

Technische Dokumentation

Projektarbeit im Modul "Webprogrammieren" - Schichtplan



Verfasser: Batliner Robert
Bösiger Matthias
Hug Sabrina
Müller Luca
Pascali Joanna
Paunovic Oliver
Schweizer Saskia
Schwendimann Nicola
Senn Melanie

Dozent: Dr. Hauser-Ehninger Ulrich

Studiengang: Digital Business Management Teilzeit 2017

Modul: Webprogrammierung

Ort, Datum: Chur, 15.06.2019

Inhalt

| | |
|---|----|
| 1. Datenbank | 3 |
| 1.1 Datenbank und Tabellen anlegen..... | 3 |
| 1.2 Tabellendefinition | 5 |
| 2. FTP Client einrichten | 6 |
| 2.1 PHPMyAdmin..... | 6 |
| 2.2 Schichtplan | 6 |
| 3. Javascript/Query Funktionsbeschreibung | 7 |
| 3.1 change_shift.js..... | 7 |
| 3.2 create_employee.js | 7 |
| 3.3 create_shift.js | 7 |
| 3.3.1 Von/-bis Zeit Schicht..... | 7 |
| 3.3.2 Zeitformat konvertieren | 8 |
| 3.4 create_workplace.js | 8 |
| 3.5 current_date.js | 8 |
| 3.6 delete_employee.js | 9 |
| 3.7 delete_shift.js | 9 |
| 3.8 delete_workplace.js | 9 |
| 3.9 empty_messages.js | 9 |
| 3.10 get_employees.js..... | 9 |
| 3.11 get_shift.js | 10 |
| 3.12 get_workplaces.js..... | 10 |
| 3.13 check_shift.js | 10 |
| 4. PHP Funktionsbeschreibung..... | 12 |
| 4.1 create_employee.php | 12 |
| 4.2 create_workplace.php..... | 12 |
| 4.3 create_shift.php | 13 |
| 4.4 delete_employee.php | 13 |
| 4.5 delete_workplace.php..... | 14 |
| 4.6 delete_shift.php | 14 |
| 4.7 get_employees.php..... | 15 |
| 4.8 get_workplaces.php | 15 |
| 4.9 get_shift.php | 16 |
| 4.10 update_shift.php | 16 |

1. Datenbank

1.1 Datenbank und Tabellen anlegen

1. HTW Chur Intranet → Campus → Web-/FTP Ausbildungsserver
2. Eine neue Webseite hinzufügen

**HTW Chur**
Hochschule für Technik und Wirtschaft
University of Applied Sciences

Angemeldet als:
batlinrobert

Deutsch
English

Kontakt | Webseite | Moodle | Downloads | Cash-Card | IT Support

Intranet
Home
Studium
Campus
Events an der HTW Chur
Sport
Respekt
Kinderbetreuung
Notfall
Vergünstigungen
Web-/FTP Ausbildungsserver
Erste Schritte
MySQL Remote-Access
HTW Card
Informatik
Bibliothek
Mitwirkung
Ideenmanagement
Qualitätsmanagement
Downloads

Web-/FTP Ausbildungsserver


Bitte beachten Sie unsere News zu diesem System:
[Zeichensatz-Codierung \(Sonderzeichen-Probleme\)](#)
[Verzeichnis-Passwortschutz](#)

Ausbildungsserver auf dem neuesten Stand


Wir haben den Ausbildungsserver auf einen aktuellen Stand gebracht:
MariaDB 10.1.23, PHP 7.0, Apache 2.4.25

Damit die bestehenden Applikationen nicht migriert werden müssen, laufen diese auf dem alten Server weiter (siehe weiter unten).
Neue Applikationen und Anpassungen können nur noch auf der neuen Umgebung gemacht werden.

NEU: MySQL-Datenbanken ausserhalb des Webserverns nutzen: → [mehr Infos](#)

Webseiten  Batliner Robert (batlinrobert) robert.batliner@stud.htwchur.ch
74185313 **Eine neue Webseite hinzufügen** 31.12.2019 

Alte Webs
Untenstehend sind die alten Webs zu finden. Diese können weiterhin benutzt werden. Nach dem Ablaufdatum werden diese gelöscht.
Neue Webs können nur noch auf dem oberen, neuen System erstellt und bearbeitet werden.

Webseiten  (batlinrobert)

3. Title, Löschdatum und Anzahl Datenbanken definieren

Benutzername

batlinrobert

Title

webprog

Löschdatum

25.06.2019

Datenbank

1

Erstellen

- Remote Zugriff aktivieren
- Anschliessend PHPMyAdmin aufrufen: <http://web1.fh-htwchur.ch/phpmyadmin/index.php>

Intranet

- Home
- Studium
- Campus**
 - Events an der HTW Chur
 - Sport
 - Respekt
 - Kinderbetreuung
 - Notfall
 - Vergünstigungen
 - Web-/FTP Ausbildungsserver
 - Erste Schritte
 - MySQL Remote-Access
 - HTW Card
- Informatik
- Bibliothek
- Mitwirkung
- Ideenmanagement
- Qualitätsmanagement
- Downloads

Web-/FTP Ausbildungsserver

Bitte beachten Sie unsere News zu diesem System:
[Zeichensatz-Codierung \(Sonderzeichen-Probleme\)](#)
[Verzeichnis-Passwortschutz](#)

Ausbildungsserver auf dem neuesten Stand

Wir haben den Ausbildungsserver auf einen aktuellen Stand gebracht:
MariaDB 10.1.23, PHP 7.0, Apache 2.4.25

Damit die bestehenden Applikationen nicht migriert werden müssen, laufen diese auf dem alten Server weiter (siehe weiter unten).
Neue Applikationen und Anpassungen können nur noch auf der neuen Umgebung gemacht werden.

NEU: MySQL-Datenbanken ausserhalb des Webservers nutzen: → [mehr Infos](#)

Webseiten Batliner Robert (batlinrobert) robert.batliner@stud.htwchur.ch

| 741853-1.web1.fh-htwchur.ch | | webp | 31.12.2019 | | |
|-----------------------------|--------------------------|--------------|------------|--|--|
| FTP-Benutzer | Host: web1.fh-htwchur.ch | | | | |
| | Benutzername | Passwort | | | |
| | | | | | |
| Datenbanken | Host: localhost | | | | |
| | DB-Name | Benutzername | Passwort | | |
| | | | | | |

Alte Webs
Untenstehend sind die alten Webs zu finden. Diese können weiterhin benutzt werden. Nach dem Ablaufdatum werden diese gelöscht.
Neue Webs können nur noch auf dem oberen, neuen System erstellt und bearbeitet werden.

Webseiten (batlinrobert)

- Datenbank auswählen
- Tabellen über Erzeuge Tabelle Dialog gemäss ERM anlegen
- Spalten pflegen (Name, Primary-Key, Auto-Increment und Unique)

The screenshot shows the phpMyAdmin interface. On the left, the database structure is visible, with '741853_1_1' selected. The main area displays a table list with columns: Tabelle, Aktion, Datensätze, Typ, Kollation, Größe, Überhang. The 'Erzeuge Tabelle' (Create Table) dialog is open, showing 'Name:' and 'Anzahl der Spalten: 4'.

1.2 Tabellendefinition

Tabelle Mitarbeiter:

Server: localhost:3306 » Datenbank: 741853_1_1 » Tabelle: mitarbeiter

Anzeigen Struktur SQL Suche Einfügen Exportieren Importieren Operationen Nachverfolgung Trigger

Tabellenstruktur Beziehungsansicht

| # | Name | Typ | Kollation | Attribute | Null | Standard | Kommentare | Extra | Aktion |
|---|-------|-------------|-----------------|-----------|------|----------|------------|----------------|---|
| 1 | id_ma | int(11) | | | Nein | kein(e) | | AUTO_INCREMENT | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Mehr |
| 2 | name | varchar(30) | utf8_general_ci | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Mehr |

Alle auswählen markierte: Anzeigen Bearbeiten Löschen Primärschlüssel Unique Index Zu zentralen Spalten hinzufügen Zentrale Spalten entfernen

Tabelle Arbeitsplatz:

Server: localhost:3306 » Datenbank: 741853_1_1 » Tabelle: arbeitsplatz

Anzeigen Struktur SQL Suche Einfügen Exportieren Importieren Operationen Nachverfolgung Trigger

Tabellenstruktur Beziehungsansicht

| # | Name | Typ | Kollation | Attribute | Null | Standard | Kommentare | Extra | Aktion |
|---|--------------|-------------|-----------------|-----------|------|----------|------------|----------------|---|
| 1 | id_ap | int(11) | | | Nein | kein(e) | | AUTO_INCREMENT | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Mehr |
| 2 | beschreibung | varchar(30) | utf8_general_ci | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Mehr |

Alle auswählen markierte: Anzeigen Bearbeiten Löschen Primärschlüssel Unique Index Zu zentralen Spalten hinzufügen Zentrale Spalten entfernen

Tabelle Schicht:

Server: localhost:3306 » Datenbank: 741853_1_1 » Tabelle: schicht

Anzeigen Struktur SQL Suche Einfügen Exportieren Importieren Operationen Nachverfolgung Trigger

Tabellenstruktur Beziehungsansicht

| # | Name | Typ | Kollation | Attribute | Null | Standard | Kommentare | Extra | Aktion |
|---|---------|---------|-----------|-----------|------|----------|------------|----------------|--|
| 1 | id_sp | int(11) | | | Nein | kein(e) | | AUTO_INCREMENT | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Zu zentralen Spalten hinzufügen |
| 2 | id_ma | int(11) | | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Zu zentralen Spalten hinzufügen |
| 3 | id_ap | int(11) | | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Zu zentralen Spalten hinzufügen |
| 4 | datum | date | | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Zu zentralen Spalten hinzufügen |
| 5 | zeitvon | time | | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Zu zentralen Spalten hinzufügen |
| 6 | zeitbis | time | | | Nein | kein(e) | | | Bearbeiten Löschen Primärschlüssel Unique Index Räumlich Volltext Unterschiedliche Werte Zu zentralen Spalten hinzufügen |

Alle auswählen markierte: Anzeigen Bearbeiten Löschen Primärschlüssel Unique Index Zu zentralen Spalten hinzufügen Zentrale Spalten entfernen

Foreign-Keys:

Server: localhost:3306 » Datenbank: 741853_1_1 » Tabelle: schicht

Anzeigen Struktur SQL Suche Einfügen Exportieren Importieren Operationen Nachverfolgung Trigger

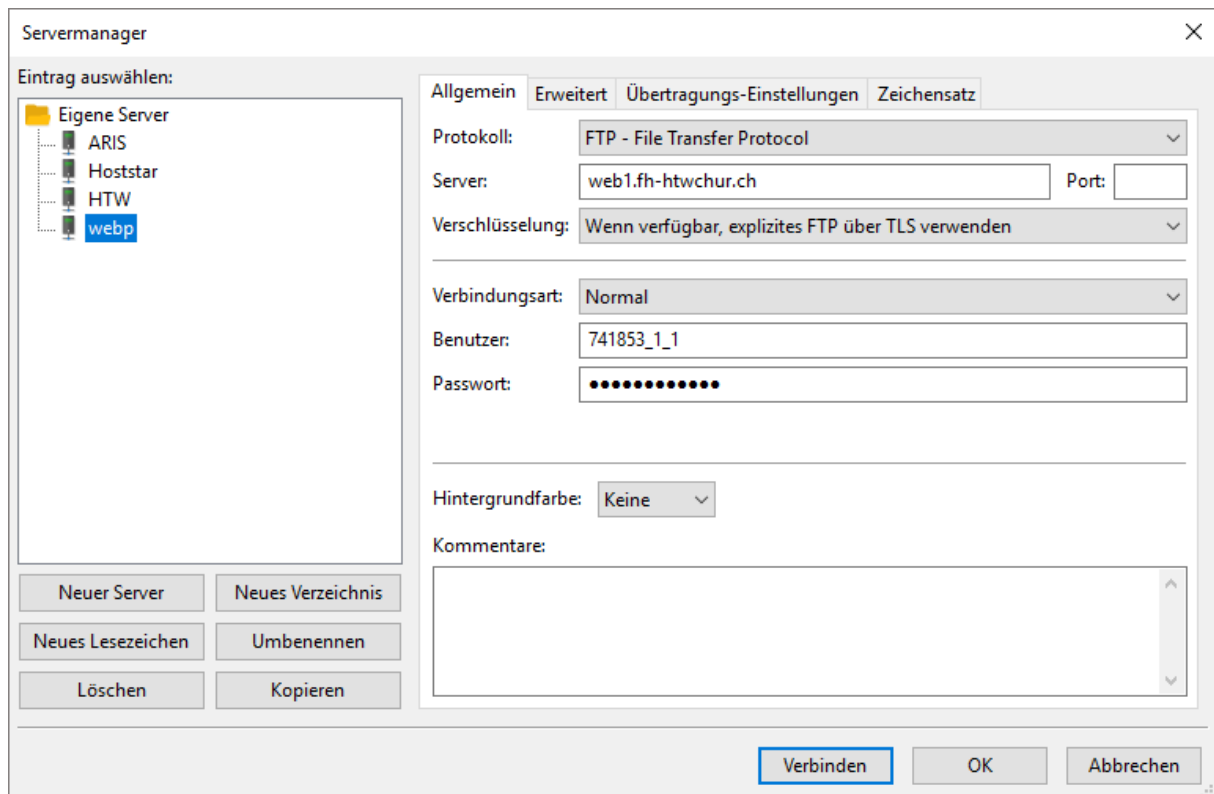
Tabellenstruktur Beziehungsansicht

Beschränkungen durch Fremdschlüssel

| Aktionen | Constrainteigenschaften | Spalte | Beschränkung für auswärtige Schlüssel (INNODB) | | |
|-------------------------|-------------------------|---------------------------------------|--|---------------------------------------|--------------------|
| | | | Datenbank | Tabelle | Spalte |
| Löschen | schicht_ibfk_1 | ON DELETE RESTRICT ON UPDATE RESTRICT | id_ma | 741853_1_1 | mitarbeiter id_ma |
| | | | + Spalte hinzufügen | | |
| Löschen | schicht_ibfk_2 | ON DELETE RESTRICT ON UPDATE RESTRICT | id_ap | 741853_1_1 | arbeitsplatz id_ap |
| | | | + Spalte hinzufügen | | |
| | | | Constraint-Name | ON DELETE RESTRICT ON UPDATE RESTRICT | 741853_1_1 |
| | | | + Spalte hinzufügen | | |
| + Constraint hinzufügen | | | | | |

2. FTP Client einrichten

FileZilla:



PW: Ck10EemjagF6

2.1 PHPMyAdmin

<http://web1.fh-htwchur.ch/phpmyadmin/index.php>

Benutzer: 741853_1_1

PW: IK3@ys8Cw4S2

2.2 Schichtplan

<http://741853-1.web1.fh-htwchur.ch/schichtplan.php>

3. Javascript/Query Funktionsbeschreibung

3.1 change_shift.js

Zu finden auf Zeilen 1 bis 50

Bei der Funktion `change_shift` wird eine vorhandene Schicht geändert. Ist eine Schicht ausgewählt, so wird diese Datenreihe mittels roter CSS `#f44336` Hintergrundfarbe gekennzeichnet (Zeilen 23 – 24) und die Daten werden oben rechts im Bereich *create shift plan* wiedergegeben.

Ist eine Schicht ausgewählt und abgeändert:

- werden durch das Betätigen des blauen “change shift” Button die neuen Daten gespeichert
- es erscheint eine grüne Meldung (Zeilen 34 – 37)
- die Daten werden in das `index.html` (Zeilen 88 und 100) eingespielt

Ist keine Schicht ausgewählt und wird der blaue Button trotzdem betätigt:

- erscheint eine rote Fehlermeldung (Zeilen 45 – 48) mit dem Text “no shift selected”
- die Daten werden nicht in das `index.html` (Zeilen 88 und 100) eingespielt

Enthalten die Felder *employees* sowie *date* keine Daten:

- erscheint eine rote Fehlermeldung (Zeilen 41 – 44), mit dem Text “no data selected”
- die Daten werden nicht in das `index.html` (Zeilen 88 und 100) eingespielt

3.2 create_employee.js

Zu finden auf Zeilen 1 bis 42

Bei der Funktion `create_employee` wird ein neuer Mitarbeiter hinzugefügt. Wird das Feld «Employee:» mit einem Namen befüllt und anschliessend auf den grünen +-Button geklickt, so wird diese Funktion aufgerufen.

Die Eingabe muss mindestens 2 Buchstaben aufweisen:

- Bei **weniger als 2 Buchstaben** wird eine entsprechende Fehlermeldung bei «success & error messages» (`<p id='msg_error'>`) ausgegeben und der Schichtplan wird nicht erstellt
- Die Fehlermeldung wird durch das CSS rot `#f44336` dargestellt (`#msg_error`)
- die Daten werden nicht in das `index.html` (Zeilen 56 und 59) eingespielt
- Bei **mehr als 2 Buchstaben** wird eine grüne `#4CAF50` Bestätigungsmeldung (CSS: `#msg_success`) angezeigt und die Daten werden abgefüllt
- die Daten werden in das `index.html` (Zeilen 56 und 59) eingespielt

Beispiel: Da Feld «Employee» kann nicht leer abgegeben werden.

3.3 create_shift.js

3.3.1 Von/-bis Zeit Schicht

Zu finden auf Zeilen 21 bis 44

Beim Erstellen einer Schicht muss der Inhalt von `tme_from_creat_shift` kleiner sein als der Inhalt von `tme_to_creat_shift`.

- Bei einer **falschen Eingabe** wird eine entsprechende Fehlermeldung im `<p id='msg_error'>` ausgegeben und die Schicht wird nicht erstellt
- Die Fehlermeldung wird durch das CSS rot `#f44336` dargestellt (`#msg_error`)
- die Daten werden nicht in das `index.html` (Zeilen 65 und 70) eingespielt

- War die **Eingabe erfolgreich**, wird eine grüne #4CAF50 Bestätigungsmeldung (CSS: #msg_success) angezeigt und die Daten werden abgefüllt
- die Daten werden in das index.html (Zeilen 65 und 70) eingespielt

Beispiel: Eine Schicht kann nicht von 12:00:00 bis 11:00:00 Uhr dauern

3.3.2 Zeitformat konvertieren

Auf Zeilen 10/11 und 13-15

Bei den beiden Zeiteingabefenster time_from und time_to sollen jeweils nur HH:MM angezeigt werden ohne Sekunden. Zusätzlich wurde auch der HTML Code angepasst.

Die beiden Zeitkonvertierungen wurden jedoch auskommentiert, da auch nach längerer Recherche keine Anpassung am Zeitformat erfolgreich durchgeführt werden konnte.

3.4 create_workplace.js

Zu finden auf Zeilen 1 – 41

Bei der Funktion create_workplace wird ein neuer Arbeitsplatz hinzugefügt. Wird das Feld «Workplace» mit einem Wort für den neuen Arbeitsplatz befüllt und anschliessend auf den grünen + Button geklickt, so wird diese Funktion aufgerufen.

Die Eingabe muss mindestens 2 Buchstaben aufweisen:

- Bei **weniger als 2 Buchstaben** wird eine entsprechende Fehlermeldung bei «success & error messages» (<p id=<'msg_error'>) ausgegeben und der Arbeitsplatz wird nicht erstellt
- Die Fehlermeldung wird durch das CSS rot #f44336 dargestellt (#msg_error)
- die Daten werden nicht in das index.html eingespielt
- Bei **mehr als 2 Buchstaben** wird eine grüne #4CAF50 Bestätigungsmeldung (CSS: #msg_success) angezeigt und die Daten werden abgefüllt
- die Daten werden in das index.html eingespielt

Beispiel: Wird in die Textbox bei «create workplace» das Wort «Toiletten» eingetragen und anschliessend auf den grünen + Button geklickt, so wird der Arbeitsplatz «Toiletten» hinzugefügt. Nach dem Klicken auf den grünen Button erscheint folgende «success message»:

success & error messages

successfully created new workplace -> Toiletten

3.5 current_date.js

Zu finden auf Zeilen 1 – 13

Bei der Funktion current_date wird im Bereich des «create shift plan» jeweils das heutige Datum angezeigt. Mit dem Laden der Seite wird das Tagesdatum eingetragen. Dabei wird das Datum wie folgt angezeigt: dd.mm.yyyy.

Beispiel: 29.05.2019

3.6 delete_employee.js

Zu finden auf Zeilen 1 – 40

Wenn der User einen Mitarbeiter (employee) löschen möchte, dann muss ein Arbeitsplatz im Feld *selected_employee* eingetragen sein (Zeile 13)

- Ist ein Mitarbeiter ausgewählt und wird dieser mittels rotem Minuszeichen gelöscht, so erscheint eine grüne Meldung (Zeilen 28 – 30)
- Ist kein Mitarbeiter ausgewählt und wird das rote Minuszeichen trotzdem betätigt, so erscheint eine rote Fehlermeldung (Zeilen 23 – 25)

3.7 delete_shift.js

Zu finden auf Zeilen 1 – 43

Wenn der User eine Schicht (shift) löschen möchte, dann muss eine Schicht im Bereich *table_shift* ausgewählt sein.

- Ist eine Schicht ausgewählt, so wird diese Datenreihe mittels roter Hintergrundfarbe gekennzeichnet. (Zeilen 23 – 24)
- Ist eine Schicht ausgewählt und wird diese mittels dem roten “delete shift” Button gelöscht, so erscheint eine grüne Meldung (Zeilen 27 – 29). Die Daten werden im *index.html* aktualisiert und eingespielt. (Zeilen 88 – 100)
- Ist keine Schicht ausgewählt und wird der rote Button trotzdem betätigt, so erscheint eine rote Fehlermeldung. (Zeilen 33 – 35) Die Datensätze im *index.html* (Zeilen 88 – 100) ändern sich nicht.

3.8 delete_workplace.js

Zu finden auf Zeilen 1 – 40

Wenn der User einen Arbeitsplatz (workplace) löschen möchte, dann muss ein Arbeitsplatz im Feld *selected_workplace* eingetragen sein (Zeile 13)

- Ist ein Arbeitsplatz ausgewählt und wird dieser mittels rotem Minuszeichen gelöscht, so erscheint eine grüne Meldung (Zeilen 28 – 30)
- Ist kein Arbeitsplatz ausgewählt und wird das rote Minuszeichen trotzdem betätigt, so erscheint eine rote Fehlermeldung (Zeilen 23 – 25)

3.9 empty_messages.js

Zu finden auf Zeilen 1 bis 4

Dieses Script ist zuständig dafür, dass die Felder mit der Fehlermeldung bzw. der Bestätigung geleert werden, sobald eine neue Abfrage getätigt wird.

3.10 get_employees.js

Zu finden auf Zeilen 1 bis 15

Bei der Erstellung einer Schicht muss man den gewünschten Mitarbeitenden wählen. Die Funktion *get_employee.js* ermöglicht eine Dropdown-Liste aller eingegebenen Mitarbeitenden aus der Datenbank (*get_employees.php*). Diese werden vorher mit der Funktion *create_employee.js* eingegeben.

3.11 get_shift.js

Zu finden auf Zeilen 1 bis 17

Die Funktion get_shift.js ist beim Ausführen der change_shift, delete_shift sowie create_shift Funktion für die Wiedergabe der einzelnen Arbeitsschichten im Bereich "table shift" zuständig. Die Funktion gibt die Werte: value.id_sp, value.name, value.beschreibung, value.datum, value.zeitvon & value.zeitbis wieder.

Dabei werden die Datensätze wie folgt angezeigt:

| Employee | Workplace | Date | From | To |
|----------|-----------|------------|----------|----------|
| Joanna | Abwasch | 2019-05-15 | 08:00:00 | 17:00:00 |

Ansicht des Bereichs "table shift"

Quelle: eigene Darstellung

Wie bei der oben erwähnten Funktion, muss beim Erstellen einer Schicht der Inhalt von `tme_from_creat_shift` kleiner sein als der Inhalt von `tme_to_creat_shift`.

- Bei einer **falschen Eingabe** wird eine entsprechende Fehlermeldung im `<p id='msg_error'>` ausgegeben und die Schicht wird nicht erstellt
- Die Fehlermeldung wird durch das CSS rot `#f44336` dargestellt (`#msg_error`)
- die Daten werden nicht in das index.html (Zeilen 65 und 70) eingespielt
- War die **Eingabe erfolgreich**, wird eine grüne `#4CAF50` Bestätigungsmeldung (CSS: `#msg_success`) angezeigt und die Daten werden abgefüllt
- die Daten werden in das index.html (Zeilen 65 und 70) eingespielt

3.12 get_workplaces.js

Zu finden auf Zeilen 1 bis 15

Bei der Erstellung einer Schicht muss man den gewünschten Arbeitsplatz wählen. Die Funktion get_workplace.js ermöglicht eine Dropdown-Liste aller eingegebenen Arbeitsplätze aus der Datenbank. Diese werden vorher mit der Funktion create_workplace.js eingegeben.

3.13 check_shift.js

Zu finden auf Zeilen 1 bis 54.

Diese Funktion ist aus den im Code beschriebenen Gründen leider nicht funktionstüchtig. Sie verfolgt jedoch folgendes Ziel:

Mit der Funktion check_shift.js soll verhindert werden, dass ein Mitarbeiter mehrere Schichten am selben Tag hat. Dies soll mittels folgender Überprüfungen erfolgen:

- `if (date == value.datum && selected_employee == value.name)`. Die erste Überprüfung prüft, ob das eingegebene Datum in der Datenbank bereits vorhanden ist **und** ob der eingegebene Name bereits vermerkt ist. Wenn ja, wird eine Fehlermeldung ausgespielt.
- `else if (selected_employee != value.name && date == value.date)`. Ist die erste Überprüfung negativ erfolgt eine zweite Prüfung. Davon ausgegangen, dass das Datum in der ersten Überprüfung falsch ist, soll in einer zweiten Prüfung geschaut werden, ob der Name nicht dem eingegebenen Namen entspricht, das Datum jedoch schon. Ist das der Fall soll eine Fehlermeldung ausgespielt werden. Wenn nein kann die neue Schicht in der Datenbank geschrieben werden.

Es wurde mit diversen Ansätzen versucht die Problematik zu lösen. Das oben beschriebene stellt lediglich die letzte versuchte Variante dar. Leider wurden mit dieser Variante bei jedem Test ca. 50 neue Schichten erstellt und wir konnten nicht ausfindig machen wo das Problem liegt.

4. PHP Funktionsbeschreibung

In der connection.php wird eine mysqli Verbindung mit den entsprechenden Parametern geöffnet. Diese Datei wird in allen php Dateien eingebunden.

```
<?php
$connection = mysqli_connect("localhost", "root", "password", "database");
?>
```

4.1 create_employee.php

Mit Hilfe von dieser PHP-Datei können neue Mitarbeiter in der Datenbank angelegt werden. Dazu wird über POST ein Parameter entgegengenommen. Im entsprechenden ajax Call muss der Parameter als „**name**“ deklariert sein.

```
//get POST parameters from ajax
$name = $_POST['name'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "INSERT INTO mitarbeiter (name) VALUES ('$name')";
```

Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```

4.2 create_workplace.php

Mit Hilfe von dieser PHP-Datei können neue Arbeitsplätze in der Datenbank angelegt werden. Dazu wird über POST ein Parameter entgegengenommen. Im entsprechenden ajax Call muss der Parameter als „**workplace**“ deklariert sein.

```
//get POST parameters from ajax
$workplace = $_POST['workplace'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "INSERT INTO arbeitsplatz (beschreibung)
VALUES ('$workplace')";
```

Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```

4.3 create_shift.php

Mit Hilfe von dieser PHP-Datei können neue Schichten in der Datenbank angelegt werden. Dazu werden über POST 5 Parameter entgegengenommen. Im entsprechenden ajax Call müssen die Parameter als „name“, „workplace“, „date“, „time_from“, „time_to“ deklariert sein.

```
//get POST parameters from ajax
$name = $_POST['name'];
$workplace = $_POST['workplace'];
$date = $_POST['date'];
$time_from = $_POST['time_from'];
$time_until = $_POST['time_until'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "INSERT INTO schicht (id_ma, id_ap, datum, zeitvon, zeitbis)
VALUES ((SELECT id_ma from mitarbeiter where name='$name'),
(SELECT id_ap from arbeitsplatz where beschreibung='$workplace'),
'$date',
'$time_from',
'$time_until')";
```

Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```

4.4 delete_employee.php

Mit Hilfe von dieser PHP-Datei können Mitarbeiter in der Datenbank gelöscht werden. Dazu wird über POST ein Parameter entgegengenommen. Im entsprechenden ajax Call muss der Parameter als „name“ deklariert sein.

```
//get POST parameters from ajax
$name = $_POST['name'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "DELETE FROM mitarbeiter WHERE name = '$name'";
```

Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```

4.5 delete_workplace.php

Mit Hilfe von dieser PHP-Datei können Arbeitsplätze in der Datenbank gelöscht werden. Dazu wird über POST ein Parameter entgegengenommen. Im entsprechenden ajax Call muss der Parameter als „workplace“ deklariert sein.

```
//get POST parameters from ajax
$workplace = $_POST['workplace'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "DELETE FROM mitarbeiter WHERE name = '$name'";
```

Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```

4.6 delete_shift.php

Mit Hilfe von dieser PHP-Datei können Schichten in der Datenbank gelöscht werden. Dazu wird über POST ein Parameter entgegengenommen. Im entsprechenden ajax Call muss der Parameter als „workplace“ deklariert sein.

```
//get POST parameters from ajax
$id_sp = $_POST['id_sp'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "DELETE FROM schicht WHERE schicht.id_sp = '$id_sp'";
```

Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```

4.7 get_employees.php

Mit Hilfe von dieser PHP-Datei können Mitarbeiter in der Datenbank abgefragt werden. Für den Aufruf werden keine Parameter benötigt.

Das SQL-Statement gestaltet sich wie folgt:

```
//execute sql statement
$result = $connection->query("SELECT name FROM mitarbeiter");
```

Als Rückgabewert werden die abgefragten Daten im JSON Format zurückgeliefert.

```
//loop through output and write to array
while ( $row = $result->fetch_assoc()) {
    $arr_employees[]=$row;
}

//send back array in json format
echo json_encode($arr_employees);
```

4.8 get_workplaces.php

Mit Hilfe von dieser PHP-Datei können Arbeitsplätze in der Datenbank abgefragt werden. Für den Aufruf werden keine Parameter benötigt.

Das SQL-Statement gestaltet sich wie folgt:

```
//execute sql statement
$result = $connection->query("SELECT beschreibung FROM arbeitsplatz");
```

Als Rückgabewert werden die abgefragten Daten im JSON Format zurückgeliefert.

```
//loop through output and write to array
while ( $row = $result->fetch_assoc()) {
    $arr_workplaces[]=$row;
}

echo json_encode($arr_workplaces);
```

4.9 get_shift.php

Mit Hilfe von dieser PHP-Datei können Schichten in der Datenbank abgefragt werden. Für den Aufruf werden keine Parameter benötigt.

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "select schicht.id_sp, mitarbeiter.name, arbeitsplatz.beschreibung, schicht.datum, schicht.zeitvon, schicht.zeitbis
from schicht
inner join mitarbeiter on schicht.id_ma = mitarbeiter.id_ma
inner join arbeitsplatz on schicht.id_ap = arbeitsplatz.id_ap;";
```

Als Rückgabewert werden die abgefragten Daten im JSON Format zurückgeliefert.

```
//loop through output and write to array
while ( $row = $result->fetch_assoc()) {
    $arr_shift[]=$row;
}

//send back array in json format
echo json_encode($arr_shift);
```

4.10 update_shift.php

Mit Hilfe von dieser PHP-Datei können neue Schichten in der Datenbank angelegt werden. Dazu werden über POST 5 Parameter entgegengenommen. Im entsprechenden ajax Call müssen die Parameter als „name“, „workplace“, „date“, „time_from“, „time_to“ deklariert sein.

```
//get POST parameters from ajax
$date = $_POST['date'];
$time_from = $_POST['time_from'];
$time_until = $_POST['time_until'];
$id_sp = $_POST['id_sp'];
$name = $_POST['name'];
$workplace = $_POST['workplace'];
```

Das SQL-Statement gestaltet sich wie folgt:

```
//prepare sql statement
$sql = "UPDATE schicht
INNER JOIN mitarbeiter ON mitarbeiter.name = '$name'
INNER JOIN arbeitsplatz ON arbeitsplatz.beschreibung = '$workplace'
SET schicht.id_ma = mitarbeiter.id_ma,
    schicht.id_ap = arbeitsplatz.id_ap,
    schicht.datum = '$date',
    schicht.zeitvon = '$time_from',
    schicht.zeitbis = '$time_until'
WHERE schicht.id_sp = '$id_sp'";
```


Als Rückgabewert werden entweder die success oder error message im JSON Format zurückgeliefert. Wenn das SQL-Statement erfolgreich abgesetzt werden konnte, wird **success=true** gesetzt. Im Fehlerfall lautet die entsprechende Passage **success=false**.

```
//write success or error messages to array for send back to ajax
if (!empty($errors)) {
    $form_data['success'] = false;
    $form_data['errors'] = $errors;
} else {
    $form_data['success'] = true;
    $form_data['posted'] = $ok;
}

//send back array in json format
echo json_encode($form_data);
```