

Anleitung

Erste Schritte

Nach deinem ersten Login gehe bitte unter Settings > User und klicke den Button **+ Add a User**. Nun kannst du dich als neuen Administrator anlegen. Danach lösche den bei Auslieferung bestehenden User „webmaster“. Unter dem Punkt Settings > Site solltest du nun alle Angaben gegen deine persönlichen Daten ersetzen. Eine ausführlichere Anleitung über das Erstellen eines neuen Nutzer findest du auch unter dem Kapitel **Einen neuen User anlegen**.

Wichtig: Die beim Formularfeld **E-Mail** eingetragene Adresse wird auch für alle Anfragen über das Kontaktformular verwendet.

Sollten bei Auslieferung der Software mehrere Layout verfügbar sein, so kannst du unter dem Punkt Settings > Theme deinen eignen Style wählen.

Die wichtigsten Schritte um deine neue Website zu erstellen sind danach getan. Nun kannst du dich deinen Inhalten widmen.

Eine bestehende Seite bearbeiten

Gehe auf Start > Pages und klicke bei der zu bearbeitenden Seite den Button **Edit**. Nun öffnet sich eine neue Seite mit dem Titel „Edit Page“, wie in Abbildung 1 zusehen. Fülle auf dieser nun folgende Pflichtfelder aus:

- Page Titel
- Slug/URL
- Headline
- Page Body

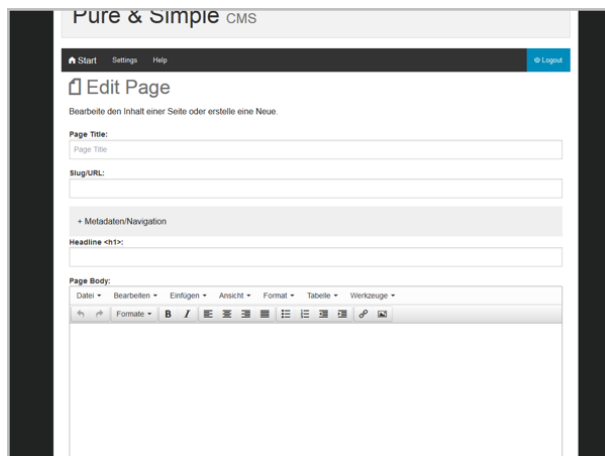


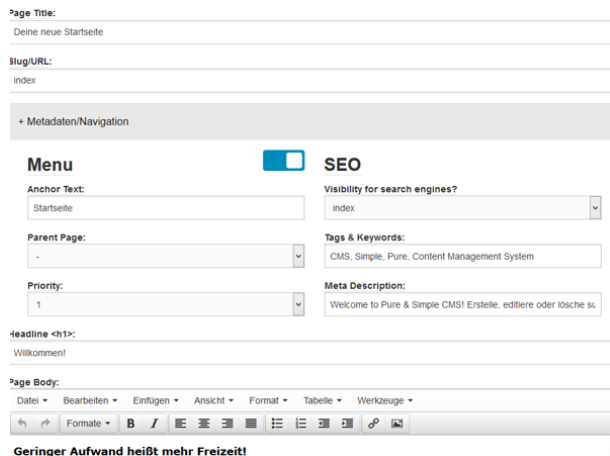
Abbildung 1: Eine Seite erstellen oder editieren auf „Edit Page“

Danach musst du deine Angaben nur noch speichern durch einen Klick auf den Button „Save Updates“. Überprüfe alles mit einem Blick auf die geänderte Seite. Löschen kann man die Seite durch Klick auf den Button **Delete**.

Wichtig: Es gibt fünf Seiten die standardmäßig zum System gehören index, contact, imprint, search und error. Bei diesen kann die Slug/URL nicht geändert oder die Seite gelöscht werden.

Alle nachfolgenden Angaben zur Navigation oder SEO beziehen sich ebenfalls auf Einstellungen von „Edit Page“.

Du möchtest die zu bearbeitende Seite auch in der Navigation verfügbar bzw. einen bestehenden Menüeintrag ändern oder löschen? Hierfür klicke auf den Button **+Metadaten/Navigation**. Es öffnen sich weitere Formularfelder, wie in Abbildung 2 zu sehen.



Page Title:
Deine neue Startseite

Slug/URL:
index

+ Metadaten/Navigation

Menu ☒ **SEO** ☐

Anchor Text:
Startseite

Parent Page:
-

Priority:
1

Visibility for search engines?:
index

Tags & Keywords:
CMS, Simple, Pure, Content Management System

Meta Description:
Welcome to Pure & Simple CMS! Erstelle, editiere oder lösche su...

Headline <h1>:
Willkommen!

Page Body:
Datei Bearbeiten Einfügen Ansicht Format Tabelle Werkzeuge
Format B I H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20 H21 H22 H23 H24 H25 H26 H27 H28 H29 H30 H31 H32 H33 H34 H35 H36 H37 H38 H39 H40 H41 H42 H43 H44 H45 H46 H47 H48 H49 H50 H51 H52 H53 H54 H55 H56 H57 H58 H59 H60 H61 H62 H63 H64 H65 H66 H67 H68 H69 H70 H71 H72 H73 H74 H75 H76 H77 H78 H79 H80 H81 H82 H83 H84 H85 H86 H87 H88 H89 H90 H91 H92 H93 H94 H95 H96 H97 H98 H99 H100 H101 H102 H103 H104 H105 H106 H107 H108 H109 H110 H111 H112 H113 H114 H115 H116 H117 H118 H119 H120 H121 H122 H123 H124 H125 H126 H127 H128 H129 H130 H131 H132 H133 H134 H135 H136 H137 H138 H139 H140 H141 H142 H143 H144 H145 H146 H147 H148 H149 H150 H151 H152 H153 H154 H155 H156 H157 H158 H159 H160 H161 H162 H163 H164 H165 H166 H167 H168 H169 H170 H171 H172 H173 H174 H175 H176 H177 H178 H179 H180 H181 H182 H183 H184 H185 H186 H187 H188 H189 H190 H191 H192 H193 H194 H195 H196 H197 H198 H199 H200 H201 H202 H203 H204 H205 H206 H207 H208 H209 H210 H211 H212 H213 H214 H215 H216 H217 H218 H219 H220 H221 H222 H223 H224 H225 H226 H227 H228 H229 H230 H231 H232 H233 H234 H235 H236 H237 H238 H239 H240 H241 H242 H243 H244 H245 H246 H247 H248 H249 H250 H251 H252 H253 H254 H255 H256 H257 H258 H259 H260 H261 H262 H263 H264 H265 H266 H267 H268 H269 H270 H271 H272 H273 H274 H275 H276 H277 H278 H279 H280 H281 H282 H283 H284 H285 H286 H287 H288 H289 H290 H291 H292 H293 H294 H295 H296 H297 H298 H299 H300 H301 H302 H303 H304 H305 H306 H307 H308 H309 H310 H311 H312 H313 H314 H315 H316 H317 H318 H319 H320 H321 H322 H323 H324 H325 H326 H327 H328 H329 H330 H331 H332 H333 H334 H335 H336 H337 H338 H339 H340 H341 H342 H343 H344 H345 H346 H347 H348 H349 H350 H351 H352 H353 H354 H355 H356 H357 H358 H359 H360 H361 H362 H363 H364 H365 H366 H367 H368 H369 H370 H371 H372 H373 H374 H375 H376 H377 H378 H379 H380 H381 H382 H383 H384 H385 H386 H387 H388 H389 H390 H391 H392 H393 H394 H395 H396 H397 H398 H399 H400 H401 H402 H403 H404 H405 H406 H407 H408 H409 H410 H411 H412 H413 H414 H415 H416 H417 H418 H419 H420 H421 H422 H423 H424 H425 H426 H427 H428 H429 H430 H431 H432 H433 H434 H435 H436 H437 H438 H439 H440 H441 H442 H443 H444 H445 H446 H447 H448 H449 H450 H451 H452 H453 H454 H455 H456 H457 H458 H459 H460 H461 H462 H463 H464 H465 H466 H467 H468 H469 H470 H471 H472 H473 H474 H475 H476 H477 H478 H479 H480 H481 H482 H483 H484 H485 H486 H487 H488 H489 H490 H491 H492 H493 H494 H495 H496 H497 H498 H499 H500 H501 H502 H503 H504 H505 H506 H507 H508 H509 H510 H511 H512 H513 H514 H515 H516 H517 H518 H519 H520 H521 H522 H523 H524 H525 H526 H527 H528 H529 H530 H531 H532 H533 H534 H535 H536 H537 H538 H539 H540 H541 H542 H543 H544 H545 H546 H547 H548 H549 H550 H551 H552 H553 H554 H555 H556 H557 H558 H559 H560 H561 H562 H563 H564 H565 H566 H567 H568 H569 H570 H571 H572 H573 H574 H575 H576 H577 H578 H579 H580 H581 H582 H583 H584 H585 H586 H587 H588 H589 H590 H591 H592 H593 H594 H595 H596 H597 H598 H599 H600 H601 H602 H603 H604 H605 H606 H607 H608 H609 H610 H611 H612 H613 H614 H615 H616 H617 H618 H619 H620 H621 H622 H623 H624 H625 H626 H627 H628 H629 H630 H631 H632 H633 H634 H635 H636 H637 H638 H639 H640 H641 H642 H643 H644 H645 H646 H647 H648 H649 H650 H651 H652 H653 H654 H655 H656 H657 H658 H659 H660 H661 H662 H663 H664 H665 H666 H667 H668 H669 H670 H671 H672 H673 H674 H675 H676 H677 H678 H679 H680 H681 H682 H683 H684 H685 H686 H687 H688 H689 H690 H691 H692 H693 H694 H695 H696 H697 H698 H699 H700 H701 H702 H703 H704 H705 H706 H707 H708 H709 H710 H711 H712 H713 H714 H715 H716 H717 H718 H719 H720 H721 H722 H723 H724 H725 H726 H727 H728 H729 H730 H731 H732 H733 H734 H735 H736 H737 H738 H739 H740 H741 H742 H743 H744 H745 H746 H747 H748 H749 H750 H751 H752 H753 H754 H755 H756 H757 H758 H759 H760 H761 H762 H763 H764 H765 H766 H767 H768 H769 H770 H771 H772 H773 H774 H775 H776 H777 H778 H779 H780 H781 H782 H783 H784 H785 H786 H787 H788 H789 H790 H791 H792 H793 H794 H795 H796 H797 H798 H799 H800 H801 H802 H803 H804 H805 H806 H807 H808 H809 H810 H811 H812 H813 H814 H815 H816 H817 H818 H819 H820 H821 H822 H823 H824 H825 H826 H827 H828 H829 H830 H831 H832 H833 H834 H835 H836 H837 H838 H839 H840 H841 H842 H843 H844 H845 H846 H847 H848 H849 H850 H851 H852 H853 H854 H855 H856 H857 H858 H859 H860 H861 H862 H863 H864 H865 H866 H867 H868 H869 H870 H871 H872 H873 H874 H875 H876 H877 H878 H879 H880 H881 H882 H883 H884 H885 H886 H887 H888 H889 H890 H891 H892 H893 H894 H895 H896 H897 H898 H899 H900 H901 H902 H903 H904 H905 H906 H907 H908 H909 H910 H911 H912 H913 H914 H915 H916 H917 H918 H919 H920 H921 H922 H923 H924 H925 H926 H927 H928 H929 H930 H931 H932 H933 H934 H935 H936 H937 H938 H939 H940 H941 H942 H943 H944 H945 H946 H947 H948 H949 H950 H951 H952 H953 H954 H955 H956 H957 H958 H959 H960 H961 H962 H963 H964 H965 H966 H967 H968 H969 H970 H971 H972 H973 H974 H975 H976 H977 H978 H979 H980 H981 H982 H983 H984 H985 H986 H987 H988 H989 H990 H991 H992 H993 H994 H995 H996 H997 H998 H999

Geringer Aufwand heißt mehr Freizeit!

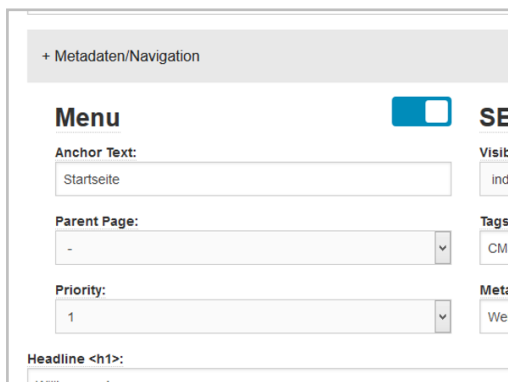
Abbildung 2: Formularfelder für Angaben zur Navigation und SEO

Eine Seite der Navigation hinzufügen, bearbeiten oder löschen

Wenn noch kein Eintrag in der Navigation angelegt ist, so klicke den grau hinterlegten Schalter neben der Überschrift Menu wie in Abbildung 3 zu sehen ist. Im Feld „Anchor Text“ ist als Pflichtangabe der sichtbare Text des Link anzugeben. Mache hier eine sehr kurze, aber prägnante Angabe. Wenn diese Seite als Untermenüpunkt erscheinen soll, dann wähle im Feld „Parent Page“ zu welcher Übergeordneten Site sie gehört. Um die Reihenfolge in deiner Navigation zu beeinflussen wählst du im Feld „Priority“ eine Ziffer. Hier gilt eine niedrigere Zahl für höhere Plätze und umgekehrt.

Zum löschen des Menüeintrag musst du nur den blau hinterlegten Schalter neben der Überschrift Menu klicken bis dieser grau hinterlegt ist. Nach getaner Änderung gilt auch hier wieder alles zu speichern.

Abbildung 3: Angaben zur Erstellung eines Menüpunkt



+ Metadaten/Navigation

Menu ☒ **SEO** ☐

Anchor Text:
Startseite

Parent Page:
-

Priority:
1

Visibility for search engines?:
index

Tags & Keywords:
CMS

Meta Description:
Welcome to Pure & Simple CMS! Erstelle, editiere oder lösche su...

Headline <h1>:
Willkommen!

Angaben für Suchmaschinen machen

Weiterhin ist es möglich Angaben zur Suchmaschinenoptimierung zu machen. Im ersten Feld unter dem Punkt SEO wirst du gefragt ob diese Seite für Suchmaschinen sichtbar sein soll. Wenn im Auswahlfeld „index“ gesetzt ist wäre dies gegeben. Gleichzeitig wird diese Seite auch im XML-Sitemap gelistet. In den beiden nachfolgenden Feldern mache Angaben zum Inhalt deiner Seite. Die Meta-Description wird übrigens in Suchmaschinen als erster Indikator für deine Besucher angezeigt und sollte nicht länger als 165 Zeichen sein. Auch nach Änderung dieser Angaben nicht vergessen alles zu speichern.

Eine neue Seite erstellen

Gehe auf Start > Pages und klicke den Button **+Add a Page**. Nun öffnet sich eine neue Seite mit dem Titel „Edit Page“. Fülle auf dieser nun folgende Pflichtfelder aus:

- Page Titel
- Slug/URL
- Headline
- Page Body

Ebenso können Angaben zum Erstellen eines Navigationseintrag oder der Suchmaschinenoptimierung gemacht werden. Details erfährst du auch im vorherigen Kapitel ***Eine bestehende Seite bearbeiten.***

Wichtig: Die Zuordnung der neuen Seite im Punkt „Menu“, zu einer Übergeordneten Seite (Parent Page), lässt sich erst nach dem Speichern vornehmen.

Bilder zu deiner Bibliothek hinzufügen

Gehe auf Start > Files und wähle durch Klick auf den Button **Datei suchen** ein Bild auf deinem lokalen Rechner aus, wie in Abbildung 4 zu sehen ist. Erlaubt sind die Dateitypen jpg, png und gif. Nun beginne den Upload durch Klicken des Button **hochladen**.

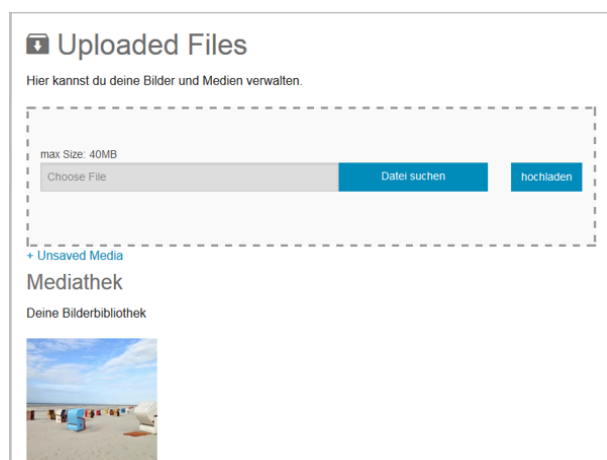


Abbildung 4: Bilder zum Hochladen auswählen und Upload starten

Bei Erfolg bekommst du die hochgeladene Datei angezeigt. Unter dem Vorschaubild befindet sich nun ein Formularfeld zur Angabe des Dateinamen, unter dem diese gespeichert werden soll. Zu sehen ist dies in Abbildung 5. Wähle einen möglichst treffenden, aber auch kurzen Namen. Dieser wird zusätzlich umgewandelt und als ALT-Tag im Quelltext verwendet.

Abbildung 5: Dateinamen für ein hochgeladenes Bild vergeben und speichern

Du hast dein Bild nach dem erfolgreichen Upload nicht gespeichert? Kein Problem. Es wird in einem temporären Ordner für dich aufbewahrt. Eine Übersicht der Inhalt in diesem Ordner findest du durch Klick auf den Button **+Unsaved Media** wie in Abbildung 6 zu sehen ist.



Abbildung 6: Übersicht der Dateien im temporären Ordner.

Im übrigen wird zu jedem hochgeladenen Bild automatisch ein kleines Vorschaubild (Thumbnail) erstellt und gespeichert.

Wichtig: Um alle Bilder problemlos im Content deiner Seite verwenden zu können, musst du nur den Button für Fotos im Editor deiner Seite klicken. Diesen siehst du in Abbildung 7 mit einem roten Kreis hervorgehoben. Unter dem Punkt Image List findet sich eine Auflistung aller, in der Mediathek befindlichen Bilder.

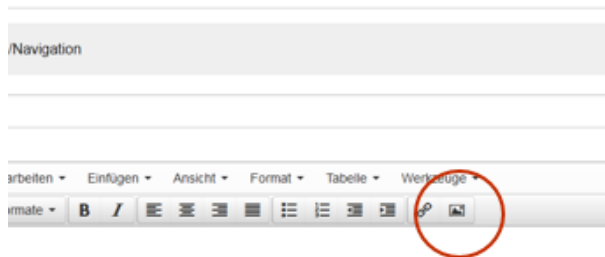


Abbildung 7: Auswahl eines Bild zum Einfügen in den Inhalt einer Seite

Arbeiten mit Variablen im Page Body deiner Seite

Diese Software ist dafür ausgelegt mit vorab definierten Variablen zu arbeiten. Das hat den Vorteil, wenn sich beispielsweise die Öffnungszeiten bzw. deine persönlichen Daten ändern. Um nicht den gesamten Text absuchen zu müssen, und die Zeiten überall von Hand zu ändern, setzt die Variable {%OPENING%}. Nun übernimmt dieser Platzhalter den jeweiligen Wert. Gleiches gilt natürlich auch für alle anderen Angaben wie z.B. Namen oder E-Mail. Eben diese Werte werden gesetzt unter Settings > Site, wie auch in Abbildung 8 zu sehen ist. Eine tolle Möglichkeit diese Variablen zu nutzen ist im Impressum deiner Seite. Ein Beispiel nach aktuellem Stand des §5 Telemediendienstgesetz ist bereits vorinstalliert.

Website Settings	
Hier kannst du deine Seiteneinstellungen verwalten.	
Slogan Pure & Simple CMS	
Die hier gesetzten Variablen kannst Du im Content deines Projekt nutzen.	
Firstname: {%FIRSTNAME%} <input type="text" value="Vorname"/>	Phone: {%PHONE%} <input type="text" value="0123 456789"/>
Lastname: {%LASTNAME%} <input type="text" value="Nachname"/>	E-Mail: {%EMAIL%} <input type="text" value="noreply@inet.marketing"/>
Street: {%STREET%} <input type="text" value="Strasse 12"/>	Company: {%COMPANY%} <input type="text" value="Musterfirma"/>
Postalzip: {%POSTALZIP%} <input type="text" value="01234"/>	Opening: {%OPENING%} <input type="text" value="Mo - Fr 10:00 bis 18:00 Uhr"/>
City: {%CITY%} <input type="text" value="Ort"/>	Variable: {%VARIABLE%} <input type="text" value="Soeben bin ich stolz mit meiner neuen Internetseite"/>

Abbildung 8: Variablen für dein gesamtes Projekt verfügbar machen

Wichtig: Du brauchst Administratorrechte um Variablen speichern oder editieren zu können.

Bezeichnung	Platzhalter	Beschreibung
Kontaktformular	{%CONTACTFORM%}	Erzeugt einen HTML-Quellcode für dein Kontaktformular
Statusmeldung Versenden	{%CONTACTFORM_MESSAGE%}	Gib deinen Usern eine Rückmeldung über Erfolg beim Versenden des Kontaktformulars. Platziere diese möglichst nahe dem Kontaktformular. Ausgabe erfolgt in einem Paragraph. Je nach Status hat dieser eine CSS-Klasse <code>error</code> oder <code>success</code> .
Suchformular	{%SEARCHFORM%}	Erzeugt einen HTML-Quellcode für dein Suchformular
Vorname	{%FIRSTNAME%}	Ersetzt den Platzhalter durch deinen gespeicherten Vornamen. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
Nachname	{%LASTNAME%}	Ersetzt den Platzhalter durch deinen gespeicherten Nachnamen. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
Straße	{%STREET%}	Ersetzt den Platzhalter durch deine gespeicherte Straße. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
Postleitzahl	{%POSTALZIP%}	Ersetzt den Platzhalter durch deine gespeicherte PLZ. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
Ort	{%CITY%}	Ersetzt den Platzhalter durch deinen gespeicherten Wohnort. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
Telefonnummer	{%PHONE%}	Ersetzt den Platzhalter durch deine gespeicherte Telefonnummer. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
E-Mail	{%EMAIL%}	Ersetzt den Platzhalter durch deine gespeicherte E-Mail Adresse. An jeder Stelle im <i>Page Body</i> beliebig verwendbar. Wichtig: Die hier eingetragene Adresse wird auch für alle Anfragen über das Kontaktformular verwendet.
Firmenname (optional)	{%COMPANY%}	Wenn du diese Seite für einen gewerblichen Internetauftritt nutzt, dann wird der Platzhalter durch deinen Firmennamen ersetzt. An jeder Stelle im <i>Page Body</i> beliebig verwendbar.
Öffnungszeiten (optional)	{%OPENING%}	Du hast Öffnungszeiten? Trage sie in dieses Feld. Dann kannst du den Platzhalter beliebig in deinem Inhalt verwenden.
Variable (optional)	{%VARIABLE%}	Diese Variable steht zu deiner freien Verfügung. Belege sie mit einem Inhalt, dann kannst du den Platzhalter beliebig in deinem Projekt verwenden.

Einen neuen User anlegen

Navigiere zur Seite **Settings > User** und klicke den Button **+Add a User**. Danach öffnet sich eine neue Seite mit der Überschrift „Edit User“, wie in Abbildung 9 zu sehen ist. Alle Felder ausfüll-

len ist Pflicht. Beachte bitte die Vergabe von Administratorrechten. Wenn du den neuen Nutzer auf den Status User setzt, so kann dieser keine neuen Nutzer anlegen oder bestehende editieren, bzw. löschen. Nicht vergessen nach der Eingabe alles zu speichern. Bitte merke dir unbedingt dein Passwort. Derzeit existiert noch keine Funktion um sich ein vergessenes zusenden zu lassen!

Pure & Simple CMS

Start Settings Help Hi Logout

Edit User

Bearbeite deine Benutzerdaten oder erstelle neue Nutzer.

Username:

User E-Mail:

User Status:

Password:
Confirm Password:

speichern

[Delete](#)

Registration:

Abbildung 9: Einen neuen User anlegen oder bestehenden editieren

Einen bestehenden User editieren oder das Passwort ändern

Du musst als Administrator eingeloggt sein um diese Änderungen durchführen zu können. Navigiere auf Settings > User und wähle in der Übersicht den User aus. Klicke den Button **Edit** wie in Abbildung 10 beispielhaft gezeigt wird. Nun gelangst du zur Eingabemaske. Nimm deine Änderungen vor und speichere alles ab. Das Passwort muss bei Änderung einmal wiederholt werden im Feld „Confirm Password“.

Wichtig: Wenn das Passwort bestehen bleiben soll, lasse dieses Feld einfach leer. Einen Benutzer löscht man durch Klick auf den Button **Delete**.

User Profile

Hier kannst du deine Benutzer verwalten.

[+ Add a User](#)

ID	Username	Status	
1	webmaster	admin	Edit

Abbildung 10: Übersicht angelegter Benutzer

Das Layout der Seite ändern

Die Software wird mit einem Standard-Thema ausgeliefert. Dieses nennt sich in jedem Falle Default. Du kannst jedoch ein eigenes erstellen und dieses als Vorlagen nutzen. Ebenso ist es möglich ein neues per FTP-Upload hinzu zu fügen. Hierfür benötigst du ein Programm wie z.B. Filezilla. Lade dein neues Template in den Ordner public > templates. Nach erfolgreichem Upload gehe im Administratorbereich auf Settings > Theme. Im Feld Template List sollte nun dein neues Template zur Auswahl stehen wie in Abbildung 11 zu sehen. Danach klicke den Button setzen und schon ist es aktiv.

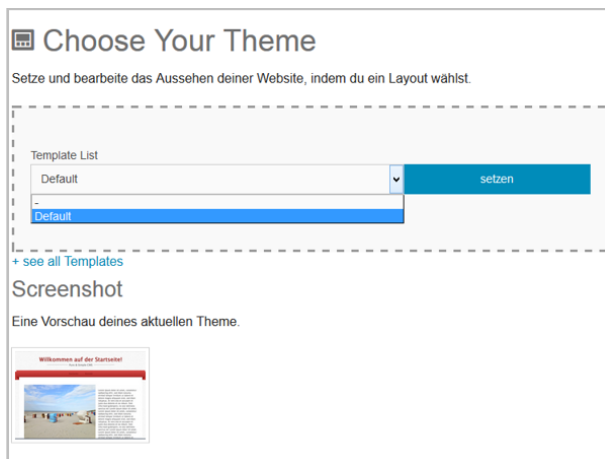


Abbildung 11: Ein neues Template setzen

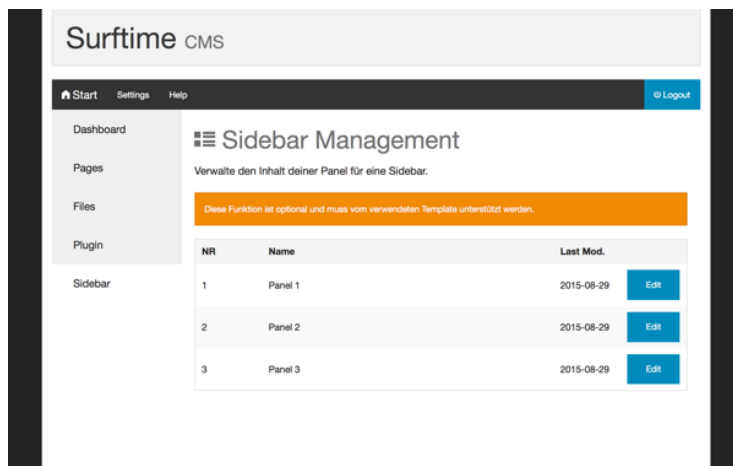
Erweitere Funktionen oder Anwendungen durch Plugin

Um auf individuelle Wünsche, nach speziellen Anwendungen eingehen zu können, ist diese Software durch Plugin erweiterbar. Ein neues Plugin kann per FTP in den Ordner public > plugin geladen werden. Hierzu benötigst du wieder ein FTP-Programm wie beispielsweise Filezilla. Wichtig ist das der Name des Ordner, in dem sich das Plugin befindet, auch gleichzeitig der Name der PHP-Funktion ist. Im Content (Page Body) ist diese Funktion nun mit folgendem Platzhalter aufrufbar **{%FUNCTION|name_func|arg1|arg2%}**. Wenn du diese bearbeiten willst, gehe auf Start > Plugin im Administratorbereich. Das laden und aufrufen von Plugin ist nur für fortgeschrittene Benutzer mit entsprechender Fachkenntnis geeignet.

Inhalte für eine Sidebar editieren (optional)

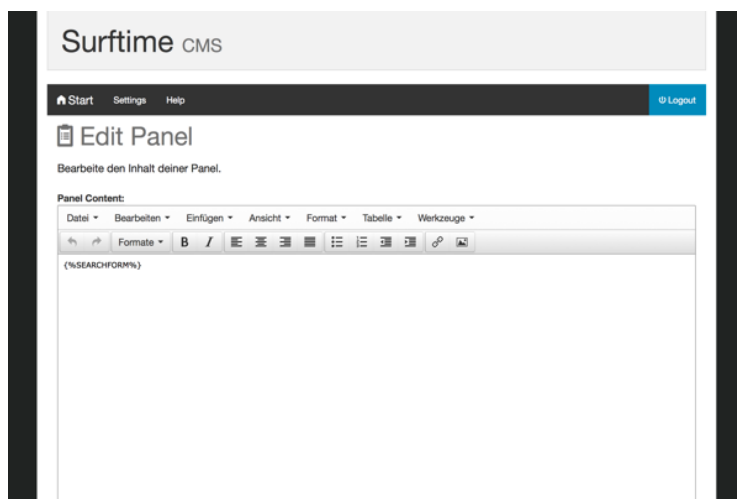
Es können optional auch Inhalte einer Sidebar editiert werden. Limitiert ist dies auf drei Stück. Gehe auf Start > Sidebar und wähle durch Klick auf den Button Edit das zu ändernde Panel.

Abbildung 12: Auswahl des zu ändernden Panel



Trage deinen Inhalt in das Formularfeld ein, und speichere anschließend alles. Du kannst bei deinen Inhalten, ebenso wie im Page-Body Variablen nutzen. Sichtbar wird dein Inhalt im vorbestimmten Bereich des Template (Design deiner Seite).

Abbildung 14: Editor (WYSIWYG) für deine Inhalte



Wichtig: Dein verwendetes Template muss diese Funktion unterstützen!

Schaue hierzu in die Hinweise und Informationen zum Template, oder wende dich an dessen Author.

Systemeinstellungen ändern

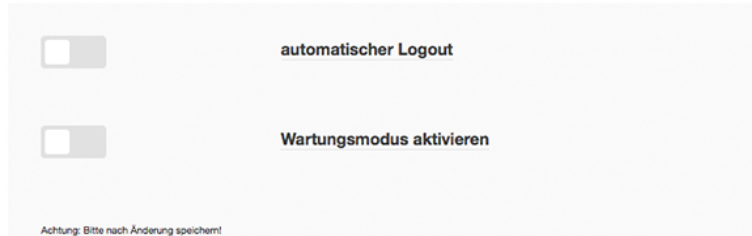
Hier kannst du Verhalten und Darstellung deines System verändern.

1. **automatischer Logout** Bestimme ob du nach einer bestimmten Zeit automatisch vom Administratorenbereich abgemeldet wirst. Schalte es ab, wenn du zum Beispiel an einem größeren Artikel arbeitest.
2. **Wartungsmodus aktivieren** Du kannst deine Website in einen Wartungsmodus setzen. Es wird ein 503 HTTP-Header gesendet und im Frontend eine Wartungsseite ausgegeben.

Wichtig: speichere seine Einstellung nach einer Änderung ab!

System

Mit diesen Einstellungen kannst du Verhalten und Darstellung deines System verändern.



☐ automatischer Logout

☐ Wartungsmodus aktivieren

Achtung: Bitte nach Änderung speichern!

Abbildung 15: Schalter zum setzen der Systemeinstellungen

Im Punkt Medien kannst du Angaben zur Bildbreite und Bildhöhe machen. Alle Bilder werden nach einem Upload auf die hier voreingestellte maximale Größe verkleinert. Dateien die unter die eingestellte Größe fallen, werden im Original beibehalten.

Medien



maximale Breite in Pixel: Bildbreite

533

maximale Höhe in Pixel: Bildhöhe

400

speichern

Abbildung 16: maximale Breite und Höhe der hochgeladenen Bilder festlegen

Developer's Guide

Plugin erstellen

Richtlinien:

- Plugin werden im Ordner root > public > plugin akzeptiert.
- Der Name des Ordner muss immer gleich der Funktion sein.
- Ausgeführt werden Plugin als Funktion auf der index.php im Pluginordner
- im gesamten Plugin dürfen keine Ausgaben durch echo oder print gemacht werden.
- Alle Dateien müssen derzeit noch per FTP auf den Server geladen werden.

Ein Beispiel: Erstelle ein Verzeichnis **my_plugin** im Pfad ./public /plugin/ auf deinem Server. Lege in diesem Ordner eine Datei index.php an. Dein Plugin wird automatisch mit der Funktion Namens my_plugin gestartet. Um eine Ausgabe zu machen, nutze return.

```

Datei index.php
<?php
function my_plugin(){
    // Plugin Data ...
    return 'say hello';
}
?>

```

Dieses einfache Plugin lässt sich nun im Content deiner Seite durch den Platzhalter {%FUNCTION|my_plugin%} aufrufen. Als Ausgabe erhält man *say hello*.

Parameter übergeben

Du kannst deinem Plugin auch Parameter übergeben. Am einfachsten geht dies im verwendeten Platzhalter nach dem Name der Funktion. Nutze als Trenner einen senkrechten Strich. Es können beliebig viele Parameter übergeben werden.

z.B.:

```
{%FUNCTION|my_plugin|param1|param2|param3%}
```

Innerhalb deiner Funktion kannst du die Parameter als numerischen **Array** aufrufen. Im Beispiel erhält man folgende Inhalte am aufgerufenen Schlüssel.

```

<?php
function my_plugin($param){
    $param[0]; // enthält param1
    $param[1]; // enthält param2
    $param[2]; // enthält param3
    // Plugin Data ...
}
?>

```

Daten auf jeder Seite im <head> oder am Ende des <body> ausgeben

Um aus deinem Plugin heraus die Metadaten zu Erweitern steht die statische Method `\Model\LoadExtensions::AddHTML()` zur Verfügung. Ausgeführt wird diese wieder auf der `index.php` innerhalb deines Plugin. Übergeben werden zwei Parameter. Im ersten erfolgt die Angabe wo hin etwas geschrieben werden soll (`head` oder `body`). Im zweiten folgt der Inhalt.

```
\Model\LoadExtensions::AddHTML('head', '  
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>');
```

System interne Bibliotheken auf jeder Seite im <head> ausgeben

Surftime CMS bietet dir den Aufruf interner Bibliotheken (API). Diese werden im `<head>` deines Quelltext eingebunden. Die Ausgabe erfolgt vor den Daten, welche per Methode `LoadExtension` an den `'head'` übergeben werden. Damit ist sicher gestellt das z.B. eine `jQuery` Bibliothek für deine eigenen Scripte zur Verfügung steht. Selbst bei mehrfachen Aufruf durch unterschiedliche Plugin wird die jeweilige Bibliothek nur einmal eingebunden.

```
\Model\LoadExtensions::AddHTML('library','jQuery');
```

Wichtig: Mehr erfährst du auch im Kapitel "Interne Bibliotheken und Framework nutzen"! Hier wird unter anderem beschrieben welche Bibliotheken dir zur Verfügung stehen.

Datenbankanbindung Innerhalb des Plugin

Es gibt mehrere Werkzeuge um übliche, allerdings teils einfache, Datenbankanwendungen zu starten. Hierzu gehören

1. SELECT
2. UPDATE
3. INSERT
4. DELETE

Es ist nicht notwendig komplett SQL Statement zu schreiben, oder sich um eine Datenbankverbindung zu kümmern. Vielmehr müssen für eine Abfrage nur wenige Parameter an eine Methode übergeben werden. Diese liefert bei einem `SELECT` das Ergebnis als `Array`. Bei einem `UPDATE`, `INSERT` oder `DELTE` gibt es als Rückgabe nur boolsches `TRUE` bzw. `FASLE`.

Wichtig: Um alle Methoden verwenden zu können muss die Klasse `HandlePluginData` instanziiert werden. Aus diesem Objekt lassen sich nun alle nachfolgend benannten Methoden ableiten.

```
$plugin = new \Model\HandlePluginData();
```

MySQL SELECT ausführen

Beispiel 1: Alle Datensätze aus einer Tabelle holen

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    // Startet eigentliche Datenbankabfrage wie SELECT * FROM 'tablename'
    $request = $data->selectDB($table);
    // Rückgabe erfolgt als Assoziativer Array und kann so weiter verarbeitet werden
    // z.B. mit einer Schleife
    foreach($request as $key => $value){
        // usw...
    }
}
?>
```

Bisher haben wir immer alle Datensätze abgefragt, aber zum Glück können wir auch gezielt einzelne Datensätze abfragen. Um die Ergebnisse zu begrenzen benutzen wir die WHERE Klausel. Um Datensätze zu erfassen die einer oder mehreren Regeln entsprechen übergeben wir diese in einem assoziativen Array. Hier entspricht der Key dem Name der Spalte und Value dem einzugrenzenden Wert. Übergeben wird dieser Array an zweiter Stelle der Methode selectDB.

Wichtig: Bei Angabe mehrerer Regeln werden diese immer mit einem AND verknüpft. Ein logisches ODER als Verknüpfung beziehungsweise eine Negation ist nicht vorgesehen.

Beispiel 2: Abgefragte Datensätze eingrenzen durch WHERE Regel

```

<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrWhere = array(
        'color' => 'red',
        'size' => 'large'
    );
    // ergibt: SELECT * FROM 'tablename' WHERE color = 'red' AND size = 'large'
    $request = $data->selectDB($table, $arrWhere);
    // Rückgabe erfolgt als Assoziativer Array und kann so weiter verarbeitet werden
    // z.B. mit einer Schleife
    foreach($request as $key => $value){
        // usw...
    }
}
?>

```

Um eine Ausgabe zu sortieren übergibt man dem Array `$arrWhere` einen Schlüssel `orderBy`, welcher mit dem Wert eines weiteren numerischen Array belegt ist. Hier ergibt sich aus dem ersten Schlüssen [0] die betreffende Spalte. Am zweiten Schlüssel [1] erwartet die Methode ob absteigend mit DESC oder aufsteigend mit ASC ausgegeben werden soll.

Beispiel 3: Suchergebnisse sortiert ausgeben lassen

```

<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrWhere = array(
        'color' => 'red',
        'orderBy' => array('size', 'DESC') //DESC = absteigend, ASC = aufsteigend
    );
    // ergibt: SELECT * FROM 'tablename' WHERE color = 'red' ORDER BY size DESC
    $request = $data->selectDB($table, $arrWhere);
    // Rückgabe erfolgt als Assoziativer Array und kann so weiter verarbeitet werden
    // z.B. mit einer Schleife
    foreach($request as $key => $value){
        // usw...
    }
}
?>

```

Bestimmte Spalten zu selektieren oder einen COUNT (Zähler) zu verwenden ist natürlich auch möglich. Hier übergibt man der Methode selectDB einen dritten, optionalen Parameter. Die Übergabe unserer Werte erfolgt in einem numerischen Array.

Wichtig: Wenn keine WHERE Klausel verwendet wird, muss dieser Parameter als leerer Array an die Methode übergeben werden.

Beispiel 4: Bestimmte Spalten auslesen

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrWhere = array(
        'size' => 'large'
    );
    $arraSelect = array('color', 'COUNT(size) AS s');
    // ergibt: SELECT color, COUNT(size) AS s FROM 'tablename' WHERE size = 'large'
    $request = $data->selectDB($table, $arrWhere, $arraSelect);
    // Rückgabe erfolgt als Assoziativer Array und kann so weiter verarbeitet werden
    // z.B. mit einer Schleife
    foreach($request as $key => $value){
        // usw...
    }
}
?>
```

Das Gruppieren bestimmter Datensätze ist ebenfalls möglich. Dies wird häufig benötigt, da nicht nur einzelne Informationen, sondern auch Zusammenfassungen gewünscht werden. Die MySQL Klausel GROUP BY wird ähnlich wie ORDER BY ausgelöst. Verwendet wird gleicher Array, nur mit dem Key groupBY.

Beispiel 5: Elemente bei der Ausgabe gruppieren

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrWhere = array(
        'size' => 'large',
        'groupBy' => 'name'
    );
    $arrSelect = array('color', 'COUNT(size) AS s');
    // ergibt: SELECT color, COUNT(size) AS s FROM 'tablename' WHERE size = 'large'
    // GROUP BY name
    $request = $data->selectDB($table, $arrWhere, $arrSelect);
    // Rückgabe erfolgt als Assoziativer Array und kann so weiter verarbeitet werden
    // z.B. mit einer Schleife
    foreach($request as $key => $value){
        // usw...
    }
}
?>
```

MySQL INSERT ausführen

Neue Datensätze in eine Tabelle zu schreiben ist genau so einfach wie das Auslesen. Nur wird hier eine andere Methode angesprochen. Instanziiere wieder das Objekt HandlePluginData und rufe die Methode insertDB auf. Da eine Eingabe ohne Daten keinen Sinn macht, ist die Übergabe dieser Pflicht. Folgende Parameter sind also zu übergeben:

1. Name der Tabelle (String)
2. Execute Array

Wichtig: Beachte das der Key des Execute Array gleich der Spalte in der Tabelle sein Muss.

Beispiel 6: Einfachen INNER JOIN realisieren

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrExec = array(
        'size' => 'large',
        'color' => 'red'
    );
    // ergibt: INSERT INTO tablename (size, color) VALUES ('large', 'red')
    $request = $data->insertDB($table, $arrExec);
    // Rückgabe erfolgt als TRUE oder FALSE und kann so weiter verarbeitet werden...
?>
```

MySQL UPDATE ausführen

Einträge zu ändern ist ebenfalls sehr einfach. Realisiere dies durch Aufruf der Methode `updateDB`, nach dem Instanziiieren der Klasse `HandlePluginData`. Die Methode erwartet 2 Parameter als Pflichtangabe. Zuerst den Name der Tabelle, in der Änderungen erfolgen sollen. Danach in einem assoziativen Array das Schlüssel-Werte-Paar, wobei der Key den Name der Spalte bezeichnet.

Wichtig: Es werden alle Inhalte der Tabelle mit den übergebenen Werten geändert.

Beispiel 7: Alle Inhalte einer Tabelle bestimmter Spalten ändern

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrExec = array(
        'timestamp' => 'NOW()',
        'status' => 'offline'
    );
    // ergibt: UPDATE tablename SET timestamp = NOW(), status = 'offline'
    $request = $data->updateDB($table, $arrExec);
    // Rückgabe erfolgt als TRUE oder FALSE und kann so weiter verarbeitet werden...
?>
```

Natürlich ist es häufig nicht gewünscht alle Datensätze mit den gleichen Werten zu belegen. Daher wird wieder eine WHERE Klausel benötigt. Diese wird optional als dritter Parameter an die Methode `updateDB` übergeben. Erwartet wird an dieser Stelle ein assoziativer Array, wobei der Key dem Name der Spalte entspricht.

Beispiel 8: Bestimmten Datensatz in einer Tabelle ändern

```

<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrExec = array(
        'status' => 'offline'
    );
    $arrWhere = array(
        'id' => '1'
    );
    // ergibt: UPDATE tablename SET status = 'offline' WHERE id = '1'
    $request = $data->updateDB($table, $arrExec, $arrWhere);
    // Rückgabe erfolgt als TRUE oder FALSE und kann so weiter verarbeitet werden...
?>

```

MySQL DELETE ausführen

Hin und wieder ist es notwendig Datensätze aus einer Tabelle zu löschen, oder gar die gesamte Tabelle zu leeren. Hierfür steht wieder eine Methode nach instanziiieren der Klasse HandlePluginData mit Namen deleteDB zur Verfügung. Diese erwartet als ersten Parameter den Namen der Tabelle. Ohne Angabe des optionalen zweiten Parameter werden so alle Datensätze aus der Tabelle gelöscht. Übergibt man den zweiten Parameter als assoziativen Array, so wird eine WHERE Bedingung auf den Datensatz angewendet und dieser bzw. diese gezielt gelöscht.

Wichtig: Du solltest mit dem Befehl DELETE sehr vorsichtig umgehen, da die Datensätze nach dem Löschen nicht mehr wieder hergestellt werden können. Sei also sicher, ob Du die Daten wirklich nicht mehr benötigst.

Beispiel 9: Datensätze löschen

```

<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrWhere = array(
        'id' => '1'
    );
    // ergibt: DELETE FROM tablename WHERE id = '1'
    $request = $data->deleteDB($table, $arrWhere);
    // Rückgabe erfolgt als TRUE oder FALSE und kann so weiter verarbeitet werden...
?>

```

MySQL CREATE TABLE ausführen

Zum Erstellen eines Datenbank gestützten Plugin ist es natürlich notwendig bei Installation entsprechende Tabellen anzulegen. Einzige Möglichkeit ist die Nutzung der Methode `createDB`. An diese wird ein assoziativer Array übergeben. Dieser muss eine bestimmte Struktur aufweisen. Daher sollte man sich vorab Gedanken machen wie die eigentliche Anweisung zerlegt werden muss.

Folgendes Beispiel zeigt: Struktur Array zum Erstellen einer Tabelle mit der Methode `createDB`

```
$arrExec = array(
    'column_definition' => array(
        // key = col_name
        'id' => array('int(10)', 'NOT NULL', 'AUTO_INCREMENT'),
        'ssid' => array('varchar(255)', 'NOT NULL'),
        'timestamp' => array('int(11)', 'NOT NULL'),
        'message' => array('text', 'NOT NULL'),
        'user' => array('enum(\'user\',\'admin\')', 'NOT NULL'),
        'status' => array('enum(\'online\',\'offline\')', 'NOT NULL', 'DEFAULT \'offline\''),
        'PRIMARY KEY' => array('(id)')
    ),
    'create_definition' => array(
        'engine' => 'MyISAM',
        'default charset' => 'utf8'
    )
);
```

Voran erstellter Array ergibt folgende SQL-Anweisung nach Übergabe an Methode `createDB`

```
CREATE TABLE IF NOT EXISTS new_chat (
    id int(10) NOT NULL AUTO_INCREMENT ,
    ssid varchar(255) NOT NULL ,
    timestamp int(11) NOT NULL ,
    message text NOT NULL ,
    user enum('user','admin') NOT NULL ,
    status enum('online','offline') NOT NULL DEFAULT 'offline' ,
    PRIMARY KEY (id) )
ENGINE=MyISAM DEFAULT CHARSET=utf8
```

Ablauf der Installation im Detail. Übergebe nach instanziiieren der Klasse

`HandlePluginData` an die Methode `createDB` als ersten Parameter den Name der Tabelle. Als zweiten Parameter erwartet die Methode den vorab behandelten Datenarray. Nach dem Ausführen erhält man `TRUE` oder `FALSE` als Statusmeldung.

Beispiel 10: Eine Datenbanktabelle installieren

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle
    $table = 'tablename';
    $arrExec = array(
        'column_definition' => array(
            // key = col_name
            'id' => array('int(10)', 'NOT NULL', 'AUTO_INCREMENT'),
            'color' => array('varchar(255)', 'NOT NULL'),
            'timestamp' => array('int(11)', 'NOT NULL'),
            'description' => array('text', 'NOT NULL'),
            'size' => array('enum(\'small\', \'large\')', 'NOT NULL'),
            'PRIMARY KEY' => array('(id)')
        ),
        'create_definition' => array(
            'engine' => 'MyISAM',
            'default charset' => 'utf8'
        )
    );
    // Starte SQL-Query
    $request = $data->createDB($table, $arrExec);
    // Rückgabe erfolgt als TRUE oder FALSE und kann so weiter verarbeitet werden...
?>
```

Abschließend sei gesagt, das bei einer geringen Datenmenge und wenigen Datenänderungen von einer Speicherung in der Datenbank abgesehen werden sollte. Willst du also konstante Daten und Zustände speichern ist eine XML-Datei im Pluginordner von Vorteil.

MySQL Query ausführen

Manchmal muss es aber eine individuelle SQL-Anweisung sein. Die Lösung bildet folgende Methode queryDB(). Als ersten Parameter wird die SQL-Anweisung übergeben. Beim Zusammenstellen dieser muss auf den Tabellennamen geachtet werden. Dieser erfordert stets ein voran setzen des Tabellenpräfix. Der zweite Parameter ist der Execute-Array (optional).

Beispiel 11: MySQL Query ausführen

```
<?php
function my_plugin(){
    // Objekt instanziiieren aus der Klasse HandlePluginData
    $data = new \Model\HandlePluginData();
    // Abzufragende Tabelle mit Tabellenpräfix übergeben
    $table = TBL_PRFX.'tablename';
    // SQL Anweisung formulieren
    $sql = 'SELECT * FROM '.$table;
    $arrExec = array(
        'color' => 'red',
        'status' => 'offline'
    );
    $request = $data->queryDB($sql, $arrExec);
    // Rückgabe erfolgt als Datenarray bzw. TRUE oder FALSE
?>
```

AJAX Anbindung

Um Werte von deiner AJAX-Schnittstelle entgegen zu nehmen, oder zu schreiben bedarf es der Funktion AjaxData(). Diese muss auf der ajax.php innerhalb deines Plugin definiert werden. Dein Plugin wird automatisch mit dem Funktionsaufruf durch das System gestartet. Um eine Ausgabe zu machen, nutze return. An die Schnittstelle übermittelte GET-Parameter können als Array aus dem Parameter der Funktion gelesen werden. Angesprochen wird die Schnittstelle über

```
http://deinedomain.de/ajax.php?load=ajax&data=plugin&extension=my_plugin&action=set
```

Beispiel 12: Funktion zur Verarbeitung von Ajax auf der ajax.php erstellen

```
function AjaxData($param){

    switch($param['action']){
        case 'set':
            //Function für SET
            break;

        case 'get':
            //Function für GET
            break;
    }
    return 'success';
}
//Ausgabe auf der ajax.php wäre success
```

Wichtig: Sollen Daten als JSON Datei verarbeitet werden, so setze den GET-Parameter **load=json**. In diesem Falle muss von der Funktion AjaxData ein Array per return ausgegeben werden.

Template erstellen

Richtlinien:

- Template werden im Ordner root > public > templates akzeptiert.
- Der Name des Ordner ist gleich dem des Template.
- Ausgeführt wird als Standard immer die **index.tpl.php** im Templateordner
- Alle Dateien müssen derzeit noch per FTP auf den Server geladen werden.
- Lege alle Ordner und Dateien wie z.B. CSS oder Javascript innerhalb des Template ab

Erstelle den Quellcode auf der index.tpl.php aus gemischtem HTML und PHP.

Beispiel 13: Erstelle den Quellcode von mytemplate

```
mytemplate/index.tpl.php

<!doctype html>
<html>
<head>
    <meta charset="<?php echo $this->Charset; ?>">
    <title><?php echo $this->Title; ?></title>
    <meta name="description" content="<?php echo $this->Description; ?>">
    <meta name="keywords" content="<?php echo $this->Keywords; ?>">
    <meta name="robots" content="<?php echo $this->Indexation; ?>, follow">
usw...
```

Wie bereits zu erkennen ist, können Variablen aus dem System gelesen und wieder gegeben werden. Folgende Variablen stehen zur Verfügung:

Variable	Beschreibung der Ausgabe
<code>\$this->Charset</code>	Zeichensatz
<code>\$this->Title</code>	Titel deiner Seite
<code>\$this->Description</code>	Seitenbeschreibung zur Suchmaschinenoptimierung
<code>\$this->Keywords</code>	Keywords zur Suchmaschinenoptimierung
<code>\$this->Indexation</code>	Indexierungsstatus
<code>\$this->PathTemplate</code>	Pfad zu deinem Template
<code>\$this->Headline</code>	Kopfzeile bzw. Überschrift
<code>\$this->Slogan</code>	Slogan deiner Seite
<code>\$this->Content</code>	Inhalt deiner Seite
<code>\$this->Firstname</code>	Unter <i>Settings</i> gespeicherten Vornamen
<code>\$this->Lastname</code>	Unter <i>Settings</i> gespeicherten Nachnamen
<code>\$this->Street</code>	Unter <i>Settings</i> gespeicherte Straße
<code>\$this->Postalzip</code>	Unter <i>Settings</i> gespeicherte Postleitzahl
<code>\$this->City</code>	Unter <i>Settings</i> gespeicherte Stadt
<code>\$this->Phone</code>	Unter <i>Settings</i> gespeicherte Telefonnummer
<code>\$this->Email</code>	Unter <i>Settings</i> gespeicherte E-Mail-Adresse
<code>\$this->Company</code>	Unter <i>Settings</i> gespeicherter Firmenname
<code>\$this->Opening</code>	Unter <i>Settings</i> gespeicherte Öffnungszeiten
<code>\$this->Variable</code>	Unter <i>Settings</i> gespeicherte Variablen Inhalte
<code>\$this->Panel1</code>	Unter <i>Edit Panel</i> gespeicherter Inhalt für Panel 1
<code>\$this->Panel2</code>	Unter <i>Edit Panel</i> gespeicherter Inhalt für Panel 2
<code>\$this->Panel3</code>	Unter <i>Edit Panel</i> gespeicherter Inhalt für Panel 3
<code>\$this->Searchform</code>	HTML-Quellcode für dein Suchformular
<code>\$this->Menu('Navigation')</code>	Hauptnavigation
<code>\$this->Menu('SingleLink', 'index', 'Home')</code>	einzelner Link / Hinweis: bitte begrenzt nutzen
<code>\$this->Library('jQuery')</code>	Aufruf einer System internen Bibliothek (z.B. jQuery)*

* Welche Bibliotheken verfügbar sind erfährst du im Kapitel "Interne Bibliotheken und Framework nutzen"

Beschreibung und Urheberrechtshinweis zum Template hinzufügen

Die geschieht durch Hinterlegen einer Textdatei readme.txt im Pluginordner. Beim Aktivieren des Plugin wird der Inhalt dieser direkt unter dem Vorschaubild ausgegeben.

Vorschaubild hinzufügen

Um das Layout schon vor dem Aktivieren zu sehen ist es möglich einen Screenshot deines Template zu hinterlegen. Lade diesen als screenshot.png in den Templateordner.

Eine statische Seite erstellen

Um einer bestimmten Seite statische Inhalte oder ein abweichendes Layout zu zuordnen, legt man eine PHP-Datei mit entsprechendem Dateinamen im Templateordner an. Nun muss im Administratorbereich eine neue Seite mit gleichnamiger Slug/URL erstellt werden, falls diese hier noch nicht existiert.

Ein Beispiel: Die **index** (Startseite) soll ein abweichendes Layout mit zusätzlichen statischen Inhalten erhalten. Lege im Templateordner eine neue index.php an. Befülle diese mit dem gewünschten Quellcode. Sobald die Datei auf dem Server bereit steht, liefert das CMS den neuen Inhalt beim Aufruf der Startseite z.B. <http://www.deinedomain.de> aus.

Interne Bibliotheken und Framework nutzen

In den voran gegangenen Kapiteln ist bereits kurz auf die Nutzung, einer System internen Bibliothek (per API), eingegangen wurden. Dies soll hier noch einmal ausführlich erläutert werden. Für viele Anwendungen muss man das Rad nicht neu erfinden. Es existieren unzählige Framework auf dem Markt. Diese erleichtern die Arbeit ungemein. Um es noch einfacher zu machen, sind bereits ein paar der beliebtesten Bibliotheken im System hinterlegt, und über eine API aufrufbar.

Der Aufruf erfolgt über zwei Möglichkeiten. Einmal können diese im Template eingebunden werden. Die zweite Möglichkeit ist der Aufruf direkt im Plugin. Mit dem Aufruf steuerst du auch die Position der Ausgabe im Quelltext.

Beispiel für die Ausgabe im Plugin

Um aus deinem Plugin heraus die Metadaten zu Erweitern steht die statische Method `\Model\LoadExtensions::AddHTML()` zur Verfügung. Übergibt man dieser Methode als ersten Parameter 'library' und als zweiten den Namen der Bibliothek, so wird diese im `<head>` deines Quelltext ausgegeben. Die Ausgabe erfolgt vor den Daten, welche per Methode LoadExtension an den 'head' übergeben werden. Damit ist sicher gestellt das z.B. eine jQuery Bibliothek für deine eigenen Scripte zur Verfügung steht. Selbst bei mehrfachen Aufruf durch unterschiedliche Plugin wird die jeweilige Bibliothek nur einmal eingebunden.


```
\Model\LoadExtensions::AddHTML('library','jQuery');
```

Ausgabe :

```
<!-- Add Library jQuery-->
```

```
<script src="./includes/library/jquery/jquery-2.1.4.min.js" type="text/javascript"></script>
```

Beispiel für die Ausgabe im Template

Für Entwickler eines Template ist das Einbinden ebenso einfach. Füge folgenden Codeschnipsel an die gewünschte Stelle für eine Ausgabe der Bibliothek:

Beispiel 14: Einbinden des Framework "Foundation" im Template

```
<?php echo $this->Library('Foundation'); ?>
```

Ausgabe :

```
<!-- Add Library Foundation-->
```

```
<link rel="stylesheet" href="./includes/library/Foundation/foundation-icons.css" />
```

```
<link rel="stylesheet" href="./includes/library/Foundation/foundation.css" />
```

Übersicht verfügbarer Bibliotheken und Framework

Bezeichnung	Typ	Beschreibung	Anwendung / in Tag
Foundation	CSS	Foundation ist ein Front-End-Framework. Es dient zur schnellen Erstellung von responsiven Websites.	innerhalb des <head>
FoundationJS	Javascript	Optionale Javascript Erweiterung des Foundation-Framework. Achtung: Funktion nur in Zusammenhang mit Foundation	vor dem schließenden <body>
Normalize	CSS	Normalize.css ist ein Stylesheet mit dem man alle Browser-Grundwerte standardisieren kann.	innerhalb des <head> (vor allen CSS-Dateien)
jQuery	Javascript	jQuery ist eine JavaScript-Bibliothek, die Funktionen zur DOM-Navigation und -Manipulation zur Verfügung stellt.	wahlweise <head> oder <body>, vor anderen Javascript-Dateien(Code)
jQueryUI	CSS/Javascript	jQuery-Benutzeroberfläche - Bibliothek für Interaktionen, Widgets, Effekte und Themen für die Erstellung von Rich Internet Applications.	innerhalb des <head>
Modernizr	Javascript	Modernizr ist eine Javascript-Bibliothek, dessen hauptsächliche Aufgabe es ist, zu testen, ob ein Browser bestimmte HTML5- oder CSS3-Features unterstützt.	innerhalb des <head>
Lightbox	CSS/Javascript	Präsentation von Fotos und Bildern.	innerhalb des <head>

Danksagung

Schön das du dich für Surftime CMS entschieden hast. Es wurde viel Zeit und Mühen in die Erstellung gelegt. Alle Funktionen und Anwendungen wurden mit größter Sorgfalt programmiert. Solltest du doch etwas finden das verbesserungswürdig ist, so freue ich mich jederzeit über ein Feedback.

Und nun viel Spaß bei der Nutzung deiner neuen Website mit Surftime CMS!