https://code.visualstudio.com/download> (am 1.4.2019)

<https://www.apachefriends.org/de/download.html> (am 1.4.2019)

<https://www.w3schools.com> (am 2.4.2019)

<https://www.w3.org> (am 2.4.2019)

<https://www.php-einfach.de> (am 4.4.2019)

<https://php.net/manual/de> (am 4.4.2019)

<https://www.udemy.com/objektorientierte-entwicklung-php7>

(Trainer: Jan Brinkman, am 1.4.2019)

<https://www.udemy.com/advanced-css-and-sass>

(Trainer Jonas Schmedtmann, am 1.4.2019)

PHP 7 u. MySQL, Buch v. Tobias Hauser, Christian Wenz (Rheinwerk Verlag 2018)

Einstieg in PHP 7 und MySQL, von Thomas Theis (Rheinwerk Verlag 2017)

<https://unsplash.com> Tonskulpturen-Photo by Ovidiu Creanga

<https://unsplash.com> Photo von Pawel-Czerwinski 1475179-unsplash.jpg

Glossar

Backend	I.d.R. durch Passwort und Authentifizierung geschützter Website-Bereich bzw. wie hier Web-Applikation, die eine Bearbeitung der Seite ermöglicht oder Funktionen, Verwaltungsmodule o.ä. bereit stellt
Bootstrap	Framework für einfachere u. schnelle Entwicklung im Web
ERM	Entity-Relationship-Modell, Datenmodell zur grafischen Darstellung von Datenbanken
Exception	PHP-Basis-Klasse zur Behandlung von Ausnahmen oder eventuellen Fehlerquellen
Frontend	Für alle Website-Besucher öffentliche Frontansicht einer Seite
GNU-Lizenz	Allgemeine Veröffentlichungserlaubnis o. -genehmigung
MIT-Lizenz	Open-Source-Lizenz, aus dem Massachusetts Institute of Technology
Open Source	Bezeichnet Software, die mit offenem, frei verfügbarem Quellcode, von vielen Entwickler gemeinsam entwickelt wird und kostenlos zur Verfügung steht
PDO	PHP Data Objects, PHP-Basis-Klasse die Datenbankverbindungen ermöglicht
SASS	Syntactically Awesom Stylesheets, Sprache zur Gestaltung von Websites, CSS-Präprozessor mit Variablen, Schleifen usw.
UML	Unified Modeling Language, hier die Spezialisierung als Anwendungsfall-Diagramm verwendet

Anlagen

Projektantrag

Kundendokumentation

Projektantrag

Teilnehmer/in: Anja Kraus	Gruppe: IT17-02	Ausbildungsberuf: FIA
Projektbezeichnung: Webapplikation zur Verwaltung von Kunstkursen		
Projektbeschreibung (Ist-/Sollzustand, Inhalte, erwartete Ergebnisse in Kurzform): <p>Teil 1: Projektumfeld und Ist-Zustand Die „KunstKiste“ ist ein kleines Geschäft für Künstlerbedarf auf dem Lande. Die Inhaber konnten bisher ganz gut davon leben, aber sie sind schon älter und auch die Kundschaft zeigt schon einigen Altersschwund.</p> <p>Teil 2: Problem Die Stammkundschaft wird zunehmend weniger. So wird ein Konzept benötigt, mit dessen Hilfe neue Kunden erreicht werden könnten, denn nur wenige junge Künstler finden den Weg ins Geschäft. Aktuell gibt es nur eine einfache, statische Website ohne Marketingkonzept. Eine Übergabe des Geschäftes an die nächste Generation wäre so nicht mehr rentabel.</p> <p>Teil 3: Soll-Zustand Um mehr Kunden zu erreichen, möchte die Tochter der Inhaber Kunst-Kurse anbieten. Zur Verwaltung dieser Kurse braucht sie eine Web-Applikation mit folgenden Funktionen:</p> <ul style="list-style-type: none"> <input type="checkbox"/> Buchungsmöglichkeit für Kunden <input type="checkbox"/> Automatische Buchungsbestätigung per Mail <input type="checkbox"/> Buchungen verwalten <input type="checkbox"/> Kunden verwalten <input type="checkbox"/> Kurstermine verwalten <input type="checkbox"/> Rundmail an Teilnehmer eines Kurses <p>Die Verwaltung soll vom Backend der Website zugänglich sein und die Daten in einer MySQL-Datenbank gespeichert werden.</p> <p>Teil 4: Inhalte Es wird eine My SQL-Datenbank für die Daten und Zusammenhänge erstellt. Mit PHP soll nach dem MVC Modell eine Webapplikation entwickelt werden, die Kunden- und Buchungsdaten in die Datenbank schreibt, bzw. daraus ausliest und im Backend der Firmenwebsite ausgibt.</p>		

Projektablauf / Gliederung des Projektablaufs incl. Zeitplanung:

Datenbank Konzeption entwickeln	8 Std.
Backend Konzeption entwickeln	8 Std.
Datenbank installieren und Tabellen anlegen	8 Std.
Backendseite erstellen und gestalten	8 Std.
Klassen, Php-Funktionen und Datenbankzugriffe entwickeln	16 Std.
Alle Funktionen testen	6 Std.
 Projektdokumentation, Kundendokumentation	16 Std.

Gesamter Zeitaufwand in Stunden: 70

Benötigtes Material:

Das Projekt soll in Visual-Studio-Code erstellt werden, die Datenbanken sollen mit Xampp bzw. phpMyAdmin angelegt werden. Xampp fungiert auch als lokaler Test-Server.

Die Anwendungen sind OpenSource,

da auch ein Windows-PC zum Entwickeln der Webapplikation und Speichern der Datenbank vorhanden ist, entstehen keine Kosten.

Ort, Datum, Unterschrift:

.....



Kundendokumentation

Web-Applikation zur
Verwaltung von Kunstkursen

Information zu
Komponenten + Funktionen
und Leitfaden zur Benutzung

Verfasst von:

Anja Kraus

Fachinformatikerin Anwendungsentwicklung

Berufsförderungswerk Schöenberg gGmbH

Inhaltsverzeichnis

1 Übergabe.....	3
2 Leistungsumfang	4
2.1 Realisierung	4
2.2 Erweiterbarkeit.....	4
3 Die Seite Kunstkurse.....	5
4 Anmeldung für Kurse	6
5 Kurs-Designer	8
6 Teilnehmer in Kurse buchen	9

Abbildungsverzeichnis

Abb. 1 Seite Kunstkurse	5
Abb. 2 Anmeldeformular für Kunstkurse	6
Abb. 3 Benutzeroberfläche „Neuer Kunde“	7
Abb. 4 Maske KursDesigner	8
Abb. 5 Maske Teilnehmer in Kurs einbuchen	9

1 Übergabe

Mit Inbetriebnahme Ihrer neuen Webablikation möchte ich mich ganz herzlich für den tollen Auftrag und die gute Zusammenarbeit bedanken.

Ich habe Ihnen hier die wichtigsten Eigenschaften und Funktionen zusammen gestellt und einen kleinen Leitfaden zur Benutzung verfasst.

Sollten dennoch Fragen bleiben oder auch mal Probleme auftauchen, finden Sie hier die Kontaktdaten um mich zu erreichen:

Anja Kraus

Fachinformatikerin Anwendungsentwicklung

www.anjakraus.de

mail@anjakraus.de

0173/000 00 00

Viel Erfolg mit Ihrem neuen Geschäftszweig und

Freude bei der Benutzung Ihrer neuen Web-Applikation.

2 Leistungsumfang

Übersicht der Anforderungen:

- Frontend -> für die Besucher Ihrer Website sichtbar
 - Erstellung der Unterseite Kunstkurse und Einbindung in Website
 - Anmeldeformular für Kunstkurse
- Backend -> Verwaltungsseiten für Mitarbeiter
 - Editor zur Erfassung Beiträgen von für die Seite Kunstkurse
 - Verwaltungsoberfläche für alle Kunden, alle Kunstkurse, die Kursteilnehmer
 - Datenbank zur Speicherung aller entstehenden Daten
 - Mailfunktion an alle Teilnehmer eines Kurses

2.1 Realisierung

Erfreulicherweise waren alle Anforderungen wie gewünscht umsetzbar, auch der zeitliche Rahmen konnte eingehalten werden. Dadurch entspricht auch die separat versandte Rechnung exakt dem vorab erstellten Angebot.

2.2 Erweiterbarkeit

Dieses Projekt wurde für Sie so angelegt, dass viele Erweiterungsmöglichkeiten gegeben sind. Da Ihre Kunden nun sowie so schon online mit Ihnen in Kontakt treten, könnte bei Ihren Kunden irgendwann der Wunsch nach einem Online-Shop entstehen. Es wäre jederzeit problemlos möglich, die entsprechenden Komponenten zu entwickeln oder ein bereits fertiges Shopsystem anzubinden. Kommen Sie dazu gerne jederzeit auf mich zu. Auch eine Erweiterung der Datenbank als Basis für ein Email-Marketing ist jederzeit möglich. Vielleicht haben Sie ja selbst auch noch ganz andere Ideen?

3 Die Seite Kunstkurse

Als ersten Arbeitsschritt und gleichzeitig als Einstiegsseite für Ihre Website-besucher wurde die Seite für Kunstkurse eingerichtet und gestaltet. Hier werden Sie zukünftig alle Ihre Kurse wieder finden und können selbst neue einstellen.



Abb. 1 Seite Kunstkurse

Ihre Kunden können Ihnen hier direkt ihr Interesse an einem Kurs mitteilen, indem Sie auf den Button: „Hier zum Kurs anmelden“ klicken (siehe Pfeil).

Dabei öffnet sich das auf der nächsten Seite dargestellte Anmeldeformular.

4 Anmeldung für Kurse

Hier können Kursinteressenten ihre Daten eintragen und wenn sie wollen, auch eine Mitteilung schreiben. Mit klicken auf den Button „senden“ werden diese Daten in die Datenbank gespeichert. Gleichzeitig bekommen Sie eine Email, mit der Info: „Kunde XXX hat sich gerade für den Kurs „XXX“ angemeldet“ (dabei wird der echte Kundenname und Kursname eingefügt). Auch der Kunde bekommt automatische eine Email, die ihn informiert, dass er als Interessent für den entsprechenden Kurs eingetragen wird.

Je nach Anzahl der Interessenten und der Plätze im Kurs müssen Sie sich dann natürlich noch abstimmen, ob eine Teilnahme am Kurs möglich ist bzw. ob der Kurs überhaupt stattfinden kann, aber das wäre ja auch ohne Online-Anmeldung immer der Fall.

The screenshot shows a web page with a dark navigation bar at the top containing links: Home, Kurse, Techniken A-Z, Künstlerportraits, Über uns, Kontakt, Impressum, and Backend. The main content area has a light orange background and is titled 'Anmeldung für den Kurs Kreative Köpfe'. Below the title, there is a welcome message and instructions. The registration form consists of several input fields: Vorname, Name, Straße, Hausnummer, Postleitzahl, Ort, Telefon, E-Mail, and a larger field for 'Bemerkungen' with the placeholder text 'Ihre Mitteilung an uns:'. A dark 'senden' button is located at the bottom of the form.

Home Kurse Techniken A-Z Künstlerportraits Über uns Kontakt Impressum Backend ▶

Anmeldung für den Kurs Kreative Köpfe

Wir freuen uns sehr über Ihr Interesse an unserem Kurs Kreative Köpfe! Sie können sich hier für den Kurs vormerken lassen.

Anschließend bekommen Sie eine Bestätigungsmail, dass Ihre Anfrage eingegangen ist.
Zeitnah werden Sie benachrichtigt ob noch Plätze frei sind und ob der Kurs stattfinden kann.

Bitte geben Sie im Formular Ihre Daten ein und drücken dann auf "senden".

Vorname

Name

Straße

Hausnummer

Postleitzahl

Ort

Telefon

E-Mail

Bemerkungen

Abb. 2 Anmeldeformular für Kunstkurse

Wenn sich mal ein Interessent telefonisch oder auch persönlich bei Ihnen meldet und an einem Kurs teilnehmen möchte, können Sie ihn (analog zur Anmeldemaske) in der Benutzeroberfläche „Neuer Kunde“ auch selbst eintragen.

Karin	Kleinert	Neue Str.	15	98745	Kleiner Ort	12368789954	bearbeiten ▶
Carlos	Creativos	Bildenweg	22	78965	Künstlerhausen	+49258963147	bearbeiten ▶
Rainer	Kleinert	Große Str.	15	98745	Kleiner Ort	12368789954	neu ändern löschen
Lisa	Logisch	Neuweg	3	223366	Großstadt	1236654778899	
Hans	Gehtdoch	Superstr.	50	55555	Musterstadt	05698745225	

Neuer Kunde

Vorname

Name

Straße

Hausnummer

Postleitzahl

Ort

Telefon

E-Mail

Bemerkungen

Abb. 3 Benutzeroberfläche „Neuer Kunde“

Die Daten aus beiden Masken finden Sie in der Tabelle auf der Seite Kunde (s.o.).

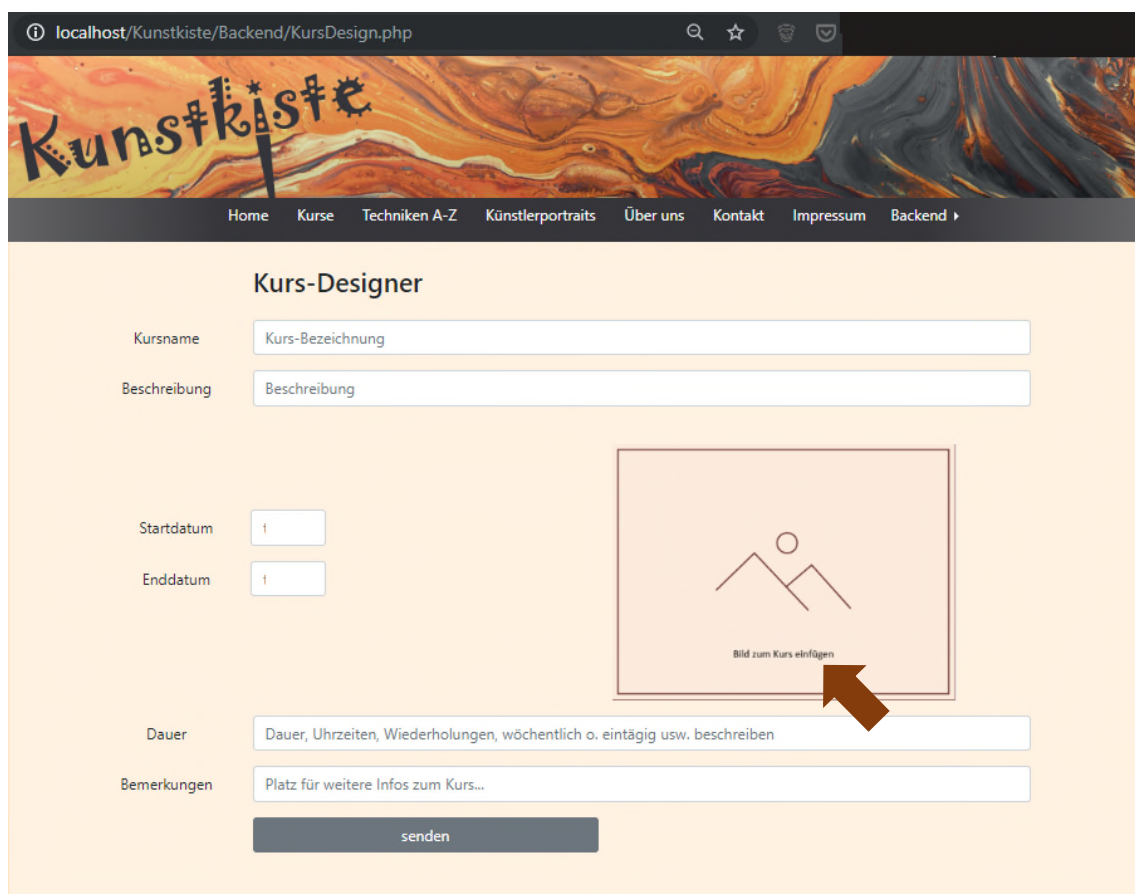
In jeder Zeile der Tabelle haben Sie jederzeit die Möglichkeit den Datensatz eines Kunde zu bearbeiten. Dazu klicken Sie in der letzten Spalte auf bearbeiten und Sie sehen die Schaltflächen neu, ändern, löschen und können die Daten bearbeiten. Sie sehen die Daten dann in der Maske Neuer Kunde bzw. wenn Sie auf löschen klicken, werden die Daten in der Zeile sofort gelöscht!

Achtung, diese Daten sind dann wirklich weg!

5 Kurs-Designer

Mit der Benutzeroberfläche KursDesigner sind Sie ab sofort in der Lage neue Kurse selbst zu erstellen!

Sie können, wie Sie sicher in der Maske schon sehen, den Kursnamen eintragen, eine Beschreibung des Kurses eingeben und alle anderen Daten und Infos eintragen.



The screenshot shows a web browser window with the address bar displaying 'localhost/Kunstkiste/Backend/KursDesign.php'. The page features a header with the 'Kunstkiste' logo and a navigation menu with links: Home, Kurse, Techniken A-Z, Künstlerportraits, Über uns, Kontakt, Impressum, and Backend. The main content area is titled 'Kurs-Designer' and contains a form with the following fields:

- Kursname:** A text input field with the placeholder 'Kurs-Bezeichnung'.
- Beschreibung:** A larger text input field with the placeholder 'Beschreibung'.
- Startdatum:** A date selection field with a calendar icon.
- Enddatum:** A date selection field with a calendar icon.
- Dauer:** A text input field with the placeholder 'Dauer, Uhrzeiten, Wiederholungen, wöchentlich o. eintägig usw. beschreiben'.
- Bemerkungen:** A text input field with the placeholder 'Platz für weitere Infos zum Kurs...'.

Below the 'Dauer' field is a large square area with a light orange background and a simple line drawing of a person standing next to a mountain. Below this drawing is the text 'Bild zum Kurs einfügen'. A large brown arrow points to this area. At the bottom of the form is a dark grey button labeled 'senden'.

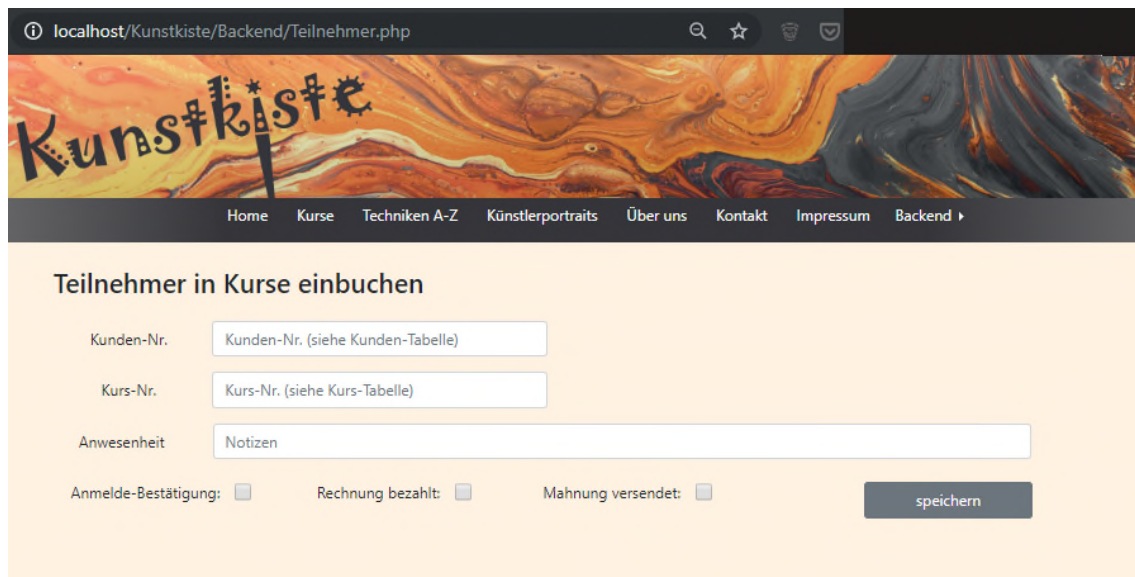
Abb. 4 Maske KursDesigner

Natürlich können Sie mit Klick auf die Schaltfläche „Bild zum Kurs einfügen“ auch ein passendes Bild platzieren.

6 Teilnehmer in Kurse buchen

Da nicht jeder Interessent automatisch einen Platz im gewünschten Kurs bekommt, müssen die Teilnehmer in die Kurse eingetragen werden.

Damit haben Sie es immer selbst in der Hand zu entscheiden, welche Teilnehmer welche Kurse besuchen. In der Übersicht über alle Teilnehmer finden Sie auch noch Möglichkeit für Rundmails, wenn es z.B. Terminverschiebungen gibt oder Sie gar einen Kurs absagen müssen.



The screenshot shows a web browser window with the address bar displaying 'localhost/KunstKiste/Backend/Teilnehmer.php'. The page features a header with the 'KunstKiste' logo and a navigation menu with links: Home, Kurse, Techniken A-Z, Künstlerportraits, Über uns, Kontakt, Impressum, and Backend. The main content area is titled 'Teilnehmer in Kurse einbuchen' and contains a form with the following fields and options:

- Kunden-Nr.:** A text input field with the placeholder text 'Kunden-Nr. (siehe Kunden-Tabelle)'.
- Kurs-Nr.:** A text input field with the placeholder text 'Kurs-Nr. (siehe Kurs-Tabelle)'.
- Anwesenheit:** A text input field with the placeholder text 'Notizen'.
- Anmelde-Bestätigung:** A checkbox.
- Rechnung bezahlt:** A checkbox.
- Mahnung versendet:** A checkbox.
- speichern:** A dark grey button.

Abb. 5 Maske Teilnehmer in Kurs einbuchen