# Overview tables

Vorname Der Vorname dient (zusammen mit dem Nachnamen) zur Identifikation des Referenzgebers.

Nachname Der Nachname dient (zusammen mit dem Vornamen) zur Identifikation des Referenzgebers.

Unternehmen Der Name des Unternehmens dient auch zur Identifikation des Referenzgebers.

URL Die URL ist wichtig, damit sich der Webseitenbesucher weitergehende Informationen auf der Webseite des Referenzgebers holen kann.

Logo Das Logo kann bei Wiederekennung zur Identfizierung beitragen.

Video Videos vermitteln einen besonders authentischen Eindruck vom Referenzgeber und können möglicherweise am besten überzeugen.

Bilder Fotos sind von Vorteil, wenn sich ein Webseitenbsucher nur schnell einen ersten Eindruck verschaffen will, und keine Zeit für ein Video hat.

Ort Die Angabe des Ortes ist wichtig zur Identifizierung mit dem Referenzgeber, vielleicht zieht der Website-Besucher auch in Betracht, diesen persönlich aufzusuchen. Darüberhinaus dient auch eine weite Verbreitung der Nutzer der belbo-Software als Eigenwerbung.

Branche Die Software wird von einer Vielzahl an Branchen genutzt, darunter Friseure, Kosmetiksalons, Fitnessstudios und Piercing- und Tatoostudios. Der Webseitenbesucher möchte vermutlich eher oder zuerst die Referenzen von Referenzgebern aus der gleichen Branche anschauen und soll die Möglichkeit haben, danach zu filtern.

Kategorie Die Filtermöglichkeit nach der Art des Mediums (Video, Bild mit Text oder nur Text) gibt Webseitenbesuchern die Möglichkeit, entsprechend ihrer Vorlieben für bestimmte Medien die Referenzen auszuwählen.

# Datenbanktabelle versus WordPress Post-Einträge

Im Folgenden diskutieren wir unsere Designentscheidung, die Datenbankschicht innerhalb von WordPress anzusiedeln. Hierbei unterscheiden wir, ob wir je Testimonial einen Post Eintrag oder eine Zeile in einer eigenen Datenbank erstellen.

# Pro Datenbanktabelle

* Das Arbeiten auf einer kleinen Datenbank ermöglicht eine bessere Performance.
* Das Arbeiten auf einer eigens erstellten Datenbank bietet eine höhere Sicherheit. Beim Einsatz von nicht selbst entwickelten Plugins besteht immer die Gefahr von potentiellen Sicherheislücken.
* Eine Datenbanktabelle ist normalerweise übersichtlicher und einfacher zu analysieren.
* Die Daten sind einfacher sicherbar und exportierbar.

# Pro Post-Eintrag in WordPress Tabelle

* WordPress hat eine Vielzahl von Helper Funktionen, mit denen die Entwicklung von Plugins erleichtert wird. So kann zum Beispiel die Helper Class WP\_Query verwendet werden, um Datenbankabfragen zu vereinfachen.
* Die WordPress Datenbank ist insofern übersichtlich, als dass die zentrale Tabelle die wp\_posts Tabelle ist. Alle Plugins greifen darauf zu.
* WordPress beinhaltet automatisch die Versionierung aller Einträge und Änderungen. Bei einer Datenbanktabelle müsste jedesmal manuell ein neuer Eintrag erstellt werden.
* Jeder Eintrag und jede Änderung wird automatisch mit dem Autor verknüpft.
* Der Status jedes Posts wird automatisch dokumentiert, um parallele Bearbeitungen zu verhindern.
* Updates können einfacher und risikofreier installiert werden.
* Es sind viele kostenlose Themes (Design) für den Backend- und Frontend-Bereich enthalten, womit die jeweiligen graphischen Oberflächen automatisch gestaltet werden.
* Es können mit wenig Aufwand Tags angelegt werden, um z.B Beiträge zu filtern.

* **Wie wird auf eine Filterung reagiert?**

* Mit der jQuery change() Methode wird eine Funktion ausgeführt, wenn ein change-Event auftritt.

Ein change-Event tritt auf, wenn der Wert eines Elementes verändert wird (also bei Texteingabe in <input> oder <textarea> Elementen, oder bei der Auswahl eines <option> Elementes innerhalb eines <select> Elementes).

* $(selector).change(function)

* Beispiel in Plugin:

**$('select').change(function(){**

**$('.testimonial-content').hide();**

**$('.testimonial-content.' + $(this).val()).show();**

**});**

Bei der Auswahl eines Select-Elementes wird die jQuery Methode hide() aufgerufen, mit der das ausgewählte Element versteckt wird, also hier die Klasse **.testimonial-content**

In dem <div> Container mit der Klasse .testimonial-content werden clientseitig alle Referenzen abgebildet, d.h. es werden zunächst alle Referenzen versteckt.

Mit der jQuery Methode show() werden alle versteckten, selektierten Elemente angezeigt, d.h. alle Elemente in den Klassen **.testimonial-content** und **$(this).val()**

Mit der jQuery Methode val() wird das value-Attribut des ausgewählten Elementes zurückgegeben, also z.B. Berlin bei der Auswahl in der Select-Box.

* Die JavaScript-Funktion wird nur clientseitig ausgeführt, wenn der Benutzer ein Filter anwählt.

* **Wie sind die Tags mit dem HTML verknüpft?**

* Mit der Methode get\_the\_terms() werden die Terms (also in unserem Fall die Tags) abgerufen, die mit dem entsprechenden Post (Referenz) verknüpft sind.

* Beispiel in Plugin:

$locationtags = get\_the\_terms(get\_the\_ID(), 'location');

$locationAsCSSClass = "";

foreach($locationtags as $term) {

$locationAsCSSClass .= $term->name." ";

}

Mit dieser foreach-Schleife können alle Tags abgerufen werden (falls mehrere gesetzt sind).

* Bei dem Aufruf der Referenzen-Seite werden die jeweiligen Tag-Klassen mit der php-Methode echo in den Quelltext geschrieben (korrekte Formulierung?), hier also z.B. mit <?php echo $locationAsCSSClass; ?>

.. in das testimonial html als css klassen angefuegt, so dass diese via javscript angesprochen werden koennen

* **Warum kann z.B. $('.München').hide() in Konsole aufgerufen werden?**
  + Mit dem Aufruf der Referenzen-Seite werden bereits alle mit den Posts verknüpften Daten an den Client übermittelt, und deshalb kann lokal auf dem Client-Rechner JavaScript ausgeführt werden.

Alle daten liegen bereits vor, nochmal genauer zusammenfassen