Projektdokumentation: Webzugriff auf eine Datenbank via PHP

Nico Schottelius
Rodenstraße 12
30826 Garbsen
nico-ihk@schottelius.org

Ausbildungsbetrieb:
Wirtschaftsgenossenschaft deutscher Tierärzte eG
Siemensstraße 14
30827 Garbsen
http://www.wdt.de

01.03.2004 - 30.04.2004

Inhaltsverzeichnis

1 Persönliche Erklärung

Ich versichere durch meine Unterschrift, dass ich die betriebliche Projektarbeit und die dazugehörige Dokumentation selbständig in der vorgegebenen Zeit erarbeitet habe. Ich habe keine anderen als die von mir angegebenen Quellen und Hilfsmittel verwendet.

Ort, Datum Unterschrift des Prüflings

Zur Kenntnis genommen:

Ausbilder/-in

2 Einleitung

2.1 Projektumfeld

Das Projekt fand in der Wirtschaftsgenossenschaft deutscher Tierärzte eG („*WDT*“) am Standort in Garbsen statt. Die WDT liefert Arzneimittel und medizinisches Zubehör für tierärztliche Praxen in ganz Deutschland. Zur Zeit arbeiten circa 170 Mitarbeiter an drei Firmenstandorten, davon sind 14 Aussendienstmitarbeiter („*ADM*“) in der Kundenbetreuung tätig. Ein Großteil der Arbeitsabläufe wird durch das *ERP-System*[1] unterstützt. So erfolgt die gesamte Auftragsbearbeitung, die Kommissionierung, der Versand, der Einkauf sowie die Buchhaltung mit Hilfe dieses Systems. Die Aussendiensmitarbeiter erhalten monatlich einen für sie relevanten, individuellen Auszug aus der Datenbank.

2.2 Ist-Zustand

Zur Zeit werden Informationen zur Planung der Kundenberatung an die Aussendienstmitarbeiter als *Excel-Tabelle*[2] via E-Mail versandt. Aufgrund der Verbindungsart (*HSCSD/GSM*[3], ISDN oder Modem) und der Größe (circa 16 *MiB*[4]) der Datei kommt es zu Problemen bei der Übertragung. Diese Probleme äußern sich wie folgt:

- der Transfer der Datei dauert sehr lange (über 30 Minuten)
- durch eine Zwangstrennung im Funknetz nach 30 Minuten kann die Datei nicht vollständig übertragen werden

Da E-Mail ein *Push-System*[5] ist, ergibt sich zusätzlich das Problem, dass grundsätzlich in dreifacher Hinsicht unnötige Daten übertragen werden können:

1. der ADM benötigt die Informationen z.Z. nicht: komplett unnötiger Versand.
   Eine Filterung nach Anforderung ist mit dem momentanen System schwer zu realisieren und könnte bestenfalls manuell mit Rücksprache geschehen.

[1] ERP-System: Enterprise Resource Planing System

[2] Excel ist ein Produkt von Microsoft. Excel-Tabellen stellen ein properitäres Format zur Speicherung von Tabellen dar

[3] GSM ist die Technik die zum Verbinden der Mobiltelefone verwendet wird, HSCSD ist die Datenübertragung über GSM

[4] KiB, MiB, GiB sind Alternativbezeichnungen zu KB, MB und GB. Bei der Verwendung von den letztern ist nicht klar, ob der Faktor 1000 oder 1024 gemeint ist, 1024Byte entsprechen **immer** 1KiB. So sind z.B. manche 80GB Platten nur 80000000000 Byte gross, was 74,51GiB entspricht.

[5] Als „Push-Systeme“ werden Systeme bezeichnet, die die Informationen ohne Aufforderung des Benutzers zu selbigen senden (neben E-Mail auch z.B. Plakate). Bei „Pull-Sytemen“ hingegen muss der Benutzer interaktiv die Informationen anfordern (z.B. Webseite).

2. selbst wenn sich die Daten nicht geändert haben, wird ein Update versandt
   Dies hat zwei Gründe:
   (a) die regelmässig empfangene E-Mail ist ein Indiz für den ADM, dass das System ordnungsgemäss funktioniert
   (b) zum Zweiten wird keine Versionsverwaltung betrieben, sondern die Excel-Tabellen jedes Mal komplett neu generiert
3. Der ADM benötigt nur einen Teil der Daten, bekommt jedoch immer eine vorgefertigte Version mit großem Umfang (Datenüberfluss)
4. Bei jeder Aktualisierung werden auch die unveränderten Stammdaten transferiert

2.3 Soll-Zustand

Auf den Transfer via E-Mail soll grundsätzlich verzichtet und der Zugriff in ein Pull-System umgewandelt werden. Der Zugriff soll über das Internet auf ein Webinterface geschehen. Die Implementation muss auf dem *LAMP-System*[6] des Providers lauffähig sein. Dazu soll ein möglichst identisches Testsystem intern verfügbar sein. Die Daten müssen aus dem ERP-System in einem für MySQL importfähigen Format bereitgestellt werden. Die entsprechenden Aufgaben sollen gemäß der Qualifizierung der Mitarbeiter delegiert werden:

- das LAMP-System soll durch die Systemintegration bereitgestellt werden
- der Export der Daten aus dem ERP-System soll durch die hauseigenen Programmierer bereitgestellt werden
- die Erstellung des Webzugriffs und der Dokumentation (inklusive Handbuch für Benutzer und Entwickler, Schnittstellendefinitionen und Pflichtenheft) soll vom Prüfling erledigt werden

Ziel ist es, ein verwendbares System zu entwickeln und die Machbarkeit zu beweisen.

3 Planung

3.1 Ablauf des Projektes

Zu Beginn des Projektes wurden die Projektgrundsteine „gelegt“:

[6] LAMP ist die Abkürzung für ein **L**inux, **A**pache, **M**ySQL, **P**HP Kombination, wobei Linux das Betriebssystem, Apache der Webserver, MySQL die Datenbank und PHP die Programmiersprache ist.

1. es wurde ein Zeitplan für die einzelnen Phasen festgelegt, inklusive der Festlegung von Besprechungsterminen

2. es wurden für die einzelnen Teams Schnittstellen definiert, damit diese unabhängig voneinander arbeiten konnten und um eine klare Trennung zwischen den einzelnen Modulen zu schaffen

3.2 Schnittstellendefinition

Durch den sauberen Einsatz von Schnittstellen kann zum Beispiel ein Mitarbeiter die Daten aus dem ERP-System auslesen und sie in einem definierten Format speichern, während ein anderer schon die Umwandlung in die neue Datenbank mit Hilfe von Testdaten programmiert.
Es sollen folgende Schnittstellendefinitionen erzeugt werden:

1. Benutzer/Weboberfläche: Layoutdefiniton

2. Weboberfläche/Datenbank: Zugriffsdefiniton

3. Datenbank/Quelldaten: Importformatdefiniton

3.3 Kompatibilität

Die Skripte müssen ein identisches Verhalten auf dem Testsystem und auf dem Echtsystem vorweisen. Um dies zu gewährleisten sind Modifizierungen, die im Rahmen der Installation von Programmen als normal zu betrachten sind (z.B. Pfad-, Limitierungs- oder Namensmodifikationen), nötig. Das Erscheinungsbild dem Benutzer gegenüber soll soweit wie möglich identisch bleiben. Eine Absprache mit dem *ISP*[7] soll vor Beginn der Entwicklung stattfinden, damit eventuelle Inkompatibilitäten vermieden werden können.
Ein System, das nur In-House funktioniert, wäre für die bestehende Situation keine adäquate Lösung. Da die Kompatibilität ein wichtiger Punkt im Rahmen des Projektes ist, wurde diese in einer eigenen Schnittstelle definiert („Test /Echtsystem“).

3.4 Layout (Benutzerschnittstelle)

Das Layout muss vor Beginn der Entwicklung der Weboberfläche definiert sein und den spezifischen Anforderungen entsprechen. An dieses Layout sollen sämtliche Skripte angepasst sein.

[7] Internet Service Provider bieten Internetanbindungen und Service an

4 Durchführung

4.1 Erzeugen des PHP Such-Skripts

Das PHP Skript wurde nach der Vorlage des Excel Dokumentes erzeugt. Zuerst wurde das Skript monolithisch erzeugt, d.h. es enthielt keinerlei Module und sämtliche Einstellungen waren in dem Skript selbst zu finden. Da sich schnell herausstellte, dass eine solche Struktur schlecht zu erweitern ist, wurde das Skript in mehrere Teile aufgeteilt, welche wiederum modularisiert sind. Die Module wurden in ein Unterverzeichnis „modules“ verschoben. Konfigurationen können nun zentral in den dazu passenden Dateien im „includes“ Verzeichnis vorgenommen werden. Es soll die Möglichkeit geben, dass die Skripte später extern gepflegt oder erweitert werden. Da dies auch durch Dienstleistungen ausländischer Firmen geschehen könnte, sind die Kommentare in Englisch. Da ein Großteil der technischen Literatur in Englisch geschrieben ist, werden deutsche Informatiker mit den englischen Kommentaren keine Probleme haben. Mit einigen Problemen behaftet war der Wechsel von Apache 1.3 zu Apache 2.0. Der Letztere setzt keine globalen Variablen, wenn selbige durch eine Form an ein PHP Skript gesendet werden. Dadurch musste nach dem Wechsel der Inhalt der Formvariablen aus einem speziellen Parameterarray namens „_REQUEST“ ausgelesen werden.

4.2 Erzeugen des Importfilters

Der Importfilter ist dem Layout des Hauptskriptes, und damit der allgemeinen Definition, nachempfunden. Er wurde ebenfalls in PHP geschrieben und importiert die hochgeladenen ASCII Tabellen in die MySQL Datenbank. Das Format der ASCII-Tabellen wurde in der entsprechenden Schnittstellendefinition hinterlegt.
Das Standardverhalten bei identischen Primary Key Feldern ist, den alten Wert mit dem neuen zu überschreiben. Die Alternative, Werte zu addieren ist nur in wenigen Fällen sinnvoll und würde in der praktischen Anwendung zu komplizierten Abhängigkeiten und einigen zusätzlichen Definitionen und Tests führen. So müsste man z.B. Felder wie die Postleitzahl oder die Telefonnummer gesondert behandeln. Ebenfalls problematisch wäre das Ergebnis, wenn jemand aus Versehen zweimal den Upload Knopf betätigt und somit zum Beispiel den Umsatz für einen bestimmten Zeitraum verdoppelt.
Problematisch ist auch das Standardverhalten von PHP, welches nur eine maximale Dateigröße von 2MiB vorsieht und einem Skript nur 8MiB Arbeitsspeicher erlaubt. Wenn die Dateien dieses Limit überschreiten, erzeugt PHP keine Fehlermeldung, sondern ignoriert die hochzuladende Datei (verifiziert bei PHP 4.3.4 und 4.3.5RC3).

4.3 Erzeugen des Initialdatenbankgenerators

Zur Vereinfachung der Arbeit wurde zusätzlich ein Skript eingerichtet, welches eine initiale Datenbank mit den entsprechenden Tabellen anlegt. Die entsprechenden Tabellen wurden in den Quelltext integriert, da eine Auslagerung der Tabellenspezifikation einen zusätzlichen Parser erfordert hätte.

4.4 Transfer in das Echtsystem

Der Transfer in das Echtsystem verlief problemlos. Der Einsatz im Echtsystem wurde mit dem ISP der WDT abgesprochen und getestet. Zuerst wurden die Sicherheitseinstellungen transferiert. Danach wurden die PHP-Skripte hochgeladen, welche dann gleich zur Initialisierung der Datenbank und dem Transfer der Echtdaten benutzt wurden. Da die Verbindung über das *HTTPS*[8] Protokoll erfolgt, sind die Daten während des Transfers verschlüsselt und fälschungssicher. Die maximale Dateigröße für einen Upload durch PHP ist beim ISP auf 8MB beschränkt. Deswegen mussten die Umsatz Tabellen, die pro Jahr eine Größe von 22MiB haben, aufgeteilt werden.

5 Dokumentationsphase

5.1 Erstellen der Programmdokumentation

Die Programmdokumentation ist für die Verwendung durch Entwickler und Administratoren konzipiert und enthält technische Fachbegriffe. Sie wurde in Stichwörtern während der Durchführungsphase niedergeschrieben und danach zusammengefasst in ein Latex-Dokument. *Latex*[9] ist ein Textsatzsystem, welches insbesondere für wissenschaftliche und technische Dokumentation verwendet wird. Die Programmdokumentation ist entsprechend der Bedürfnisse von Administratoren und Entwicklern gegliedert und aufgebaut. Dies bedeutet z.B., dass der Administrator schnell die entsprechenden Sektionen zur Installation und der Entwickler ebenso einfach den Aufbau der Programme finden kann. Als Ausgabeformat wurde PDF gewählt, da dieses Format auf jeder Plattform lesbar ist.

5.2 Erstellen des Benutzerhandbuches

Das Benutzerhandbuch wurde nach der Durchführung erstellt. Dadurch wurde garantiert, dass die Screenshots der Oberfläche sich nicht mehr verändern können. Vom Aufraggeber wurde eine Dokumentation gefordert, die den Ablauf möglichst präzise und zugleich simpel beschreibt. Dies wurde durch den Einsatz von Screenshots bewerkstelligt. Als Ausgabeformat wurde eine

[8] Sicheres, verschlüsselt übertragen von Dateien; Alternative zu HTTP
[9] Siehe auch die deutsche TeX Webseite http://www.dante.de

Präsentation (entweder im *PowerPoint oder OpenOffice*[10] Format) gewählt, da in einer Präsentation wie in einem Buch am Bildschirm geblättert werden kann.

5.3 Erstellen der IHK-Projektdokumentation

Bei der Erstellung dieser Dokumentation wurde ähnlich verfahren wie bei der Erstellung der Programmdokumentation. Jedoch wurde diese Dokumentation zuerst in OpenOffice geschrieben und nachher in Latex konvertiert, da Latex wesentlich effektiver für die Erstellung von großen Dokumenten ist.

6 Testphase

6.1 Testabschnitte während der Erstellung

Während der Entwicklungsphase wurde die Funktionalität der Skripte mit Hilfe einer Datenbank mit Pseudodaten getestet. Da die Anzahl der Sätze in der Datenbank sehr gering war, waren die Ergebnisse schnell verfügbar und es musste in Betracht gezogen werden, dass die Arbeit mit Echtdaten wesentlich langsamer laufen könnte. Dies bestätigte sich jedoch nicht, die Auslieferung der Daten durch die MySQL Datenbank verlief weiterhin gewohnt schnell. Die Ausgabe erfolgte teilweise mit Debug Informationen. Diese beinhalteten z.B. Variableninhalte oder Rückgabewerte von Funktionen.

6.2 Eigentest aller Programme

Nach der Entwicklungsphase wurden die Programme auf Funktionalität und saubere Ausgaben überprüft. So durften keine Debug-Nachrichten mehr erscheinen und es sollten keine Fehler beim „normalen Benutzen“ auftreten. Es wurde kritisch darauf geachtet, dass keine Situation auftritt, die ein normaler Benutzer nicht verstehen kann (z.B. Bildschirm ohne Inhalt). Des weiteren wurden generelle Sicherheitslücken überprüft, wie *Buffer Overflows, SQL-Injektion und Cross-Site-Scripting*[11].

6.3 Fremdtest aller Programme

Zum Abschluß wurden die Programme von Mitarbeitern aus der EDV Abteilung getestet und zusätzlich auf Sicherheitslücken geprüft. Durch diese Hilfe konnten Fehler entdeckt werden, die ein Entwickler selbst nicht bemerkt. So wurde

[10] PowerPoint ist ein Programm aus der Programmgruppe „Microsoft Office“ von Microsoft, OpenOffice ist ein OpenSource Konkurrenzprodukt zu Microsoft Office

[11] Buffer Overflows, SQL-Injektion und Cross-Site-Scripting sind bekannte Fehler die bei der Programmierung auftreten. Die letzteren beiden besonders bei der Programmierung von Skripten mit Webzugriff

z.B. bemerkt, dass auf einer Seite ein Teil des Firmenlogos anders angezeigt wurde („stehende Karawane“, sollte „bewegte Karawane“ sein).

7 Fazit

7.1 Rückblick

Die Programmierung des Skriptes und die Einrichtung des Apache 2.0 mit PHP 4.3.4 aus dem Quelltext verliefen ohne Probleme. Dies wurde durch den Einsatz des LAMP-Systems gefördert, denn die einzelnen Komponenten, Linux, Apache, MySQL und PHP, sind sehr gut aufeinander abgestimmt. Einen Großteil der Zeit musste in die Entwicklung der Schnittstellen und das Testen investiert werden.
Das Teamwork erleichterte die Entwicklungsarbeit, da durch vorhergehende Schulungen und Tipps während der Entwicklung Probleme im Design frühzeitig vermieden werden konnten. So empfahlen Kollegen das Datenbankdesign in die dritte Form der Normalisierung zu übertragen um eine Umsetzung in eine relationale Datenbank zu erleichtern und zeigten an Beispielen, wie dies zu geschehen hat. Die strikte Trennung der einzelnen Aufgaben erleichterte die Arbeit. Dadurch konnte problemlos an einem anderen Teil weitergearbeitet werden, während noch die Echtdaten für die Datenbank fehlten.

7.2 Zeitplan mit Abweichungen

Tabelle 1: Zeitplan

Aufgabe	**benötigte Zeit** (in h)	**geplante Zeit** (in h)
1 Vorarbeiten [Differenz: -1h]	**10**	**11**
1.1 Vorbesprechung mit unserem Außendienst	1	3
1.2 Eruieren und dokumentieren des genauen Problems und Definition des IST-Status	4	3
1.3 Rücksprache mit dem Außendienst und Abgleich des dokumentierten IST-Status mit dem Außendienst	1	1
1.4 Definieren des SOLL-Status	3	3
1.5 Rücksprache mit dem Außendienst und Abgleich des dokumentierten SOLL-Status mit dem Außendienst	1	1
2 Planungsphase [-2h (gesamt: -3h)]	**10**	**12**
2.1 Grob-Modell entwickeln, das die Anforderungen des Soll-Status widerspiegelt	2	2
2.2 Schnittstellen definieren		
2.2.1 Benutzer/Webinterface („Layoutdefinition“)	1	2
2.2.2 Webinterface/Datenbank	2	2
2.2.3 Datenbank/Quelldaten	1	2
2.2.4 Echtsystem/Testsystem	2	2
2.3 Pflichtenheft erstellen	2	2
3 Realisierungs- und Testphase [+2h (gesamt: -1h)]	**28**	**25**
3.1 Erstellen des Webinterfaces		
3.1.1 Erstellen der Zugriffseinstellungen	1	1
3.1.2 Erstellen des Hauptprogrammes inklusive Datenbankverbindung	3	2
3.1.3 Erstellen der Loginprozedur	2	2
3.1.4 Erstellen der Suchprozedur	3	4
3.1.5 Erstellen der Anzeigeprozedur	5	2
3.2 Testen des Webinterface in der Testumgebung	5	5
3.3 Erstellen des Konverters (CSV Datenbank in MySQL-Format)	3	4
3.4 Testen des Konverters in der Testumgebung	2	2
3.5 Fremdtest des Webinterface in der Testumgebung	1	1
3.6 Fremdtest des Konverters in der Testumgebung	1	1
3.7 Transfer des Testsystems in das Echtsystem des Providers	2	2
4 Dokumentationsphase [-5h (gesamt: -6h)]	**12**	**17**
4.1 Erstellen der Programmdokumentation	7	7

4.2 Erstellen des Benutzerhandbuches	5	10
5 Abschlussphase [(gesamt: -6h)]	**1**	**1**
5.1 Vorstellung des Systems in der Außendienstleitung	1	1
6. Pufferzeit für nicht vorhersehbare Ereignisse	**9**	**3**
Gesamtzeit	**70**	**70**

Einige Differenzen erläutere ich detaillierter, da sie wichtige Aspekte der Projektentwicklung widerspiegeln.
Vorarbeiten:

- Die Besprechung mit dem Außendienst verkürzte sich, da diesem die Problematik des alten Systems schon bekannt war

Realisierungs- und Testphase:

- Die Anbindung an die Datenbank verlief nicht so problemlos wie erwartet, da während der Entwicklung der Datenbankserver zwei kurze Ausfälle hatte. Dies führte zu unvermuteten Ergebnissen im Hauptprogramm.
- Die Gestaltung der Anzeigeprozedur erwies sich als wesentlich komplexer als anfangs angenommen, da das Aussehen so wenig wie möglich vom alten System abweichen sollte.

Dokumentationsphase:

- Die Benutzerhandbücher wurden als einfache Präsentation realisiert und bedurften kein spezielles Layout oder des Drucks.

Pufferzeit für nicht vorhersehbare Ereignisse:

- Im Zeitplan wurde der Punkt „Erstellung der Projektdokumentation“ vergessen und musste deswegen hier einsortiert werden.

7.3 Ausblick

7.3.1 Erweiterung der bestehenden Programme

Zusätzlich zu den bestehenden Anzeigen könnten noch Auswertungen programmiert werden, die sowohl schriftlich als auch graphisch präsentiert werden. Des weiteren könnte die Tabellenspezifikation aus init-database.php ausgelagert und selbiges Skript um einen Parser erweitert werden.

7.3.2 Hinzufügen weiterer Programme

Die Verwaltung der Benutzer könnte durch ein Skript vereinfacht werden, welches das Hinzufügen, Löschen oder Modifizieren der MySQL Datenbank und der htpasswd Datei übernimmt. Die Aktualisierung könnte automatisiert werden, wenn von ProAlpha aus regelmässig (z.B. tagesweise) die aktuellen Umsätze an ein Programm übermittelt werden, welches die Daten via HTTPS an das Upload Skript überträgt.

7.4 Kostenrechnung

7.4.1 altes System

Die Kosten vorher setzten sich wie folgt zusammen: 14 Außendienstler holen 15 MiB große Dateien zweimal im Monat ab. Die Internetverbindung erlaubt eine Transferrate von ca. 4 KiB pro Sekunde.
Rechnung:

$$\text{Transferrate} = \frac{4\text{KiB}}{\text{s}}$$

$$15\text{MiB} * 14\text{ADM} = 210\text{MiB} = 210 * 1024\ \text{KiB} = 215040\text{KiB}$$

$$\frac{215040\ \text{KiB}}{\frac{4\ \text{KiB}}{\text{s}}} = 53760\ \text{Sekunden} = 896\ \text{Minuten}$$

$$1\ \text{Minute} \hat{=} 0,39\text{Euro}$$

$$896\ \text{Minuten}\ * \frac{0,39\ \text{Euro}}{\text{Minute}} = 349,44\ \text{Euro pro Update}$$

$2 *$ Monatlich wird das Update gesendet.

$$349,44\ \text{Euro}\ * 2 = 698,88\ \text{Euro pro Monat}$$

Dies ergibt eine Online-Nutzdauer von 53.760 Sekunden, was 896 Minuten entspricht. Bei einem Minutenpreis von 39 Cent pro Minute enstehen **698,88 Euro** monatliche Kosten.

7.4.2 neues System

Die momentane Kosten setzen sich zusammen aus den Kosten für das Hosting des Webinterfaces und den Onlinekosten. Der Provider bietet einen Webauftritt, der über eine HTTPS Verbindung abgesichert ist und eine Unterstützung für PHP und MySQL enthält, für 50 Euro pro Monat an.
Rechnung:

MySQL Datenbank + PHP + Webspace = „Webauftritt“ = 50 Euro pro Monat

$$\text{Nutzhäufigkeit} = \frac{2\text{mal}}{\text{Tag}} \text{ (benutzt der ADM das Webinterface)}$$

Dies macht er an 20 Tagen (Arbeitstage) im Monat.

$$\frac{\text{zweimal}}{\text{Tag}} * \frac{\text{Arbeitstage}}{\text{Monat}} = 40\text{mal Nachsehen pro Monat (pro ADM)}$$

$$14 \text{ ADM} * 40\text{mal Nachsehen} = 560\text{mal}\frac{\text{Nachsehen}}{\text{Monat}}$$

Einmal Nachsehen entspricht statistisch nach einer Umfrage einer Minute (da nicht jeder das Angebot nutzen würde und manche länger brauchen).

$$560\text{Minuten} * \frac{0,39 \text{ Euro}}{\text{Minute}} = 218.40\text{Euro}$$

$$218,40 + 50,00 = 268,40\frac{\text{Euro}}{\text{Monat}}$$

Damit ergeben sich kumuliert Kosten von **268,40 Euro** pro Monat.
In dieser Rechnung sind jedoch nicht die Entwicklungskosten für das neue System enthalten. Diese setzen sich zusammen aus dem Stundenlohn und der Anzahl der benötigten Stunden.
Rechnung:

$$70 \text{ Stunden} * 5 \text{ Euro} = 350 \text{ Euro}$$

Dieser Betrag ist nur einmal fällig und amortisiert sich im Normalfall über die Jahre. In unserem speziellen Fall sind die Entwicklungskosten sogar schon nach einem Monat kompensiert.

7.4.3 Kostenvergleich

Das neue System erbringt eine Kostenvorteil von **430,48 Euro** beziehungsweise verbraucht nur **38%** der alten Kosten pro Monat.

8 Anhang

8.1 Literatur

8.1.1 PHP

- „PHP kurz & gut“, O'Reilly, ISBN 3-89721-225-0
- „Programmieren lernen in PHP4“, Hanser, ISBN 3-446-21754-1
- http://www.php.net

8.1.2 Apache

- http://httpd.apache.org/

8.1.3 HTTP

- HTTP Pocket Reference, O'Reilly, ISBN 1-56592-862-8

8.1.4 MySQL

- Managing and Using MySQL, 2nd Edition, O'Reilly, ISBN 0-596-00211-4

8.2 Schnittstellen

8.2.1 Benutzer/Weboberfläche: Layoutdefiniton

```
------------------------------------------------------------------------------
Nico Schottelius, v0.6
Schnittstelle: Webinterface/Layout
------------------------------------------------------------------------------

1. Generelle Layoutdefiniton
2. Benutzerinteraktion
3. Authentifizierung
4. Sicherheit
5. Einstiegsbildschirm
6. Suchergebnisse
7. Suchdetails


1. Generelle Layoutdefiniton

   Die Weboberfläche sollte von der Farbwahl der Firmenhomepage [WL1] ähneln.
```

Zudem sollte ein firmentypisches Objekt (Logo, Motto, etc.) präsent sein.

2. Benutzerinteraktion

   Der Benutzer soll die Oberfläche intuitiv bedienen können.
   Fachwörter und technische Details sind zu verbergen, Fehlermeldungen wenn
   möglich durch einfache Hinweise zu ersetzen.

3. Authentifizierung

   Der Webserver (hier: Apache) sendet nach dem GET Aufruf des Clients
   einen 401 (Unauthorized) Code und den WWW-Authenticate Header an den Browser. [WI
   Dieser Header kann zusätzlich noch einen Text enthalten, wie z.B.
   "Nur für Außendienstler erlaubt".

   Die Authentifizierung wird dann in einem vom Browser selbstdefinierten
   Authentifizierungsfenster durchgeführt, das den zusätzlichen Text
   des Serverheaders beinhalten kann, jedoch nicht muss.

   Das bedeutet das man Serverseitig nur definieren kann, das sich der Client
   authentifizieren muss, jedoch nicht wie dieses Fenster aussieht (im Gegensatz
   zu Javascript basierten Authentifizierungen, die jedoch keine echte Sicherheit bi

   Die Authentifizierung wird Serverseitig durch die htaccess Methodik [WL3]
   definiert. Diese benötigt die htaccess Einstellungen selbst [WL4] und
   zur Benutzerverwaltung die htpasswd [WL5].

   Des weiteren müssen die in der htpasswd vorhandenen Benutzer noch in der
   MySOL Benutzerverwaltung angelegt werden. Diese erhalten dort ein
   leeres Passwort vergeben, da die htaccess Authentifizierung ausreichend ist.

4. Sicherheit

   Folgende Sicherheitsanforderungen sind vorhanden:

   - Datenintegrität: die Daten müssen gewährleistet unveränderbar sein
   - Vertraulichkeit: niemand darf unauthorisiertes Daten lesen können
   - Verfügbarkeit: die Datenverbindung muss ständig verfügbar sein

   Die ersten beiden Anforderungen werden durch das TLS Protokoll [WL5] erfüllt.

   Die Verfügbarkeit des Webinterface im Internet ist abhängig von der Verfügbarkeit

5. Einstiegsbildschirm

   Die Einstiegsseite zeigt den Namen des aktuell angemeldeten Benutzers an.
   Des weiteren befindet sich ein Auswahlfeld, das die Suchoptionen enthält,
   und des Suchbegrifffeld auf dieser Seite.

   Dem Benutzer muss eine angemessen Suchfunktion zur Verfügung stehen, die
   das Suchen nach den erforderlichen Parametern erlaubt.

   Als Suchoptionen müssen die folgenden vorhanden sein:

   - Kundennummer
   - Kundenname
   - Praxennummer
   - Telefon
   - PLZ
   - Ort

6. Suchergebnisse

   Auf der Seite der Ergebnisse sollen der Suchbegriff und das Suchkriterium
   als Überschrift dargestellt werden.

   Die Suchergebnisse werden tabellarisch dargestellt, die Ergebnisse sind
   geordnet nach Kundennummern, aufsteigend.

   Des weiteren muss die Möglichkeit bestehen die Ergebnisse nach
   den anderen angezeigten Feldern zu sortieren.

7. Suchdetails

   Diese Seite soll sich an dem Beispiel Excel Dokument [WL6] orientieren.
   Sie muss Details über den Kunden enhalten, u.a. die Anschrift
   und soweit vorhanden die Telefonnummer.

[WL1]: http://www.wdt.de
[WL2]: RFC 2616
[WL3]: http://httpd.apache.org/docs-2.0/howto/auth.html
[WL4]: Beispiel: siehe Anhang dot-htaccess
[WL5]: RFC 2246, RFC 3546, Beispiel: htpasswd
[WL6]: Internes Dokument: Beispiel_Excel.xls

8.2.2 Weboberfläche/Datenbank: Zugriffsdefiniton

```
-------------------------------------------------------------------------------
Nico Schottelius, v0.2
Schnittstelle: Webinterface/Datenbank
-------------------------------------------------------------------------------

Tabellen und Felder in der Datenbank
------------------------------------

Die Typen der Felder und Tabellen sind in [WED1] dokumentiert.
Die Namen der Felder und Tabellen sind in [WED2] definiert.
Der Datenbankname und Datenbankserver sind frei wählbar, muessen
jedoch in der Konfigurationsdatei [WED3] angegeben werden.


Quellen:
[WED1]: Siehe Anhang: 2.2.3 Schnittstelle Datenbank/Quelldaten
[WED2]: Siehe Anhang: Quellcode zu modules/init-db-create.php,
                      Funktion create_db, array "$creat_query"
[WED3]: Siehe Anhang: Quellcode zu includes/db-settings.php
```

8.2.3 Datenbank/Quelldaten: Importformatdefiniton

```
-------------------------------------------------------------------------------
Nico Schottelius, v0.6
Schnittstelle: Quelldaten/Datenbank
-------------------------------------------------------------------------------

1. Quelldaten
2. Zielform
3. Ziellayout / Typdefinition
4. Verhalten beim Import

1. Quelldaten

Die Quelldaten liegen in der Progressdatenbank [QDD1]. Sie müssen mithilfe von
Progress oder Alternativ ProAlpha [QDD2] ausgelesen werden.

2. Zielform
```

```
Es sollen CSV Tabellen entsprechend der Typdefinition erstellt werden, wobei pro Tab
Der Trenner soll jedoch kein Komma sein, sondern ein Semikolon.

Tabellenspalten sind getrennt durch Semikolen, Tabellenzeilen getrennt durch
Zeilenumbruch, kein Header (wie z.B. ADM Name; ADM Nr) ist vorhanden,
Reihenfolge wie unten in der Typdefinition.

3. Ziellayout / Typdefinition

      Feldnamen(*)                  genutzte Feldergroessen
      * = Bemerkung vorhanden

   0. ADM-Tabelle
      ADM-Nr.*                      [int]
      ADM-Name.                     [text 255 stellig]

      ADM-Nr. ist >=0 und <=999.
      Nummern >=1000 und <=9999 werden fuer adminstrative Zwecke genutzt.

   1. Kundentabelle
      KundenNr.*                    [int]
      Prax.Nr.*                     [int]
      ADM-Nr.                       [int] [wie oben]
      Name                          [text 255 stellig]
      Strasse                       [text 255 stellig]
      Strassennummer                [text 255 stellig]
      PLZ                           [text 255 stellig]
      Ort                           [text 255 stellig]
      Telefon                       [text 255 stellig]

      KundenNr sind 6 stellig
      PraxNr sind 6 stellig

   2. Umsatztabelle
      KundenNr.                     [int] [wie oben]
      ArtNr.                        [int] [wie unten]

      Umsatz                        [float]
      Menge                         [int]

      Tag                           [int]
      Monat                         [int]
      Jahr*                         [int]
```

```
      Jahr ist vierstellig

   3. Artikeltabelle
      ArtNr.                          [int]
      ArtName                         [text 255 stellig]
      GruppenName                     [text 255 stellig]


4. Verhalten beim Import

Wenn ein Datensatz mit identischen Primary Key vorhanden ist,
so werden die alten Daten in der Zieldatenbank mit den neuen Werten
überschrieben.


Quellen:
[QDD1]: http://www.progress.de/, http://www.progress.com/
[QDD2]: http://www.proalpha.de/
```

8.2.4 Test-/Echtsystem

```
-------------------------------------------------------------------------------
Nico Schottelius, v0.2
Schnittstelle: Test-/Echtsystem
-------------------------------------------------------------------------------

1 generelle Anpassungen
2 Probleme mit anderen PHP oder Apache Versionen oder Konfigurationen


1 generelle Anpassungen

Der Datenbankname und Datenbankserver müssen in der Konfiguration
"includes/db-settings.php" angepasst werden.


2 Probleme mit anderen PHP oder Apache Versionen oder Konfigurationen

Der Apache 2.0 bietet bei PHP keine globalen Variablen.
Der Aufruf von "/script.php?option=test" setzt nur im Apache 1.3.x die Variable
"option". Im Apache 1.3.x und im Apache 2.0 kann man jedoch über den Array
_REQUEST und dem index des Variablennamens (z.B. $_REQUEST['option']) den Inhalt
```

```
auslesen.

Für das Verzeichnis in dem die Skripte liegen muss in der Apache Config
"AllowOverride Auth" angeschaltet sein, damit die .htaccess Datei beachtet wird.

In der Standard Installation von PHP dürfen Dateien beim Upload nicht
größer sein als 2MiB.
Somit muss unter Umständen die Tabellengröße auf 2MiB limitiert werden.
Dies ist jedoch unproblematisch, da die Aktualisierung der Datenbank
inkrementell geschehen kann. Somit ist die Möglichkeit gegeben z.B.
monatlich, wöchentlich, oder sogar täglich eine Aktualisierung vorzunehmen.
Je kleiner die Abstände sind, desto kleiner sind logischerweise auch die Dateien.
Sollte eine Tabelle größer werden als 2MiB, so kann man sie auch teilen und
in mehreren Schritten hochladen.

Das Limit muss mit dem Provider abgeklärt werden.
```

8.3 Quelltext

8.3.1 includes/search.php

```
<?php

/* definiere die suchbegriffe nach feldern in der DB */

   /* suche in welcher tabelle, modular ergaenzbar */
   $suchbegriff_tabelle = array(
            "Kundennummer" => "kunden",
            "Kundenname"   => "kunden",
            "Praxennummer" => "kunden",
            "Telefon"      => "kunden",
            "PLZ"          => "kunden",
            "Ort"          => "kunden"
            );

   /* felder in den tabellen */
   $suchbegriff_felder = array(
            "Kundennummer" => "nr",
            "Praxennummer" => "praxisnr",
            "Kundenname"   => "name",
            "Telefon"      => "telefon",
            "PLZ"          => "plz",
```

```
            "Ort"          => "ort",
            );

   /* felder in den tabellen */
   $suchbegriff_map_field_eqto_kdnr = array(
            "Kundennummer" => "nr",
            "Praxennummer" => "praxisnr",
            "Kundenname"   => "name",
            "Telefon"      => "telefon",
            "PLZ"          => "plz",
            "Ort"          => "ort",
            );


/*
   Ausgabe: wie in excel

   PraxenNummer
   Kundennummer
   Kundenname
   PLZ
   Telefon
   Ort
   WDT-Laufendes Jahr
   Differenz Vorjahr
   PRX-Laufendes Jahr
   Differenz Vorjahr
   GRO-Laufendes Jahr
   Differenz Vorjahr

*/



?>
```

8.3.2 includes/db-settings.php

```
<?php
   $dbserver   = "localhost";
//   $dbserver   = "fs1";
   $database   = "projekt06";
```

```
?>
```

8.3.3 init-database.php

```
<?php
/*********************************************************************
 * Nico Schottelius (c) 2004
 * create the initial database
 *********************************************************************/

   $HEADER='
      <HTML>
      <HEAD>
         <TITLE>Initialisieren der Datenbank</title>
         <META NAME="Author" CONTENT="Nico Schottelius <nico-wdt@schottelius.org>">
      </head>
      <body bgcolor="#ffffff" LINK="#133E70" VLINK="#B52243" TEXT="#000000">
      <P><IMG SRC="images/logo-wdt.png" ALT="WDT" HEIGHT="30" WIDTH="150"></P>
   ';

   $FOOTER='
      <P ALIGN=RIGHT><IMG SRC="images/tiere-animiert.gif"></P>
      </BODY>
      </HTML>
   ';

   /* output header */
   echo $HEADER;

   /* select what todo */
   switch($_REQUEST["option"]) {
      case 1: /* create it */
         include "modules/init-db-create.php";
         create_db($_REQUEST['dbserver'],
                   $_REQUEST['database'],
                   $_REQUEST['dbuser'],
                   $_REQUEST['dbpass']);
      break;

      default: /* login */
         include "modules/init-db-login.php";
         login($dbserver,$database,$_SERVER["PHP_SELF"]);
```

```
      break;
   }

   echo $FOOTER;

?>
```

8.3.4 modules/init-db-login.php

```
<?php
/*********************************************************************
 * Nico Schottelius (c) 2004
 * create the initial database - login
 * v0.4
 *********************************************************************/

/* default function */
function login($dbserver,$database,$self)
{

   include "includes/db-settings.php";

   echo "<h3>Erzeugen der initialen Datenbank</h3>\n";

   /* the upload form */
   echo "<FORM ACTION=\"$self\" METHOD=POST>\n";

   $dbserver_name="Datenbankserver";
   $dbuser_name="Benutzer";
   $dbpass_name="Passwort";
   $database_name="Name der Datenbank";

   /* fields */
   echo "<TABLE>\n";
   echo "<TR><TD>$dbserver_name:</TD>"
   . '<TD><INPUT TYPE="text" name="dbserver" value="' . "$dbserver" . '" size="30">
   echo "<TR><TD>$dbuser_name:</TD>"
   . '<TD><INPUT TYPE="text" name="dbuser" size="30">' . "</TD></TR>\n";
   echo "<TR><TD>$dbpass_name:</TD>"
   . '<TD><INPUT TYPE="password" name="dbpass" size="30">' . "</TD></TR>\n";
   echo "<TR><TD>$database_name:</TD>"
   . '<TD><INPUT TYPE="text" name="database" value="' . "$database" . '" size="30">
   echo "</TABLE>\n";
```

```
   /* hidden */
   echo '<input type="hidden" name="option" value="1">' . "\n";
   echo '<INPUT TYPE="submit" value="Erstellen">' . "\n";
   echo '</form>' . "\n";

}
?>
```

8.3.5 modules/init-db-create.php

```
<?php
/*********************************************************************
 * Nico Schottelius (c) 2004
 * create the initial database - do the create
 * v0.3
 *********************************************************************/

function create_db($server, $database, $user, $pass)
{

   echo "<h3>Initialisere die Datenbank '$database' auf $server als $user</h3>";

   /* open connection */
   $conn = mysql_connect($server,$user,$pass) or
   die("<H3>Verbindung zur Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

   /* choose database */

   /* initial query */
   $query= "CREATE DATABASE IF NOT EXISTS $database;";
   $resultcode = mysql_query($query,$conn) or
   die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");

   echo "<P><B>Datenbank ist angelegt.</B></P>";

   /* select the new db */
   $db_check = mysql_select_db($database,$conn) or
   die("<H3>Selektion der Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

   /* create tables */
   $create_query = array(
```

```
      "CREATE TABLE IF NOT EXISTS adm (
         nr       INT PRIMARY KEY,
         name     TINYTEXT NOT NULL
         );",
      "CREATE TABLE IF NOT EXISTS kunden (
         nr       INT PRIMARY KEY,
         praxisnr INT NOT NULL,
         admnr    INT NOT NULL,
         name     TINYTEXT NOT NULL,
         strasse  TINYTEXT NOT NULL,
         strnr    TINYTEXT NOT NULL,
         plz      TINYTEXT NOT NULL,
         ort      TINYTEXT NOT NULL,
         telefon  TINYTEXT NOT NULL
         );",
      "CREATE TABLE IF NOT EXISTS umsatz (
         kundennr    INT NOT NULL,
         artikelnr   INT NOT NULL,
         umsatz      FLOAT,
         menge       INT,
         tag         INT,
         monat       INT,
         jahr        INT
         );",
      "CREATE TABLE IF NOT EXISTS artikel (
         nr       INT PRIMARY KEY,
         name     TINYTEXT NOT NULL,
         gruppe   TINYTEXT NOT NULL
         );"
   );

   foreach($create_query as $query) {
      echo "<P><B>SQL-Kommando:</B> $query ...\n";
      $resultcode = mysql_query($query,$conn) or
      die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");
      echo "done</p>";
   }

   echo "<P><B>Datenbank und die Tabellen sind angelegt.</B></P>";
}

?>
```

8.3.6 search.php

```
<?php
/* Nico Schottelius (c) 2004  *
 * Suchen in der Datenbank    *
 * v0.4                       */

   $HEADER='
      <HTML>
      <HEAD>
         <TITLE>WDT Datenbankzugriff</title>
         <META NAME="Author" CONTENT="Nico Schottelius <nico-wdt@schottelius.org>">
      </head>
      <body bgcolor="#ffffff" LINK="#133E70" VLINK="#B52243" TEXT="#000000">
      <P><IMG SRC="images/logo-wdt.png" ALT="WDT" HEIGHT="30" WIDTH="150"></P>
   ';

   $FOOTER='
      <P ALIGN=RIGHt><IMG SRC="images/tiere-animiert.gif"></P>
      </BODY>
      </HTML>
   ';


   /* database settings */
   $dbuser = $_SERVER["REMOTE_USER"];
   $dbpass     = "";
   include "includes/db-settings.php";

   /* output header */
   echo $HEADER;

   /* select option */
   switch($_REQUEST['option']) {
      case 1: /* display list of results */
         include "modules/search_results.php";
         search_results($dbserver,$database,$dbuser,$dbpass,$_REQUEST['finde'],$_RE
         break;

      case 2: /* display details */
         include "modules/search_details.php";
         search_details($dbserver,$database,$dbuser,$dbpass,$_REQUEST['kdnr']);
         break;
```

```
      default: /* login */
         include "modules/search_login.php";
         search_login($dbserver,$database,$dbuser,$dbpass,$PHP_SELF);
         break;
   }

   echo $FOOTER;

?>
```

8.3.7 modules/search_results.php

```
<?php
/* (c) 2004 Nico Schottelius
 * display search results
 * code: missing sanity checks */

function search_results($server, $database, $user, $pass, $begriff, $kriterium, $so
{

   include "includes/search.php";

   /* open connection */
   $conn = mysql_connect($server,$user,$pass) or
   die("<H3>Verbindung zur Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

   /* choose database */
   $db_check = mysql_select_db($database,$conn) or
   die("<H3>Selektion der Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

   /* display name of ADM */
   echo "<h3>Suchergebnisse: (gesucht in \"$kriterium\", nach \"$begriff\")</h3>\n"

   /* do the query */
   $table   = $suchbegriff_tabelle[$kriterium];
   $field   = $suchbegriff_felder[$kriterium];

   /* really sort by */
   $rsortby = $suchbegriff_felder[$sortby];

   /*****************************************************/
```

```
/* table begin */
/*****************************************************/
echo "<TABLE BORDER=1>\n";

/* table headers */
$query="DESCRIBE $table;";
$resultcode = mysql_query($query,$conn) or
die("<H3>Query1 fehlgeschlagen: ". mysql_error() ."</h3>");

/* display header */
echo "<TR>\n";

/* create links to sort */
while ($row = mysql_fetch_array($resultcode) ) {
   echo "<TH><A HREF=\"$self?option=1&suchoption=$kriterium&finde=$begriff&sortby
}
echo "</TR>\n";

/* construct the query */
$query_begin="SELECT * FROM `$table` ";
$query_end=" admnr = '$user' ORDER BY ";

$query = $query_begin;

/* sort */
if($sortby != "") $query_end .= " '$sortby'; ";
else $query_end .= " 'nr'; ";

/* search string */
if($begriff == "") $query .= " WHERE ";
else $query .= " WHERE $field LIKE " . "'$begriff" . "%'" .
               " AND ";

/* complete the search */
$query .= $query_end;

$resultcode = mysql_query($query,$conn) or
die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");

/* count if we got results */
$count=0;

/* get number of fields */
$size=mysql_num_fields ($resultcode);
```

```
   /* print all entries */
   while ($row = mysql_fetch_array($resultcode) ) {
      $count++;

      echo "<TR>\n";

      for($i=0;$i<$size;$i++) {
         if($i == 0) {
             $TD="<TD><A HREF=\"$PHP_SELF?option=2&kdnr=$row[$i]\">$row[$i]</A></TD
         } else {
            $TD="<TD>$row[$i]</TD>\n";
         }
         echo "$TD";
      }
      echo "</TR>\n";
   }
   echo "</TABLE>\n";

   /* sanity check */
   if(!$count) {
      echo "<p><B>Keine Ergebnisse gefunden.</B></p>\n";
   }

}

?>
```

8.3.8 modules/seach_details.php

```
<?php
/*****************************************************************
 * (c) 2004 Nico Schottelius
 * display customer details
 *****************************************************************/

function search_details($server, $database, $user, $pass, $nr)
{

   /*****************************************************************/
   /* includes */
   /*****************************************************************/
```

```
include "includes/search.php";

/*****************************************************************/
/* get times */
/*****************************************************************/
$date_info     =  getdate();
$this_year     =  $date_info['year'];
$last_year     =  $this_year -1;
$oldest_year   =  $last_year -1;
$this_month    =  $date_info['mon'];
$this_day      =  $date_info['mday'];
$all_years     =  array("$oldest_year",
                        "$last_year",
                        "$this_year");


/*****************************************************************/
/* sanity checks for variables FIXME! */
/*****************************************************************/


/*****************************************************************/
/* start database connection */
/*****************************************************************/

/* open connection */
$conn = mysql_connect($server,$user,$pass) or
die("<H3>Verbindung zur Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

/* choose database */
$db_check = mysql_select_db($database,$conn) or
die("<H3>Selektion der Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

echo "<h3>Details f&uuml;r Kunde $nr</h3>\n";

/*****************************************************/
/* table begin / headers                             */
/*****************************************************/
$TABLE_BEGIN="<TABLE BORDER=0>\n";
$TABLE_END="</TABLE>\n";

echo $TABLE_BEGIN;

$query=  "SELECT praxisnr , name , strasse , strnr ," .
```

```
          "plz , ort , telefon FROM kunden WHERE nr = $nr " .
          "AND admnr = $user;";

$resultcode = mysql_query($query,$conn) or
die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");

$row  =  mysql_fetch_array($resultcode);
$size =  mysql_num_fields ($resultcode);

echo "<TR><TH ALIGN=LEFT>".$row['name']."</TH></TR>";
echo "<TR><TH ALIGN=LEFT>".$row['strasse']. " " . $row['strnr'] . "</TR>";
echo "<TR><TH ALIGN=LEFT>".$row['plz'].     " " . $row['ort'] . "</TR>";
echo "<TR><TH ALIGN=LEFT>".$row['telefon']."</TH></TR>";
echo "<TR><TH ALIGN=LEFT>".$nr ."</TH></TR>";

echo $TABLE_END;

/*****************************************************************/
/* last 3 months FIXME */
/*****************************************************************/
echo $TABLE_BEGIN;

$last_month = $this_month-1;
$before_last_month = $last_month-1;

/* get data like in .xls */
$query=  "SELECT umsatz FROM umsatz WHERE " .
          "kundennr = $nr " .              /* Kunde */
          "AND jahr  = $this_year " .      /* Jahr  */
          "AND monat = ($this_month OR $last_month OR $before_last_month);" ;

$resultcode = mysql_query($query,$conn) or
die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");

$row  =  mysql_fetch_array($resultcode);
$size =  mysql_num_fields ($resultcode);

echo "<TR><TD>A: $row[0]</td></tr>";
echo "<TR><TD>B: $query</td></tr>";
echo "<TR><TD>C: $row[1] :: $row[2]</td></tr>";

echo $TABLE_END;

/*****************************************************************/
```

```
/* kummultative results */
/*****************************************************************/
echo $TABLE_BEGIN;

/* get data like in .xls */
$query=  "SELECT praxisnr , name , strasse , strnr ," .
         "plz , ort , telefon FROM kunden WHERE nr = $nr " .
         "AND admnr = $user;";

echo $TABLE_END;

/*****************************************************************/
/* display all articles (details) */
/*****************************************************************/

echo $TABLE_BEGIN;

/* get all articles from this customer */
/*
$query=  "SELECT artikelnr FROM umsatz WHERE kundennr = $nr " .
         "GROUP BY artikelnr;";
*/
$query=  "SELECT umsatz.artikelnr FROM umsatz,kunden WHERE " .
         "umsatz.kundennr = $nr AND umsatz.kundennr = kunden.nr " .
         "AND kunden.admnr = $user " .
         "GROUP BY artikelnr;";

$rs_all_artikel = mysql_query($query,$conn) or
die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");

/* display table header */
echo "<TR><TD> </td></tr>\n";
echo "<TR>\n";
echo "\t<TH ALIGN=LEFT>Artikelnummer</TH>\n";
echo "\t<TH ALIGN=LEFT>Umsatz $oldest_year</TH>\n";
echo "\t<TH ALIGN=LEFT>Menge $oldest_year</TH>\n";
echo "\t<TH ALIGN=LEFT>Umsatz $last_year</TH>\n";
echo "\t<TH ALIGN=LEFT>Menge $last_year</TH>\n";
echo "\t<TH ALIGN=LEFT>Umsatz $this_year</TH>\n";
echo "\t<TH ALIGN=LEFT>Menge $this_year</TH>\n";
echo "</TR>\n";

/* count columns */
$count=0;
```

```
/* go through all articles */
while ($row = mysql_fetch_array($rs_all_artikel) ) {
   $count++;
   $artikelnr="$row[0]";
   echo "<TR>\n";
   echo "<TD>$artikelnr</TD>\n";

   /* go through all years */
   foreach($all_years as $work_year) {
      $query =
         "SELECT SUM(umsatz.umsatz) , SUM(umsatz.menge) " .
         "FROM umsatz WHERE umsatz.kundennr = $nr AND " .
         "umsatz.artikelnr = $artikelnr AND umsatz.jahr = $work_year;";

      $rs_cur_artikel = mysql_query($query,$conn) or
      die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");

      $cur_umsatz_menge =  mysql_fetch_array($rs_cur_artikel);
      $cur_umsatz=$cur_umsatz_menge[0];
      $cur_menge=$cur_umsatz_menge[1];

      /* select 0 if nothing was found */
      if($cur_umsatz == "") {
         echo "\t<TD>0</TD>\n";
      } else {
         echo "\t<TD>" . $cur_umsatz . "</TD>\n";
      }

      /* same here */
      if($cur_menge == "") {
         echo "\t<TD>0</TD>\n";
      } else {
         echo "\t<TD>" . $cur_menge . "</TD>\n";
      }

   }
   echo "</TR>\n";
}

echo $TABLE_END;

/* sanity check if there were no articles bought*/
if(!$count) {
```

```
      echo "<p><B>Keine Ergebnisse gefunden.</B></p>\n";
   }
}

?>
```

8.3.9 upload.php

```
<?php
/*********************************************************************
 * Nico Schottelius (c) 2004
 * Update der Datenbank
 *********************************************************************/

   $HEADER='
      <HTML>
      <HEAD>
         <TITLE>Update der Datenbank</title>
         <META NAME="Author" CONTENT="Nico Schottelius <nico-wdt@schottelius.org>">
      </head>
      <body bgcolor="#ffffff" LINK="#133E70" VLINK="#B52243" TEXT="#000000">
      <P><IMG SRC="images/logo-wdt.png" ALT="WDT" HEIGHT="30" WIDTH="150"></P>
   ';

   $FOOTER='
      <P ALIGN=RIGHt><IMG SRC="images/tiere-animiert.gif"></P>
      </BODY>
      </HTML>
   ';


   /* DB settings */
   include "includes/db-settings.php";

   /* database settings */
   $dbuser = $_SERVER["REMOTE_USER"];
   $dbpass     = "";

   /* output header */
   echo $HEADER;

   /* select option */
```

```
   switch($_REQUEST["option"]) {
      case 1: /* do the update */
         $filenames=array( 'adm'     => $_FILES['tabelle_adm'],
                           'kunden'  => $_FILES['tabelle_kunden'],
                           'umsatz'  => $_FILES['tabelle_umsatz'],
                           'artikel' => $_FILES['tabelle_artikel']
                           );
         include "modules/upload-it.php";
         update_it($dbserver,$database,$dbuser,$dbpass,$filenames);
         break;

      default: /* login */
         include "modules/upload-login.php";
         login($dbserver,$database,$dbuser,$dbpass,$PHP_SELF);
         break;
   }

   echo $FOOTER;

?>
```

8.3.10 modules/upload-login.php

```
<?php
/*********************************************************************
 * Nico Schottelius (c) 2004
 * create the initial database - do the create
 * v0.5
 *********************************************************************/

/* default function */
function login($dbserver, $database, $dbuser, $pass, $self)
{

   $adm_tab       = "ADM Tabelle";
   $kunden_tab    = "Kunden Tabelle";
   $umsatz_tab    = "Umsatz Tabelle";
   $artikel_tab   = "Artikel Tabelle";

   /* 120 MB */
   $max_tab_size  = 120 * 1024 * 1024;
```

```
   echo "<H3>Update der Datenbank...</h3>\n";

   /* the upload form */
   echo "<FORM enctype=\"multipart/form-data\" ACTION=\"$self\" METHOD=POST>\n";

   /* fields */
   echo "<TABLE>\n";
   echo "<TR><TD>$adm_tab:</TD>"
   . '<TD><INPUT TYPE="file" name="tabelle_adm" size="30">' . "</TD></TR>\n";
   echo "<TR><TD>$kunden_tab:</TD>"
   . '<TD><INPUT TYPE="file" name="tabelle_kunden" size="30">' . "</TD></TR>\n";
   echo "<TR><TD>$umsatz_tab:</TD>"
   . '<TD><INPUT TYPE="file" name="tabelle_umsatz" size="30">' . "</TD></TR>\n";
   echo "<TR><TD>$artikel_tab:</TD>"
   . '<TD><INPUT TYPE="file" name="tabelle_artikel" size="30">' . "</TD></TR>\n";
   echo "</TABLE>\n";

   /* hidden */
   echo '<input type="hidden" name="MAX_FILE_SIZE" value="' . "$max_tab_size\">\n";
   echo '<input type="hidden" name="option" value="1">' . "\n";
   echo '<INPUT TYPE="submit" value="Update!">' . "\n";
   echo '<INPUT TYPE="reset" value="Zur&uuml;cksetzen">' . "\n";
   echo '</form>' . "\n";

}
?>
```

8.3.11 modules/upload-it.php

```
<?php
/*****************************************************************
 * Nico Schottelius (c) 2004
 * Update der Datenbank
 * v0.3
 *****************************************************************/

function update_it($dbserver, $database, $dbuser, $pass, $filenames)
{

   /* open connection */
   $conn = mysql_connect($dbserver,$dbuser,$pass) or
   die("<H3>Verbindung zur Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");
```

```
   /* choose database */
   $db_check = mysql_select_db($database,$conn) or
   die("<H3>Selektion der Datenbank fehlgeschlagen: ". mysql_error() ."</h3>");

   $i=0;
   foreach($filenames as $tablename => $tableinfo) {

      /* debug */
//      echo "<P>table: $tablename\n";
//      echo "<BR>tableinfo: $tableinfo</P>\n";


      $wert = $tableinfo["tmp_name"];
      $filename = $tableinfo["name"];

      /* check if this table should get updated */
      if($wert != "none" and $wert != "") {

         echo  "<P><B>Update die Tabelle '$tablename' " .
               "aus Datei $filename ...</B>\n";

         /* reset query */
         $query = "";

         /* basic query array */
         $sql = array ('LOAD DATA LOCAL INFILE', " '$wert'", ' REPLACE INTO TABLE '
                       " `$tablename` ", 'FIELDS TERMINATED BY \';\' ENCLOSED BY \'

         /* construct query */
         foreach($sql as $partofquery) {
            $query .= $partofquery;
         }

         /* query() */
         echo "<P>$query </P>";
         $resultcode = mysql_query($query,$conn) or
         die("<H3>Query fehlgeschlagen: ". mysql_error() ."</h3>");
         echo "<B>done</B><br>\n";
         $i++;
      }
   }

   /* result display */
```

```
   echo "<p>$i Tabelle(n) verarbeitet.</p>\n";
}

?>
```

8.4 Testtabellen

Diese Tabellen wurden zum Testen benutzt, während noch keine Echtdaten verfügbar waren. Sie werden von upload.php verarbeitet (siehe Seite 36).

8.4.1 Aussendienstmitarbeiter („adm.test“)

```
120; Heinz Martin
140; Mutter Albert
150; Kruenling
1000; Der Admin
```

8.4.2 Artikel („artikel.test“)

```
12345; Elefantenpritze; wdt
12346; Elefantenbesteck; wdt
12347; Kaenguruhnapf; prx
```

8.4.3 Kunden („kunden.test“)

```
019066; 201749; 120;prakt. Tierärztin aus Test;Testwegrein;23;22527;Hamburg;01234/45
010646; 201749; 120;prakt. Tieraerztin2 aus Test3;Testwegrein;23;22527;Hamburg;0123
019067; 019066; 140; Tierliebhaber Burgberg; veilenweg; 42; 30456; Hannover; 00123/
019068; 019069; 150; Tierer Bargberg;blumenstarre;235;33334; Testhause; 0042455/553
```

8.4.4 Umsatz („umsatz.test“)

```
019066;12345;120,00;12; 01; 07; 2004
019066;12346;120,00;12; 02; 06; 2004
019066;12347;120,00;12; 03; 05; 2004
019066;12347;150,00;12; 19; 04; 2003
019066;12347;140,00;12; 19; 03; 2003
```

```
019066;12346;110,00;12; 19; 02; 2002
019066;12345;100,00;12; 10; 01; 2002
019066;12346;150,00;12; 12; 01; 2004
019067;12347;160,00;12; 12; 01; 2004
019068;12346;160,00;12; 22; 01; 2004
```

8.5 Konfigurationsdateien

8.5.1 Apache: .htaccess (oder „dot-htaccess“)

.htaccess ist die Standardmethode für Authentfizierung beim Apache Webserver. In der gleichnamigen Datei wird definiert, um was für eine Art Authentifizierung es sich handelt („AuthType“), welcher Text optional angezeigt wird beim Darstellen der Passwortbox („AuthName“), woraus die Authentifizierungsinformationen gelesen werden („AuthUserFile“) und welche Bedingung erfüllt sein muss („Require“).

```
AuthType Basic
AuthName "Willkommen zum WDT Aussendienstler Programm"
AuthUserFile /home/user/nico/www/projekt/source/testpasswd
Require valid-user
```

8.5.2 Apache: htpasswd

In der htpasswd werden die Passwörter für die Authentifizierung über das Webinterface definiert.

```
120:8ZnzPTAQ5IJkM
9999:$apr1$/9xFh...$/Qy28iJzLaPWfhFWwsy1C/
1000:$apr1$shOFt...$xoRiz8F4lClo2hv/MGi7n.
```

8.5.3 PHP: Auszug php.ini

Im Gegensatz zur normalen Konfiguration ist hier die maximale Dateigröße 120MB erhöht. Es ist der *Manpage* nicht zu entnehmen, ob MB hier MiB entspricht oder der Faktor 1000 als Basis genommen wurde.

```
;;;;;;;;;;;;;;;;;;;
; Resource Limits ;
;;;;;;;;;;;;;;;;;;;
```

```
max_execution_time = 30     ; Maximum execution time of each script, in seconds
max_input_time = 60 ; Maximum amount of time each script may spend parsing request d
memory_limit = 128M      ; Maximum amount of memory a script may consume (8MB)
;;memory_limit = 8M      ; Maximum amount of memory a script may consume (8MB)

; Maximum size of POST data that PHP will accept.
;;post_max_size = 8M
post_max_size = 120M

;;;;;;;;;;;;;;;;
; File Uploads ;
;;;;;;;;;;;;;;;;

; Whether to allow HTTP file uploads.
file_uploads = On

; Temporary directory for HTTP uploaded files (will use system default if not
; specified).
;upload_tmp_dir =

; Maximum allowed size for uploaded files.
;;upload_max_filesize = 2M
upload_max_filesize = 120M
```