Aufgabensammlung zu ICT-BBCH Modul M133

Version BIVO 2014

Inhaltsverzeichnis

[1 Lokaler Server einrichten (Kap 2) 2](#_Toc458599331)

[2 Serverseitige Konfigurationen (Kap 2) 3](#_Toc458599332)

[3 Werkzeuge und Erweiterungen (Kap 2) 4](#_Toc458599333)

[4 Versionierung und Kontrolle (Kap 2) 5](#_Toc458599334)

[5 PHP Code und HTML (Kap 3 + 8) 6](#_Toc458599335)

[6 Variablen und Konstanten (Kap 3 + 8) 7](#_Toc458599336)

[7 Variablen in PHP einsetzen (Kap 3 + 8) 8](#_Toc458599337)

[8 Arrays (Teil 1) (Kap 3, 8 + 9) 9](#_Toc458599338)

[9 Arrays (Teil 2) (Kap 3, 8 + 9) 9](#_Toc458599339)

[10 Kontrollstrukturen (Teil 1) (Kap 3 + 8) 10](#_Toc458599340)

[11 Kontrollstrukturen (Teil 2) (Kap 3 + 8) 11](#_Toc458599341)

[12 Kontrollstrukturen (Teil 3) (Kap 3 + 8) 12](#_Toc458599342)

[13 Kontrollstrukturen & Parameterübergabe (Kap 3 + 8) 13](#_Toc458599343)

[14 Parameterübergabe (Teil 1) (Kap 3 + 8) 14](#_Toc458599344)

[15 Parameterübergabe (Teil 2) (Kap 3 + 8) 15](#_Toc458599345)

[16 Basic Repetitorium (Teil 1) (Kap 3 + 8) 16](#_Toc458599346)

[17 Basic Repetitorium (Teil 2) (Kap 3 + 8) 17](#_Toc458599347)

[18 Cross-Site Scripting (Kap 3) 18](#_Toc458599348)

[19 Datenintegrität (Kap 4 + 7) 19](#_Toc458599349)

[20 Zeitfunktionen (Kap 9) 20](#_Toc458599350)

[21 Suchen, Finden und Ersetzen (Kap 9) 21](#_Toc458599351)

[22 Funktionen für Arrays (Kap 9) 22](#_Toc458599352)

[23 Datei- und Verzeichnisfunktion I (Kap 9) 23](#_Toc458599353)

[24 Datei- und Verzeichnisfunktion II (Kap 9) 24](#_Toc458599354)

[25 Cookies & Sessions I (Kap 9) 25](#_Toc458599355)

[26 Cookies & Sessions II (Kap 9) 26](#_Toc458599356)

[27 Reguläre Ausdrücke I (Kap 10) 27](#_Toc458599357)

[28 Reguläre Ausdrücke II (Kap 10) 28](#_Toc458599358)

[29 Einführung Projekt «Kundenerfassung» (Fallstudie) 29](#_Toc458599359)

[30 Spezifikation erstellen (Fallstudie) 30](#_Toc458599360)

[31 Testplan erstellen (Kap 11) 31](#_Toc458599361)

[32 Kundenerfassung (Teil 1) (Kap 11) 32](#_Toc458599362)

[33 Kundenerfassung (Teil 2) (Kap 11) 33](#_Toc458599363)

[34 Kundenerfassung (Teil 3) (Kap 12) 34](#_Toc458599364)

[35 Sicherheitstests durchführen (Kap 12) 35](#_Toc458599365)

# Lokaler Server einrichten (Kap 2)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können sich die notwendige Arbeitsumgebung selbstständig bereitstellen und sicher Betreiben. |
| **Aufgabentyp** | Einzel- und Gruppenaufgabe & Präsentation |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Sie richten sich individuell Ihre Arbeitsumgebung ein. Konfigurationen und Sicherheitseinstellungen müssen überprüft und gegebenenfalls angepasst werden. |
| **Aufgabe** | * Installation von XAMPP. Erstellen Sie im Webserververzeichnis eine php Datei mit dem Namen info.php. Als Inhalt benutzen Sie die Funktion ***phpinfo().*** * Erstellen Sie eine Liste der wichtigsten Sicherheitsoptionen in der ***php.ini*** * Erstellen Sie eine Liste der wichtigsten Einstellungen in der ***httpd.conf*** * Prüfen Sie die PHP Einstellungen mittels ***phpinfo()***. * Erstellen Sie eine Präsentation über die wichtigsten Server Einstellungen, bezüglich Sicherheit und dem Einsatz als lokale Entwicklungsumgebung. |
| **Hinweis Hilfsmittel** | * Sprechen Sie sich im Plenum ab, über welche Aspekte der Sicherheit und Konfiguration Sie einen Kurzvortrag halten. |
| **Ergebnis** | * Installierte und lauffähige XAMPP-Installation * Die Datei info.php liegt vor * Präsentationen steht bereit |

# Serverseitige Konfigurationen (Kap 2)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können mit ***htaccess*** ihre Arbeitsumgebung einrichten, konfigurieren und sicher Betreiben. |
| **Aufgabentyp** | Einzel- und Gruppenaufgabe & Präsentation |
| **Zeitaufwand** | 45 Min. Vortrag 6 x ~5 Min = Total 75 min |
| **Ausgangslage** | Der Apache ermöglicht die dezentrale Verwaltung der Konfiguration mittels spezieller Dateien innerhalb des Web-Verzeichnisbaums. Diese speziellen Dateien heissen .***htaccess***. Damit stehen ihnen mächtige Konfigurationsmöglichkeiten zur Verfügung. |
| **Aufgabe** | Erstellen Sie eine der aufgelisteten Aufgaben in Ihrer lokalen Umgebung und präsentieren Sie die Lösung als Gruppe in einem Kurzvortrag.  Mögliche Themen:   * Verzeichnisschutz durch Passwort Login * HTTPS erzwingen * Ändern der Standardseite * Eigene Fehlermeldungen * Redirect * Kompression, cache und gzip |
| **Hinweis Hilfsmittel** | * Arbeiten Sie im neu zu erstellenden Verzeichnis «htaccess». * Erstellen Sie alle nötigen Dateien und Unterverzeichnisse um darin Ihre Lösungen zu speichern. |
| **Ergebnis** | * Es liegt eine eine ***.htaccess*** Datei vor. * Präsentationen steht bereit |

# Werkzeuge und Erweiterungen (Kap 2)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Editoren, Browser und Erweiterungen einrichten und sicher Betreiben. |
| **Aufgabentyp** | Einzelaufgabe, Plenum |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Um effizient Code zu schreiben ist es von Vorteil, eine optimal ausgerüstete Entwicklungsumgebung zu besitzen. Mit aktuellen Browsern und den wichtigen Addons kann der sequentielle Serverzugriff kontrolliert werden. Ihr Editor sollte alle wichtigen (PHP spezifischen) Erweiterungen enthalten. |
| **Aufgabe** | * Installieren Sie einen aktuellen Browser inkl. Entwicklerumgebung und nötige Plugins. * Installieren Sie einen Editor mit allen nötigen Erweiterungen. * Erstellen Sie zwei Listen mit den nötigen Erweiterungen für Editor und Browser |
| **Hinweis Hilfsmittel** | * Umfrage im Plenum bezüglich verwendeter und geeigneter Browser. * Recherche nach geeigneten Editoren und Erweiterungen. * Auswahl der Werkzeuge. Installation der nötigen Software. |
| **Ergebnis** | * Aktueller Browser mit Erweiterungen installiert * Geeigneter Editor mit Erweiterungen installiert |

# Versionierung und Kontrolle (Kap 2)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie kennen die Versionsverwaltung mit GIT. Sie können damit arbeiten, verwalten und sichern. |
| **Aufgabentyp** | Einzelaufgabe als Lernaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Damit Sie zu jedem Zeitpunkt auf Versionen und Änderungen Ihres Codes zugreifen können, sollten Sie ein Versionskontrollsystem (VCS) benutzen. VCS protokollieren Änderungen an Dateien und ermöglichen es Ihnen jederzeit zu einem vorherigen, funktionierenden Zustand zurückzugehen. |
| **Aufgabe** | * installieren Sie GIT. * erstellen Sie ein neues Repository im Verzeichnis Git. * kontrollieren Sie mit ***git status*** * Erstellen Sie zusätzlich eine Datei ***git.txt*** mit beliebigem Inhalt. * kontrollieren Sie mit ***git status*** * Nehmen Sie die Datei in den GIT Index auf und commiten Sie die Datei mit der Beschreibung «mein-Text». * kontrollieren Sie mit ***git status*** * Ändern Sie nun den Inhalt der ***git.txt*** * kontrollieren Sie mit ***git diff*** * Commiten Sie mit der Beschreibung «mein-neuer-Text». * kontrollieren Sie mit ***git log*** * Halten Sie den Ablauf im Moduljournal fest |
| **Hinweis Hilfsmittel** | * mit commit werden Änderungen im Head gespeichert. (git commit -m mein-text) |
| **Ergebnis** | * Sie haben einen Eintrag im Moduljournal zu GIT |

# PHP Code und HTML (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie kennen die allgemeine Regeln und Eigenschaften von PHP. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 15 Min. |
| **Ausgangslage** | Lernen Sie allgemeine Regeln und Eigenschaften von PHP kennen. Fügen Sie PHP in HTML ein und geben Sie mit PHP HTML Code aus. |
| **Aufgabe** | Erstellen Sie eine PHP Seite mit einem HTML Gerüst. Erstellen Sie einen vollständigen Programmkopf, der im HTML Quelltext nicht sichtbar ist. Generieren Sie im title Element den Titel mit PHP. Geben Sie im Body mittels echo Befehl folgenden Inhalt aus:  <p>Mit 'PHP' geht's besser</p> |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue Datei ***B14-133\_05.php*** und lösen Sie obige Aufgabe. Kommentieren Sie Ihren Code. |
| **Ergebnis** | * Die Datei ***B14-133\_05.php*** |

# Variablen und Konstanten (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können mit Variablen und Konstanten arbeiten. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Um Daten in den Speicher zu bekommen, werden diese in sogenannte Variablen (auch Container genannt) gehalten. Der Typ einer Variable kann, wie der Name schon sagt variieren. Konstanten hingegen sind unveränderlich. Diese werden einmal festgelegt und nicht mehr verändert |
| **Aufgabe** | Erstellen Sie einen Währungsrechner Prototyp für den TOPOMEDICS Online Shop. Definieren Sie die Konstante ***MWST*** mit dem Wert ***0.08*** |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue Datei ***B14-133\_06.php***. |
| **Ergebnis** | Die Datei ***B14-133\_06.php*** mit der Berechnung Bruttopreis inkl. MWST mit   * einer Konstante ***MWST*** * einer Variablen ***preis*** * einer Variablen ***gesamt*** |

# Variablen in PHP einsetzen (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Variablen in PHP gezielt und sicher einsetzen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 50 Min. |
| **Ausgangslage** | Um in Scripts Text und Zahlen behalten zu können, speichert man sie in Variablen ab. Diese bleiben während des Scripts bestehen, können verändert werden und später im Code wieder ausgelesen werden. |
| **Aufgabe** | Erstellen Sie eine neue, leere PHP Datei mit vollständigem HTML Gerüst.   * Erstellen Sie eine Variable ***color*** und weisen sie Ihr den Wert ***#999999*** zu. * Geben Sie nun den Inhalt der Variable ***color*** als Hintergrundfarbe der HTML-Seite aus. (CSS) * Probieren Sie nun, durch Ändern des Wertes der Variablen ***color*** die Hintergrundfarbe zu verändern. * Setzen Sie nun folgenden Code ein. Er generiert für die Variable ***color*** einen Zufallswert.   mt\_srand((double)microtime()\*1000000);  $zufallswert = mt\_rand(0, 16777215);  $color = "#" . dechex($zufallswert);   * Versuchen Sie herauszufinden, wie die Zahl 16777215 zustande kommt. Eine Hilfe könnte der Systemtaschenrechner mit seiner Dezimal-Hexadezimal Umrechnungsfunktion sein. * Ein solcher RGB Farbindex stellt ja ein "grau" dar, wenn alle drei RGB Werte gleichgross sind. Probieren Sie den obigen Code so abzuändern, dass die Seite nur noch Grauwerte annimmt. |
| **Hinweis Hilfsmittel** | * Erstellen Sie zwei neue Dateien und lösen Sie die Aufgabe. |
| **Ergebnis** | * Die Datei B14-133\_07-1.php für die ersten fünf Schritte * Die Datei B14-133\_07-1.php für die folgenden Schritte |

# Arrays (Teil 1) (Kap 3, 8 + 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Inhalte in Arrays erfassen und ausgeben |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 15 Min. |
| **Ausgangslage** | Bisher konnte jede Variable nur einen Wert speichern, in PHP können in einem Array beliebig viele Werte gespeichert werden, auch von unterschiedlichen Typen. |
| **Aufgabe** | Erstellen Sie eine neue PHP Datei mit einem assoziativen Array.   * Erzeugen Sie ein Array mit den Schlüsseln: vorname, nachname, alter, mail und füllen Sie das Array mit Ihren persönlichen Daten. * Erzeugen Sie ein mehrdimensionales Array, indem Sie zwei weitere Personen hinzufügen. * Preformatierte Ausgabe des Arrays in eine HTML Seite |
| **Hinweis Hilfsmittel** | * Speichern Sie ihre Arbeit unter ***B14-133-08.php***. |
| **Ergebnis** | * Die Datei B14-133-08.php, die die Arrays mit der Funktion print\_r("Variable") ausgibt. |

# Arrays (Teil 2) (Kap 3, 8 + 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Inhalte in Arrays erfassen und ausgeben |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Bisher konnte jede Variable nur einen Wert speichern. In PHP können in einem Array beliebig viele Werte gespeichert werden. Benutzen Sie für diese Aufgabe das Array der Aufgabe 3-4. |
| **Aufgabe** | Erstellen Sie eine neue PHP Datei und lösen Sie folgende Aufgaben:   * Sie wollen wissen wie viele Einträge Ihr Array hat, sie benutzen dazu „count“ * Sortieren Sie das Array in der Ausgabe alphabetisch * Entfernen Sie den ersten Schlüssel aus dem Array |
| **Hinweis Hilfsmittel** | * Speichern Sie ihre Arbeit unter ***B14-133-09.php***. |
| **Ergebnis** | * Die Datei ***B14-133-09.php***, die die Arrays wie in der Aufgabenstellung ausgibt. |

# Kontrollstrukturen (Teil 1) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Kontrollstrukturen in PHP erstellen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Um Programmteile zu überspringen oder beliebig zu wiederholen, setzt man Kontrollstrukturen ein, also Verzweigungen und Schleifen. |
| **Aufgabe** | * Geben Sie die Inhalte des untenstehenden Arrays in einer While Schleife aus.   $standorte = array(  "Bern",  "Zürich",  "St.Gallen",  "Shanghai",  "Kobe"  );   * Geben Sie das Array ***$standort*** in einer if-Verzweigung aus * Geben Sie die Inhalte des untenstehenden Arrays in einer While Schleife aus.   $mitarbeiter = array(  array("Bern",250),  array("Zürich",7),  array("St.Gallen",35),  array("Shanghai",30),  array("Kobe",50), |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-10.php***. |
| **Ergebnis** | * Die korrekt arbeitende Datei ***B14-133-10.php*** in Ihrem Verzeichnis. |

# Kontrollstrukturen (Teil 2) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Kontrollstrukturen in PHP erstellen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Um Programmteile zu überspringen oder beliebig zu wiederholen, setzt man Kontrollstrukturen ein, also Verzweigungen und Schleifen. |
| **Aufgabe** | Benutzen Sie das Array aus der Aufgabe 3-4. Geben Sie alle Namen und den Vornamen (Mit einer kleinen Begrüssung) in einer foreach Schleife aus.  Resultat im Browser:   * Guten Tag Mia Monaro * Guten Tag Jon Bent * Guten Tag Gita Gonzuetta |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-11.php***. |
| **Ergebnis** | * Die korrekt arbeitende Datei ***B14-133-11.php*** in Ihrem Verzeichnis. |

# Kontrollstrukturen (Teil 3) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Kontrollstrukturen gezielt einsetzen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Um Programmteile zu überspringen oder beliebig zu wiederholen, setzt man Kontrollstrukturen ein, also Verzweigungen und Schleifen. |
| **Aufgabe** | Erstellen Sie eine Variable mit dem Namen text, der Sie den Wert "TOPOMEDICS: Messbar besser" zuweisen. Erstellen Sie dann eine eigene Funktion mit dem Namen textumwandlung, die zwei Parameter verlangt. Die Funktion sollte einen String (den man als ersten Parameter übergibt) entweder komplett in Gross- oder in Kleinbuchstaben verwandeln, je nach Wert, den man als zweiten Parameter übergibt. Erstellen Sie also unter der Variable text diese Funktion und konstruieren Sie in der Funktion eine Schleife mit folgenden Aufgaben:   * Ist der zweite Parameter "g", soll der String (= erster Parameter) komplett in Grossbuchstaben verwandelt werden. * Ist der zweite Parameter "k", soll der String (= erster Parameter) komplett in Kleinbuchstaben verwandelt werden. * Ist der zweite Parameter weder "g" noch "k", soll eine Fehlermeldung zurückgegeben werden. * Das Resultat soll jeweils als Rückgabewert definiert werden.   Beim Aufruf textumwandlung($text, "g"); sollte folgende Ausgabe erscheinen:  TOPOMEDICS: MESSBAR BESSER  Beim Aufruf textumwandlung($text, "k"); sollte folgende Ausgabe erscheinen:  topomedics: messbar besser  Und beim Aufruf textumwandlung($text, "x"); sollte folgende Ausgabe erscheinen:  Achtung! Falscher Modifikationsparameter! Gültig sind nur "g" und "k"! |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-12.php***. |
| **Ergebnis** | * Eine Datei, welche die Resultate, wie sie oben beschrieben sind, vollständig und korrekt ausgibt. Speichern Sie diese Datei anschliessend in Ihrem persönlichen Verzeichnis. |

# Kontrollstrukturen & Parameterübergabe (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Kontrollstrukturen gezielt einsetzen |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Um Programmteile zu überspringen oder beliebig zu wiederholen, setzt man Kontrollstrukturen ein, also Verzweigungen und Schleifen. Zudem können über die GET Variable Inhalte übergeben werden. |
| **Aufgabe** | Erstellen Sie in einer neuen PHP Seite 4 Links in der Form:  <a href="B14-133-13.php?link=1">Link 1</a><br />  <a href="B14-133-13.php?link=2">Link 2</a><br /> usw.  Benutzen Sie die Funktion switch, um bei dem Aktivieren der Links einen unterschiedlichen Inhalt zu zeigen. |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-13.php***. |
| **Ergebnis** | * Die korrekt arbeitende Datei ***B14-133-13.php*** in Ihrem Verzeichnis. |

# Parameterübergabe (Teil 1) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie wissen, wie Sie Parameter sowohl mit HTML-Formularen als auch über die URL übergeben können. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 40 Min. |
| **Ausgangslage** | Grundsätzlich sind Variablen nur innerhalb eines Scripts gültig. Was bieten sich für Möglichkeiten, wenn man Werte von Variablen von einem Script zu einem Anderen übertragen möchte? |
| **Aufgabe** | Erstellen Sie ein HTML Formular mit 5 Radiobuttons und einem Submit-Button, das auf dieselbe Datei linkt. Zu jedem Radiobutton schreiben Sie eine Farbe (Tomato, Sienna, Grün, Blau, Grau). Folgende Aufgabe sollte die Datei erfüllen:   * klicken Sie einen dieser Radiobuttons an und drücken Sie den Submit-Button, sollte die Seite neu geladen werden, aber mit der entsprechenden Hintergrundfarbe. * Wird der Submit-Button geklickt, ohne dass ein Radiobutton angeklickt worden ist, sollte folgender Text erscheinen und die Hintergrundfarbe weiss sein.   „**Es wurde keine Hintergrundfarbe ausgewählt**.“ |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-14.php***. |
| **Ergebnis** | * Die korrekt arbeitende Datei ***B14-133-14.php*** in Ihrem Verzeichnis. |

# Parameterübergabe (Teil 2) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie wissen, wie Sie Parameter sowohl mit HTML-Formularen als auch über die URL übergeben können. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Grundsätzlich sind Variablen nur innerhalb eines Scripts gültig. Was bieten sich für Möglichkeiten, wenn man Werte von Variablen von einem Script zu einem Anderen übertragen möchte? |
| **Aufgabe** | Erstellen Sie ein Formular wo ein Username und ein Passwort eingegeben werden können. Senden Sie diese Informationen an eine zweite Seite. Bei korrektem Username und Passwort geben Sie bitte die Information aus:  „**Sie befinden sich im Administrationsbereich!“**  Bei falschem Login soll der User zurück zur ersten Seite geleitet werden, wo die Fehlermeldung erscheint:  „**Leider falsches Login**“ |
| **Hinweis Hilfsmittel** | * Erstellen Sie zwei neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-15-1.php***. und * ***B14-133-15-2.php*** |
| **Ergebnis** | * Die korrekt arbeitenden Dateien gemäss den genannten Angaben |

# Basic Repetitorium (Teil 1) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie sind in der Lage, die grundlegenden Elemente von PHP zielgerichtet bei einfachen Problemen einzusetzen. |
| **Aufgabentyp** | Repetitionsaufgabe, Einzelaufgabe |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Sie haben nun alle elementaren Methoden einmal angewendet. Wie sehen Ihre Problemlösungsstrategien bei kombinierten Problemen aus? |
| **Aufgabe** | * Erstellen Sie eine eigene Funktion **umrechnung**, die einen Parameter verlangt. Die Funktion sollte eine Zahl, die als Parameter übergeben wird, sowohl in das Binär- als auch ins Hexadezimalsystem umrechnen und als Rückgabewert hübsch dargestellt zurückgeben: * Geben Sie eine Titelzeile und die beiden Resultate untereinander aus. Es sollte folgendes im Browser zu sehen sein, wobei die Zahlen selbstverständlich durch den entsprechenden Parameter und die entsprechenden Ausrechnungen ersetzt werden soll. (Hier war der Parameter 24):   Umrechung der Dezimalzahl 24:  ins Binärsystem: 11000  ins Hexadezimalsystem: 18   * Erstellen Sie oben auf der Seite ein Formular mit einem Textfeld. Wenn eine Zahl im Textfeld eingegeben wurde, soll diese Zahl als Parameter bei der Funktion eingesetzt werden. |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue, leere PHP Datei * Speichern Sie ihre Arbeit unter ***B14-133-16.php***. * Probieren Sie Ihre Lösungen jeweils gleich in Ihrem lokalen Webserver aus. |
| **Ergebnis** | * Die Datei ***B14-133-16.php***, welche die Resultate, wie sie oben beschrieben sind, vollständig und korrekt ausgibt. Speichern Sie diese Datei anschliessend in Ihrem persönlichen Verzeichnis. |

# Basic Repetitorium (Teil 2) (Kap 3 + 8)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie sind in der Lage, die grundlegenden Elemente von PHP zielgerichtet bei einfachen Problemen einzusetzen. |
| **Aufgabentyp** | Repetitionsaufgabe, Einzelaufgabe |
| **Zeitaufwand** | 70 Min. |
| **Ausgangslage** | Sie haben nun alle elementaren Methoden einmal angewendet. Wie sehen Ihre Problemlösungsstrategien bei kombinierten Problemen aus? |
| **Aufgabe** | * Probieren Sie mit Hilfe der Random Funktion eine Funktion **wuerfel** zu generieren, die eine beliebige Zahl zwischen 1 und 6 zurückgibt. Parameter wird keiner verlangt. Erstellen Sie ein HTML-Formular auf derselben Seite, welches nur aus einem Submit-Button besteht. Mit Hilfe dieses Knopfes soll diese Funktion aufgerufen werden. Es soll nur eine Zufallszahl erscheinen, wenn dieser Knopf gedrückt worden ist. * Bauen Sie die Funktion Würfel der letzten Ausgabe zu einer Funktion **lotto** um. Diese neue Funktion soll eine Zufallszahl zwischen 1 und 49 ausgeben. * Erweitern Sie die Funktion Lotto nun so, dass immer 6 Zufallszahlen gleichzeitig dargestellt werden (5 Zahlen und eine Zusatzzahl). Die Zahlen dürfen aber jeweils nur einmal vorkommen. Erstellen Sie von Ihrer Programmidee zuerst ein Struktogramm und programmieren Sie dieses anschliessend aus! |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-17-1.php*** für Aufgabe 1 und 2 * die Datei ***B14-133-17-2.php*** für die Aufgabe 3. * Probieren Sie Ihre Lösungen jeweils gleich in Ihrem lokalen Webserver aus. |
| **Ergebnis** | * Die korrekt arbeitenden Dateien gemäss den genannten Angaben |

# Cross-Site Scripting (Kap 3)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Cross-Site Scripting erkennen und bekämpfen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Manipulationen von Formularen ist einfacher als manch einer sich das denkt. Die einfachste Möglichkeit ein Formular zu manipulieren, ist den HTML-Quellcode des Formulars zu kopieren und in einer lokalen Datei zu speichern. Zunächst könnte man davon ausgehen, dass ein XSS-Angriff das Problem des Clients ist, der ihn ausführt.  **Ist es so, dass wenn jemand in seinem eigenen Browser ein JavaScript-Popup erzeugt keine Gefahr für Ihre PHP-Anwendung besteht?**  Könnte der Angreifer die URL zum verwundbaren Skript nicht so modifizieren, dass sie XSS enthält? Diese URL muss er dann allen potenziellen Opfern so unterschieben, dass diese auf die merkwürdigen Zeichen nicht aufmerksam werden. Am einfachsten lässt sich dies mit einem Link bewerkstelligen. |
| **Aufgabe** | * Hacken Sie Ihre bisherigen Aufgaben, die Formularelemente beinhalten. Versuchen Sie in Ihren Arbeiten XSS Lücken zu finden. Manipulieren Sie die URL der Übung ***B14-133-14.php***. Versuchen Sie eine URL zu konstruieren die beim Öffnen im Browser ein Popup -Alert öffnet, mit dem Inhalt: «XSS -HACK!!!» * Beschreiben Sie die Problematik der gefundenen Lücke. Treffen Sie Gegenmassnamen. Erwartet werden zwei verschiedene Möglichkeiten um die Sicherheitslücke zu schliessen. |
| **Hinweis Hilfsmittel** | --- |
| **Ergebnis** | * Manipulierte Datei ***B14-133-14.php*** * Beschreibung der Lücke, inkl. 2 Lösungen um die XSS Lücke zu schliessen. |

# Datenintegrität (Kap 4 + 7)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können die Datenintegrität einer Datei überprüfen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 15 Min. |
| **Ausgangslage** | Um feststellen zu können, ob Daten seit einer letzten Kontrolle nicht verändert wurden, verwendet man sog. Hashfunktionen, die je nach Zustand einer Menge von Daten einen eindeutigen Wert zurückgeben. |
| **Aufgabe** | * Erstellen Sie eine neue Datei ***content.txt*** die Sie mit folgendem Text füllen: * «Dies ist die Information, die immer gleichbleibt! ». * Erstellen Sie eine neue PHP-Datei, die prüft, ob die Datei ***content.txt*** unverändert geblieben ist. * Wenn die Datei unverändert ist, soll folgender Text erscheinen: «Datei ok!» * Wenn die Datei aber nicht mehr der ursprünglichen Form entspricht (also verändert wurde) soll folgender Text erscheinen: «Datei wurde verändert!» * Um festzustellen, ob sich die Datei verändert hat, verwenden Sie die Funktion ***md5\_file***, die als Parameter den Dateinamen erwartet und den Hashwert zurückgibt. Damit Sie Ihre Routine eichen können, müssen Sie zuerst den Hash der Datei ermitteln. Als Rückgabewert erhalten Sie einen String in dieser Form: 39dedaf5b0a30b85bdc689fdfefac648 * Lassen Sie zu diesem Zweck den Hash am Bildschirm ausgeben und kopieren Sie ihn anschliessend in Ihre Abfrage. * Erstellen Sie damit nun die eigentliche Programmdatei, die prüft, ob der Hash der Datei mit demjenigen übereinstimmt, den sie soeben ermittelt haben. |
| **Hinweis Hilfsmittel** | * Erstellen Sie eine neue PHP-Datei und lösen Sie die obigen Aufgaben. |
| **Ergebnis** | * Die Datei ***B14-133-19.php***, welche die Resultate, wie sie oben beschrieben sind, vollständig und korrekt ausgibt. Speichern Sie diese Datei anschliessend in Ihrem persönlichen Verzeichnis. |

# Zeitfunktionen (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie wissen, wie Sie Zeitdaten berechnen und formatieren können. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Datums- und Uhrzeitfunktionen können für zeitgesteuerte Inhalte, Kalender oder Zeit- und Datumsausgabe genutzt werden. Der Timestamp wird von Unix-Systemen verwendet und zählt die Sekunden seit dem 1.1.1970 um 00:00 Uhr. Mit dem daraus entstandenen 10-stelligen Zahlencode lässt sich sehr gut rechnen. |
| **Aufgabe** | * Geben Sie den Satz aus «Heute ist Montag, der 21. März 2018, 14:28 Uhr». Benutzen Sie dazu ***„strftime“.*** * Mit ***„setlocale“*** können Sie den Server dazu bringen, die Wochentage und Monatsnamen in Ihrer gewünschten Sprache auszugeben. * Berechnen Sie Ihr genaues Alter. Die Genauigkeit der Berechnung soll mindestens auf den Tag genau ausgeführt sein. * Erstellen Sie eine Funktion, die den Wochentag jedes beliebigen Datums, in Ihrer Sprache ausgibt. |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-20.php*** und testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-20.php*** anschliessend in Ihrem Verzeichnis ab. |

# Suchen, Finden und Ersetzen (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Suchfunktionen für Strings anwenden. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Viele Funktionen helfen Ihnen dabei, wenn Sie prüfen möchten, ob ein Teilstring in einem String enthalten ist oder wenn Sie in einem String bestimmte Zeichen ersetzen möchten. |
| **Aufgabe** | * Suchen Sie nach dem Teilstring ***bar*** in dem Satz «TOPOMEDICS: Messbar besser». Überprüfen Sie ob der Teilstring vorkommt. Geben Sie bei Erfolg eine Bestätigung mit der genauen Position aus. * Ändern Sie den String:   «TOPOMEDICS: Messbar besser» in  «TOPOMEDICS — Messbar besser» |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-21.php*** und testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-21.php*** anschliessend in Ihrem Verzeichnis ab. |

# Funktionen für Arrays (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie kennen verschiedene Funktionen für Arrays. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Auch für Arrays gibt es viele nützliche Funktionen. Eine Zusammenfassung aller Array Funktionen finden Sie hier: http://php.net/manual/de/ref.array.php. |
| **Aufgabe** | * Wandeln Sie ein Array in einen String um: array('Bern', 'St.Gallen', 'Zürich') * Benutzen Sie dazu die Funktion ***implode()***. Geben Sie den String aus. * Wandeln Sie die einzelnen Array-Einträge in Variablen um und geben Sie die einzelnen Variablen aus. Setzen Sie ein ***und*** zwischen die Array-Einträge. * Durchsuchen Sie das Array nach dem Wort Zürich. Bestätigen Sie wenn das Wort im Array vorkommt. |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-22.php*** und testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-22.php*** anschliessend in Ihrem Verzeichnis ab. |

# Datei- und Verzeichnisfunktion (Teil 1) (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Dateien öffnen, bearbeiten und Verzeichnisse darstellen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 60 Min. |
| **Ausgangslage** | Um Daten dauerhaft zu speichern, können sie in Dateien geschrieben werden. |
| **Aufgabe** | * Erstellen Sie eine Textdatei ***beispiel.txt***. Schreiben Sie drei Zeilen mit Text hinein. Zum Beispiel so:   ***Erste Zeile***  ***Zweite Zeile***  ***Dritte Zeile***   * Erstellen Sie eine Datei ***B14-133-23.php*** mit einem PHP Script, das die Textdatei als Ganzes am Browser ausgibt. * Ändern Sie das Script so ab, dass alle vorhandenen Zeilen untereinander ausgelesen werden. |
| **Hinweis Hilfsmittel** | * Testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-23.php*** anschliessend in Ihrem Verzeichnis ab. |

# Datei- und Verzeichnisfunktion (Teil 2) (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Dateien öffnen, bearbeiten und Verzeichnisse darstellen. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 40 Min. |
| **Ausgangslage** | Um Daten dauerhaft zu speichern, besteht die Möglichkeit, diese in Dateien zu schreiben. |
| **Aufgabe** | * Erstellen Sie ein Script, welches das aktuelle Verzeichnis, in dem die Datei liegt, ausliest. Alle Elemente, die im Verzeichnis liegen, sollen untereinander angezeigt werden * Verändern Sie das Script so, dass die Elemente als anklickbare Links gestaltet sind. * Verändern Sie das Script nun so, dass alle Verzeichnispünktlein (".", "..") nicht mehr erscheinen. |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-24.php*** und testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-24.php*** anschliessend in Ihrem Verzeichnis ab. |

# Cookies & Sessions (Teil 1) (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie wissen, wie man Daten ausserhalb eines Script speichern kann (cookies). |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Mit Cookies hat man genau so die Möglichkeit, wie bei Sessions, bestimmte Daten während einer Folge von Aufrufen einer Website festzuhalten. Allerdings kann man mit Cookies auch Daten über die Sitzung hinaus speichern (z.B. 80 Tage nach der Speicherung), denn Cookies werden als Datei auf dem Computer des Besuchers gespeichert. |
| **Aufgabe** | * Erstellen Sie in der neuen PHP Datei ***B14-133-25.php*** ein HTML-Formular, das aus einem Textfeld und einem Submit-Button besteht. * Das Script soll Ihren Namen, der ins Textfeld geschrieben und abgeschickt wurde, in ein Cookie speichern. * Das Cookie soll den Namen ***text*** haben und 10 Minuten gültig sein. * Zeigen Sie das Formular nur an, wenn kein Cookie existiert. * Bei gültigem Cookie geben Sie eine Begrüssung mit Namen aus. |
| **Hinweis Hilfsmittel** | * Testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-25.php*** anschliessend in Ihrem Verzeichnis ab. |

# Cookies & Sessions (Teil 2) (Kap 9)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie wissen, wie man Daten ausserhalb eines Script speichern kann (cookies). |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Mit Cookies hat man genau so die Möglichkeit, wie bei Sessions, bestimmte Daten während einer Folge von Aufrufen einer Website festzuhalten. Sessions werden auf dem Server gespeichert und helfen Ihnen dabei, Variablen für die gesamte Zeit, die ein User auf Ihrer Internetseite verbringt, zu speichern. |
| **Aufgabe** | * Erstellen Sie die neue PHP Datei ***B14-133-26.php*** und starten Sie die Session. * Sie müssen einen Zugriffszähler einbauen, um zu sehen wie oft innerhalb einer Session auf diese Seite zugegriffen wurde. * Beim ersten Zugriff setzen Sie die Variable ***$\_SESSION['anzahlbesuche']***  auf 1. Bei jedem weiteren Besuch erhöhen Sie den Zähler um 1. * Erstellen Sie einen Link mit einer ***$\_GET*** Variablen ***delete*** (auf diese Seite) um die Session zu löschen. * Der Zugriffszähler soll nur sichtbar sein, wenn die Session nicht gelöscht wurde. * Sonst soll auf der Seite einen Zurück-Link mit [***javascript:history.back***](javascript:history.back)***()*** realisiert werden. |
| **Hinweis Hilfsmittel** | * Testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-26.php*** anschliessend in Ihrem Verzeichnis ab. |

# Reguläre Ausdrücke (Teil 1) (Kap 10)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können reguläre Ausdrücke erstellen und verwenden. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 20 Min. |
| **Ausgangslage** | Mit regulären Ausdrücken werden explizite Eingaben von Benutzern in Formularen auf ihre Korrektheit überprüft. Damit wird überprüft, ob die eingegebene Zeichenkette dem definierten Muster entspricht. Es wird nun ersichtlich, dass einzugebenden Felder innerhalb eines Formulares mit diesen Mustern versehen werden können. Somit wird nicht nur gewährleistet, dass die gewollten Informationen erhalten werden, es entspricht auch dem Sicherheitsaspekt, sich vor diversen Attacken zu schützen. |
| **Aufgabe** | * Benutzen Sie die Funktion ***preg\_match*** um eine in der Schweiz gültige Postleitzahl zu filtern. * Geben Sie einen Hinweis bei gültigem oder ungültigem Format aus. * Benutzen Sie die Funktion ***filter\_var*** um eine in gültige Email Adresse zu filtern. * Geben Sie einen Hinweis bei gültigem oder ungültigem Format aus. |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-27.php*** und testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-27.php*** anschliessend in Ihrem Verzeichnis ab. |

# Reguläre Ausdrücke (Teil 2) (Kap 10)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie stellen Sicherheit mittels regulärer Ausdrücke sicher. |
| **Aufgabentyp** | Einzelaufgabe |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Sie erhalten den Auftrag ein PHP Script zu schreiben, um die Qualität von Passwörtern zu überprüfen. |
| **Aufgabe** | Das Passwort soll auf Qualität überprüft werden:   * Prüfen ob Kleinbuchstaben enthalten sind = Stärke 1 * Prüfen ob Grossbuchstaben enthalten sind = Stärke + 1 * Prüfen ob Zahlen enthalten sind = Stärke + 1 * Prüfen ob Sonderzeichen enthalten sind = Stärke + 1 * Passwort Länge prüfen min. 6 max. 15 Zeichen = Stärke + 1 * Passwort länger als 15 Zeichen = Stärke + 2 * Passwort kürzer als 6 Zeichen = Stärke = 1   Ausgabe von Qualitäts – Kommentar:   * = 'Das Passwort ist sehr schwach.' * = 'Das Passwort ist schwach.' * = 'Das Passwort ist nicht wirklich sicher.' * = 'Das Passwort ist stark.' * = 'Das Passwort ist sehr stark.' * = 'So sollte ein sicheres Passwort aussehen!' * default = 'Bitte ein Passwort eingeben.'; |
| **Hinweis Hilfsmittel** | * Erstellen Sie die Datei ***B14-133-28.php*** und testen Sie die Aufgaben in Ihrem lokalen Webserver. |
| **Ergebnis** | * Wenn das Script das Resultat liefert, das in der Aufgabenstellung gefordert wird, ist die Aufgabe erfüllt. Speichern Sie die Datei ***B14-133-28.php*** anschliessend in Ihrem Verzeichnis ab. |

# Einführung Projekt «Kundenerfassung» (Fallstudie)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können ein interaktives Formulars im Rahmen eines Projektes erstellen. |
| **Aufgabentyp** | Anwendungsaufgabe |
| **Zeitaufwand** | 1. Gemäss den einzelnen Aufgaben. |
| **Ausgangslage** | Die in der medizinischen Industrie tätige TOPOMEDICS AG ist spezialisiert auf Entwicklung, Produktion und Vertrieb von elektronischen Messgeräten. TOPOMEDICS AG ist mit 35% Marktanteil (weltweit ausser USA) das führende Unternehmen im Bereich der optischen Kohärenztomographie und stellt zudem eine ganze Reihe von weiteren hochwertigen, computerunterstützten, medizinischen Diagnose- und Messgeräte her.  TOPOMEDICS AG ist ein global tätiges Unternehmen, welches seinen Hauptsitz in Bern (CH) hat und dort 250 Personen beschäftigt. Zudem betreibt die TOPOMEDICS AG in Zürich ein Forschungslabor (7 Mitarbeitende) und in St. Gallen eine Produktionsfirma mit 35 Mitarbeitenden. Nebst dem Produktionsstandort Schweiz befindet sich ein weiterer Produktionsstandort in Shanghai (China) mit 30 Mitarbeitenden und in Kobe (Japan) mit 50 Mitarbeitenden. In Polen, Spanien, Deutschland und Dänemark hat die Unternehmung kleine Niederlassungen (Verkauf und Service) mit je 5 - 7 Mitarbeitenden. Weltweit erzielt sie einen Jahresumsatz von 230 Millionen Franken und beschäftig insgesamt 292 Personen in der Schweiz und 102 Mitarbeitende im Ausland.  Vor zwei Wochen ist der Verkauf & Marketing-Verantwortliche der TOPOMEDICS AG, Herr Meyer, mit der Bitte an Sie gelangt, die Webseite um eine erweiterte Funktion der Neukundenerfassung zu ergänzen. Ihr Vorgesetzter hat Sie mit dem Projekt beauftragt. In einem ersten Telefongespräch mit Herrn Meyer haben Sie seine Vorstellungen und Wünsche aufgenommen. Zuerst erklärt er Ihnen, die Rahmenbedingungen des Kundenbereichs:   * Der Kunde kann in seinem Bereich alle wichtigen Informationen einsehen. * Der Kunde hat die Möglichkeit, persönliche Informationen zu ergänzen oder zu korrigieren.   Die Anmeldung gewährt Zugang zu:   * Bestellungen über den Onlineshop (B2B) * Handbüchern, Foren und FAQ's * Newsletter und Kontakt.   Seine Wünsche an der Kundenerfassung lauten:   * Neue Kunden können sich einfach registrieren. * Die Komplexität des Passwortes wird überprüft. * Der Kunde wird persönlich mit Namen begrüsst. * Bessere Qualität der Kundendaten. * Der Usernamen sollte eindeutig sein. |
| **Aufgabe** | Erstellen Sie im Rahmen eines Projektes die komplette interaktive Kundenerfassung. Das Formular soll zweiteilig sein. Der erste Teil dient dem Eintragen der persönlichen Informationen. Diese Infos werden an ein zweites Formular gesendet, um da die eigentlichen Login-Daten entgegenzunehmen. |
| **Hinweis Hilfsmittel** | * Lösen Sie schrittweise die weiteren Aufgaben . |
| **Ergebnis** | * Ein komplett durchgeführtes Projekt zur Erstellung einer interaktiven Kundenerfassung. * Die einzelnen Resultate im Detail sind der jeweiligen Aufgabe zu entnehmen. |

# Spezifikation erstellen (Fallstudie)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Ausarbeiten von Vorgaben in eine umsetzbare Spezifikation. |
| **Aufgabentyp** | - |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Sie befinden sich nun in der Phase der Realisation. Vor der eigentlichen Gestaltungs- und Programmierarbeit werden die wichtigen Angaben zur Umsetzung in einer Spezifikation aufgeführt. Viele Angaben können aus dem Konzept übernommen werden und müssen ggf. noch detaillierter ausgearbeitet werden. |
| **Aufgabe** | * Im Aufgabenverzeichnis Datei ***Spezifikation\_Kundenerfassung.docx***. Viele Informationen aus dem Konzept sind darin bereits eingetragen. Ihre Aufgabe ist es, die Spezifikation fertig zu stellen. * Zudem beschreiben Sie die Funktionalität des Kundenerfassungsmoduls in einigen Sätzen so, dass ein Mitarbeiter eines anderen Teams, nach dem durchlesen, eine Vorstellung des Moduls hat. |
| **Hinweis Hilfsmittel** | --- |
| **Ergebnis** | * Die fertiggestellte Spezifikation ***Spezifikation\_Kundenerfassung.docx*** im eigenen Verzeichnis. |

# Testplan erstellen (Kap 11)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können einen Testplan erstellen. |
| **Aufgabentyp** | - |
| **Zeitaufwand** | 30 Min. |
| **Ausgangslage** | Damit nach der Programmierarbeit die nötigen Tests gemacht werden können, müssen Sie einen Testplan erstellen. |
| **Aufgabe** | Erstellen Sie Aufgrund des Projektbeschreibung und der Spezifikationen eine Testtabelle mit den Testsituationen und den dazugehörigen Testfällen.   * Erfassen Sie Testsituationen wie z. B: * Normale Benutzung mit gültigen Eingaben * Normale Benutzung mit Varianten gültiger Eingaben * Normale Benutzung mit ungültigen Eingaben * Unvorhergesehene Benutzung des Formulars * Missbrauch der Applikation * Erstellen Sie eine Testtabelle mit den Einträgen * Testsituation * Testfall * Beschreibung * Erwartetes Resultat. |
| **Hinweis Hilfsmittel** | --- |
| **Ergebnis** | * Es liegt eine Testtabelle mit den Einträgen notwendigen Einträgen vor |

# Kundenerfassung (Teil 1) (Kap 11)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Vorgaben für eine fertige Lösung erstellen. |
| **Aufgabentyp** | Anwendungsaufgabe |
| **Zeitaufwand** | 45 Min. |
| **Ausgangslage** | Nach der ersten Projektsitzung wurde beschlossen, dass das interaktive Formular der Kundenerfassung folgende Attribute erfüllen soll.   * Technologie Serverseitig PHP & MySql * Funktionalität der Web-Applikation mittels Session-Handling * Serverseitige Formularüberprüfung * Starke Verschlüsselung der Passwörter in der Datenbank * Sichere Übergabe der Formulardaten * Schutz gegen Cross-SiteScripting, Skrip-Injection und Session-Hijacking |
| **Aufgabe** | * Eröffnen Sie in ihrem Server das Projekt ***kundenerfassung***. * Erstellen Sie die MySql Datenbank ***anmeldung***. * In der Tabelle ***login*** erstellen Sie die Spalten ***id, username, passwort, email***. * Verschaffen Sie sich einen Überblick über die erhaltenen Unterlagen. Erstellen Sie die erste Formularseite ***anmeldung.php*** und einen Ordner ***inc*** im Projekt-Rootverzeichnis. * Testen Sie die Clientlösung * Lagern Sie den Header und den Footer der Dateien in den inc Ordner aus. * Verschieben Sie allfällige CSS Angaben im Header in die benutzte CSS-Datei (Verzeichnis „css“). |
| **Hinweis Hilfsmittel** | --- |
| **Ergebnis** | * Die erstellte Datei ***anmeldung.php*** im Verzeichnis ***kundenerfassung*** inkl. den nötigen Unterverzeichnissen und deren Dateien. |

# Kundenerfassung (Teil 2) (Kap 11)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Vorgaben für eine fertige Lösung erstellen. |
| **Aufgabentyp** | Anwendungsaufgabe |
| **Zeitaufwand** | 60 Min. |
| **Ausgangslage** | Die Ausgangslage ist dieselbe wie in Aufgabe 32 |
| **Aufgabe** | * Erstellen Sie die zweite Seite der Kundenerfassung. Speichern Sie die Seite im Verzeichnis ***kundenerfassung***. * Nehmen Sie die Formulardaten der ersten Seite entgegen. * Überprüfen Sie die Formularinhalte auf Vollständigkeit. * Überprüfen Sie die Formularinhalte auf unerlaubte Zeichen. * Speicher Sie die Inhalte in Sessions. * Der direkte Zugriff auf diese Seite soll unterbunden werden |
| **Hinweis Hilfsmittel** | --- |
| **Ergebnis** | * Die erstellte Datei ***anmeldung2.php*** liegt Im Verzeichnis ***kundenerfassung*** inkl. den nötigen Unterverzeichnissen und deren Dateien vor. |

# Kundenerfassung (Teil 3) (Kap 12)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Vorgaben für eine fertige Lösung erstellen. |
| **Aufgabentyp** | Anwendungsaufgabe |
| **Zeitaufwand** | 60 Min. |
| **Ausgangslage** | Die Ausgangslage ist dieselbe wie in Aufgabe 32 |
| **Aufgabe** | * Erstellen Sie die dritte Seite der Kundenerfassung. Speichern Sie die Seite im Verzeichnis ***kundenerfassung***. * Nehmen Sie die Formulardaten der ersten und zweiten Seite entgegen. * Überprüfen Sie die Formularinhalte auf Vollständigkeit. * Überprüfen Sie die Formularinhalte auf unerlaubte Zeichen. * Der direkte Zugriff auf diese Seite soll unterbunden werden. * Erstellen Sie ein Email als Bestätigung für den Kunden. * Tragen Sie die relevanten Inhalte in die Datenbank ein. * Bestätigen Sie die Anmeldung dem Kunden auf dieser Seite |
| **Hinweis Hilfsmittel** | --- |
| **Ergebnis** | * Die erstellte Datei ***anmeldung3.php*** liegt Im Verzeichnis ***kundenerfassung*** inkl. den nötigen Unterverzeichnissen und deren Dateien vor. |

# Sicherheitstests durchführen (Kap 12)

|  |  |
| --- | --- |
| **erwünschtes Ergebnis** | Sie können Vorgaben für eine fertige Lösung erstellen. |
| **Aufgabentyp** | - |
| **Zeitaufwand** | 100 Min. |
| **Ausgangslage** | Um die Funktionsfähigkeit und Sicherheit eines Programmes sicherzustellen, testet man dieses nach Abschluss der Programmierarbeit nach Fehlern und Lücken. |
| **Aufgabe** | Testen Sie Die Applikation ***Kundenerfassung*** anhand des Testplanes von Aufgabe **31 Testplan erstellen (Kap 11)**. Füllen Sie die Testtabelle mit den Resultaten der durchgeführten Testfällen aus. |
| **Hinweis Hilfsmittel** | Erfassen Sie Testsituationen wie z. B:   * Normale Benutzung mit gültigen Eingaben. * Normale Benutzung mit varianten gültiger Eingaben. * Normale Benutzung mit ungültigen Eingaben. * Unvorhergesehene Benutzung des Formulars. * Missbrauch der Applikation. |
| **Ergebnis** | * Eine ausgefüllte Testtabelle mit den erhaltetenen Resultaten. |