



Protokoll zum Projekt:

KöTaf

Antonios Fesenmeier

Björn Bittner

Florian Wasielewski

Georg Schmid

Lucas Kögl

Michael Müller

Dietmar Sach

Patrick Vogt

Stand: 22. Juli 2013

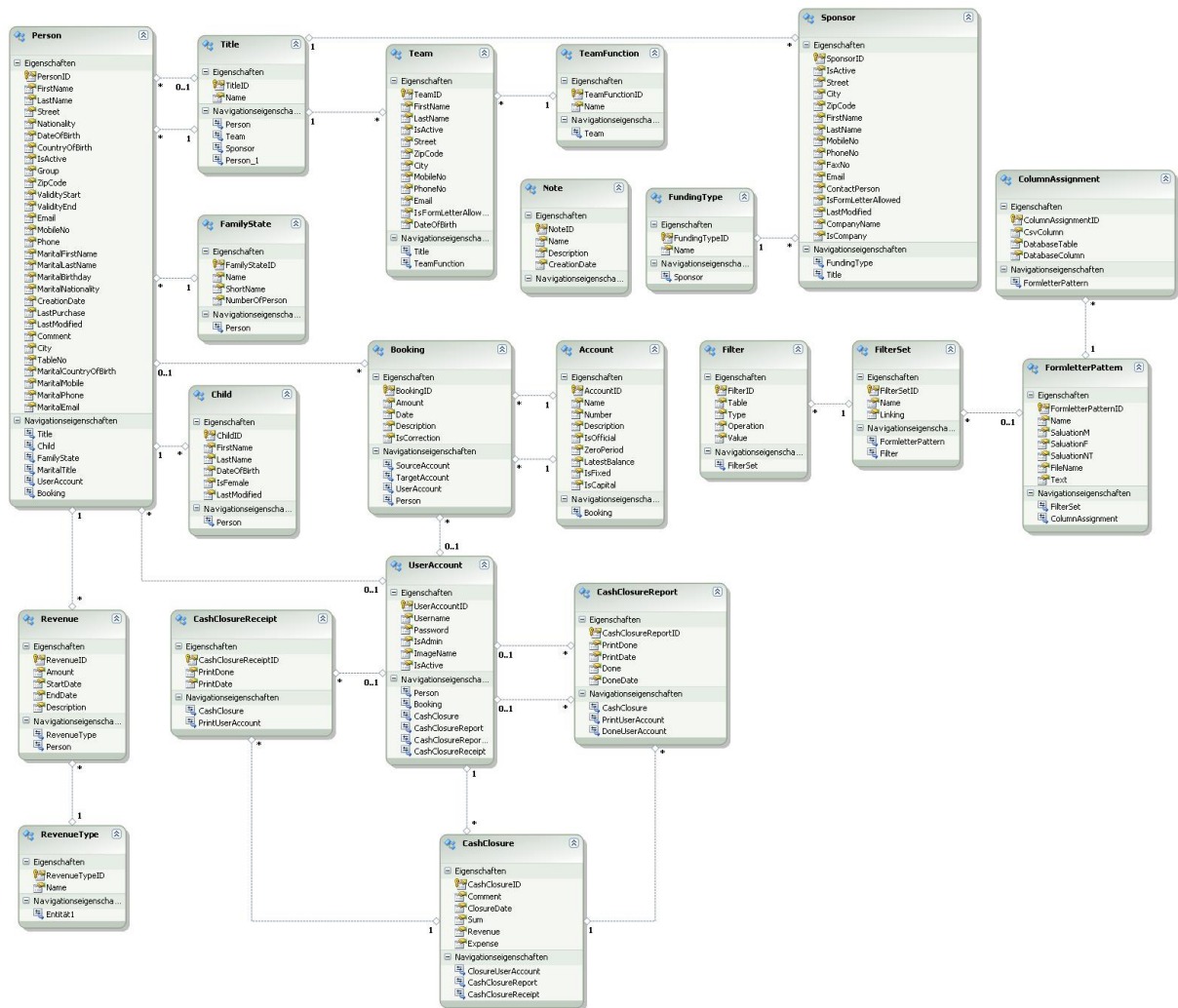
Inhalt

Datenmodell.....	4
Entity Framework Datenbank Modell	4
Klassendiagramm.....	4
Verzeichnisstruktur	5
Installation.....	6
Funktionsweise.....	6
Programmeinstieg:	6
Beenden des Programms:.....	7
Buchhaltung	7
Paging	7
Bedienung	8
Serienbriefe	8
Erstellen eines Serienbriefs	8
Serienbrief-Verwaltung	9
Buchhaltung	14
Schnellbuchung.....	14
Buchungen.....	14
Kontensummen.....	16
Kassenabschluss	17
Benutzerverwaltung.....	19
Übersicht	19
Neuer Benutzer	20
Wiederherstellung einer Sicherungsdatei	21
Kontenverwaltung	21
Übersicht	21
Konto anlegen	22
Druck.....	23
Notizen	23
Verwaltung	24
Backup der Datenbank	24
Allgemein	24
Details	24
Die Konfigurationsdatei.....	25
[APPSETTINGS]	25
[RESTORE_TIMER].....	25
[USB].....	25
[PAGING]	25
[VALIDATION]	25

Anlegen und Editieren von Kunden.....	26
Personalien	26
Partner & Kinder	27
Einkommen	28
Speichern der Daten	28
Editieren der Datensätze	28
Anlegen und Editieren von Sponsoren	29
Neuer Sponsor	29
Editieren von Sponsoren.....	30
Anlegen und Editieren von Teammitgliedern	30
Anlegen von Teammitgliedern.....	30
Editieren von Teammitgliedern	31
Listen – Klassen –Hierarchie.....	31
Import der Bestandsdaten	31
Arbeitsaufteilung	32
Florian Wasielewski.....	32
Projekt KöTaf.DataModel.....	32
Projekt KöTaf.WPFApplication.....	32
Projekt KöTaf.Utils.....	32
Sonstiges.....	32
Michael Müller.....	32
Projekt KöTaf.WPFApplication.....	32
Projekt KöTaf.Utils.....	32
Sonstiges.....	33
Antonios Fesenmeier.....	33
Projekt KöTaf.DataModel.....	33
Projekt KöTaf.WPFApplication.....	33
Projekt KöTaf.Utils.....	34
Sonstiges.....	34
Georg Schmid	34
Projekt KöTaf.DataModel.....	34
Projekt KöTaf.WPFApplication.....	34
Sonstiges.....	34
Dietmar Sach.....	34
Projekt KöTaf.Utils.....	34
Projekt KöTaf.WPFApplication.....	34
Lucas Kögel	35
Projekt KöTaf.WPFApplication.....	35
Sonstiges.....	35

Björn Bittner	36
Projekt KöTaf.WPFApplication.....	36
Sonstiges.....	36
Patrick Vogt.....	36
Wartungshinweise	37
Modul Druck	37
ToDo-Liste.....	37
Sortierfunktion in den DataGrids	37
Serienbriefe	39
Notizen	39
Kundenmodul	39
Backup	39
Globale Stile	39
Buchhaltung	39
Login-Fenster.....	40

Datenmodell



Entity Framework Datenbank Modell

Um die einzelnen Tabellen im Detail zu betrachten, klickt man auf die *.edmx Datei (Projekt: KöTaf.DataModel) mit der rechten Maustaste und wählt „Öffnen mit“ → „ADO.NET Entity Data Model Designer“

Näheres zum EntityFramework bitte dem Dokument [EntityFramework.docx](#), im selben Verzeichnis, entnehmen.

Klassendiagramm

Die Klassendiagramme befinden sich im Ordner /_DOKUMENTATION/CLASS_DIAGRAM, und sind dort als Bilddatei verfügbar.

Verzeichnisstruktur

KöTaf.DataModel: Enthält die zu verwendende .sdf Datenbankdatei

- _SQL
 - o _DATABASE_MODEL: *Datenbankmodell*
 - o _SCRIPTS: SQL-Statements für Testdaten, Fehlerbehebung
- Enums: Datenbankspezifische Enums
- Partial Implementations: Enthält Klassen für Zugriff auf die Datenbank

KöTaf.Utills

- _ExternalDLL: Enthält benötigte .dll-Dateien
- BackupRestoreDB: Klassen für das Backup-Modul
- Database: Datenbank-Tools
- FileOperations: *Dateisystem-Tools*
- Parser: INI/Registry/String-Parser
- Printer: Druck über LibreOffice Calc
- UserSession: Klasse für den aktuelle Benutzer
- ValidationTools: Klasse mit Funktionen zur Validierung von Feldern

LibreOffice: Enthält den VBA-Code, der für vorheriges Printer-Modul in Calc erforderlich ist

Kötaf.WPFApplication

- Im Verzeichnis:
 - o App.xaml: Globale Stile
 - o config.ini: Konfigurationsdatei
 - o MainWindow.xaml: *Hauptseite*
- Converter: Hilfsklassen für DataGrids
- Documents: Office-Dokumente für den Druck und Serienbriefe
- ExternalDLL: Enthält benötigte .dll-Dateien
- Fonts: Enthält Schriftarten
- Helper: Enthält Hilfsklassen für komplexere Aufgaben für verschiedene Module
- Images: Bilder für Programm und Benutzer
- Models: Datenmodelle, die von Controls zur Repräsentation verwendet werden
- ResourceDictionaries: Enthält Style-Definitionen im XAML-Format
- Template: Enthält die für das Design, Toolbar verantwortlichen Klassen
- ViewModels: Enthält Modellklasse für den Login

- Views: Enthält anzuzeigende Programmseiten mit zugehörigem Code
 - o Im Verzeichnis:
 - CloseProgram: Klasse zum Beenden des Programms
 - KPage: Vorlage für eine Programmseite
 - LoginWindow: *Login-Fenster*
 - Print: *Druck*
 - pWelcomeScreen: Willkommenseite mit Lizenz
 - Restore: Backup-Wiederherstellung
 - SplashScreen: Anfangsbildschirm
 - USB_Identification: Seite für das Backup-Modul
 - o Accounting: *Buchhaltung*
 - o Client: Kundenverwaltung
 - o Formletter: Serienbrief-Verwaltung und –Druck
 - o Lists: *Listen*
 - o Note: *Notizen*
 - o Sponsor: Sponsorenverwaltung
 - o Statistics: *Statistiken*
 - o Team: Teamverwaltung
 - o User: Benutzerverwaltung

Installation

Folgende Software muss zum erfolgreichen Start der KöTaf-Software installiert sein:

- SQLCE4.0 (<http://www.microsoft.com/de-de/download/details.aspx?id=17876>)
- LibreOffice
 - o Vorhandenes VBA Makro einfügen (/src/LibreOffice/KÖTAF_VBA_CODE.txt)

Zuweisung in LibreOffice unter den Punkt „Makros verwalten -> zuweisen -> Ereignisse -> Dokument aktivieren“.
- Falls die Datenbank komplett leer sein sollte, so bitte die Datei _SQL/BaseData.sql ausführen. Diese Datei enthält Basis Daten die für den Start der Applikation vonnöten sind

Funktionsweise

Programmeinstieg:

- Die App.xaml.cs ruft das Fenster SplashScreen() auf.

- Nachdem die Daten von der Datenbank geladen, und Benutzer überprüft wurden, wird als nächstes das Login Fenster(LoginWindow) geladen.
- Das Login Fenster besteht aus einem WPF Steuerelement welches zuständig für die Login Daten und das Anzeigen eines Benutzerbildes ist.
- Wurde das Login Fenster erfolgreich bestätigt, wird in der Klasse: Models/LoginViewModel.cs durch die Funktion: ExecuteSubmit() das MainWindow, also die Oberfläche des Kötaf Programms geladen.
- Falls nicht wird der Benutzer erneut gebeten sich anzumelden.

Beenden des Programms:

- Das Programm kann auf zwei Arten beendet / verlassen werden.
- Die erste Methode im angemeldeten Zustand(rechts oben)
- Die zweite Methode im Anmeldefenster
- In beiden Fällen wird die das Fenster /Views/CloseProgram aufgerufen, welche wiederum über die Klasse USB_Identification die Notwendigkeit eines Backups prüft. Anschließend wird das Programm komplett geschlossen/beendet.

Buchhaltung

Das neue System arbeitet mit doppelter Buchführung. Dadurch ist jede Buchung grundsätzlich positiv und besitzt ein Quell- und Zielkonto. Bei einer Betrachtung aller Konten würden sich Einnahmen und Ausgaben gegenseitig eliminieren und ein Saldo von 0,- würde entstehen.

Daher wird unterschieden, ob ein Konto ein offizielles Konto für Eigenkapital ist oder nicht. Konten für Eigenkapital sind alle Konten, bei denen sinnbildlich gesprochen das Geld „im Haus“ bleibt. Mögliche Beispiele sind dafür die Bargeldkasse, ein Bargeldtresor oder die Kaffeekasse.

Alle Konten, die als „Ein- oder Ausfluss“ für Geld dienen, sind keine Eigenkapitalkonten. Beispielsweise wären dies der Kunde, Benzingeld, Spenden, Mitarbeiterlohn, usw.

Paging

Um die Datensätze der verschiedenen Module, zum Beispiel der Buchhaltung oder der Kundenverwaltung, optisch ansprechend darzustellen wird nur ein Ausschnitt der Gesamtdaten angezeigt. Die einzelnen Ausschnitte können dann über Buttons „durchgeblättert“ werden.



Um die Schrittweite des Paging anzupassen kann der Wert des Labels STEPSIZE=12 im Abschnitt [PAGING] der Konfigurationsdatei verändert werden.

Die Buttons werden in der Klasse PagingBar.cs im Verzeichnis KöTaf.WPFApplication/Template erstellt. Die Logik des Paging wurde in der Klasse DataGridPaging.cs im Verzeichnis KöTaf.WPFApplication/Helper realisiert.

Bedienung

Serienbriefe

Erstellen eines Serienbriefs

Das erstmalige Erstellen eines Serienbriefs erfordert ein wenig Handarbeit. Vor dem Anlegen einer Serienbrief-Vorlage im Programm muss ein Serienbrief mit dem Textverarbeitungsprogramm (z.B. LibreOffice) erzeugt werden.

Zunächst wird ein neuer Serienbrief (z.B. meinSerienbrief.odt) angelegt. Diese kann nun nach Belieben formatiert werden, der Briefftext muss hierbei ebenfalls festgelegt werden.

Nun, da ersichtlich ist, an welche Position im Brief welches Feld kommt (z.B. Vorname, Name, Adresse, usw. im Briefkopf), muss eine CSV-Datei erstellt werden.

Die CSV-Datei muss den gleichen Dateinamen haben wie der Serienbrief (z.B. meinSerienbrief.csv) und muss abgelegt werden in *Anwendungsverzeichnis/documents/serienbrief_vorlagen*.

In der CSV-Datei müssen alle Attribute wie folgt per Semikolon getrennt definiert sein:

Adresse;Anrede;Briefftext;Name;Ort;PLZ;Titel;Vorname

Es reicht die Angabe dieser einen Zeile, die Empfängerdaten werden beim Druck vom Programm generiert. Diese Angabe der Attribute definiert die Feldnamen, die in der Abbildung zum Serienbrief-Verwaltungs-Tab *Dateiverknüpfungen* links angezeigt werden.

Nachdem die CSV-Datei angelegt ist, kann zum Textverarbeitungsprogramm zurück gewechselt werden. Die folgenden Schritte beziehen sich auf LibreOffice 4.0.4.2.

Es muss eine neue Datenbank mittels Datei -> neu -> Datenbank angelegt werden. Hier muss Verbindung zu einer bestehenden Datenbank herstellen -> Text ausgewählt werden.

Im nächsten Schritt muss das Verzeichnis

Anwendungsverzeichnis/documents/serienbrief_vorlagen angegeben werden. Folgende Einstellungen sind erforderlich:

- ‚Comma separated value‘ Dateien (*.csv)
- Feldtrenner: ; (Semikolon)
- Texttrenner: “ (doppeltes Anführungszeichen)
- Dezimaltrennzeichen: , (Komma)
- Tausendertrennzeichen: . (Punkt)

Im nächsten Schritt wird der Speicherort für die Datenbank abgefragt. Hier sind wieder der gleiche Dateiname und das gleiche Verzeichnis wie bei der .odt/.csv-Datei auszuwählen. Die Datenbank muss danach nicht bearbeitet werden, falls sich dazu ein Fenster öffnet, kann dies geschlossen werden.

Nun muss die Verknüpfung zu den Feldern der Datenquelle vorgenommen werden. Dazu werden die Datenquellen mit *Ansicht -> Datenquellen* eingeblendet. Es muss die zuvor angelegte Datenbank ausgewählt werden, danach der Punkt *Tabellen* und danach die Bezeichnung der korrekten CSV-Datei.

Nun sieht man rechts davon eine leere Tabelle. Die Spaltenüberschriften der Tabelle entsprechen den in der CSV-Datei eingetragenen Attribut-/Spaltennamen. Per Drag&Drop kann die Spaltenbeschriftung direkt an die entsprechende Position ins Dokument gezogen werden, um eine Verknüpfung anzulegen.

Nach diesen Schritten ist das Erstellen des Serienbriefs abgeschlossen. Nun kann im Programm eine oder mehrere Serienbrief-Vorlagen erstellt werden, die diesen Serienbrief ausfüllen.

Serienbrief-Verwaltung

Serienbrief-Verwaltung Serienbrief-Verwaltung Backup-Wiederherstellen

Neue Serienbriefvorlage

Vorhandene Serienbrief-Vorlagen:

- Alle Kunden zwischen 20 und 30 Jahren mit Wohnsitz in Augsburg oder Bobingen ✕
- Alle aktiven Mitarbeiter ✕

In der Serienbrief-Verwaltung können Serienbrief-Vorlagen erstellt und gelöscht werden. Mit Serienbrief-Vorlagen können Serienbriefe anhand von fest definierten Regeln jederzeit neu erzeugt werden. Es können mehrere Serienbriefe auf dasselbe Serienbrief-Dokument zugreifen, um eine Vielzahl von möglichen Kombinationen von Serienbriefen zu erzeugen.

Neue Serienbrief-Vorlage

Tab Empfänger

In dieser Sektion können Empfängergruppen definiert werden. Diese bestehen aus Filter-Sets, welche wiederum aus einer beliebigen Anzahl Filtern bestehen. Filter-Sets kann man also als Empfänger-Gruppen und Filter als deren Eigenschaften betrachten. Zur Aggregation der Empfänger aus den einzelnen Filter-Sets wird am Schluss die Schnittmenge aller Filter-Sets gebildet.

Beim Anlegen eines neuen Filter-Sets ist im Vorab zu entscheiden, ob die Filter darin mit UND-Logik oder mit ODER-Logik verknüpft werden sollen (z.B.: Wohnort: Augsburg UND/ODER Bobingen).

Neues Filter-Set

Ist ein neues Filter-Set ausgewählt, können zu diesem Filter-Set eine beliebige Anzahl an Filtern hinzugefügt werden. Dabei gibt es die Auswahl zwischen Kunde, Team und Sponsor (Feld Gruppe). Zu diesen Gruppen können bestimmte Eigenschaften zum Vergleich ausgewählt werden, so beim Kunden beispielsweise Vorname, Nachname, Alter, Wohnort, usw. Diese Attribute werden mittels passender Operatoren mit dem eingegebenen Wert verglichen.

Neuer Filter

Gruppe	Kriterium	Operation	Wert	
<input type="button" value="Kunde"/>	<input type="button" value="Vorname"/> <input type="button" value="Vorname"/> <input type="button" value="Nachname"/>	<input type="button" value="gleich"/>	<input type="text" value="Dietmar"/>	<input type="button" value="Filter hinzufügen"/>

Alle Datentypen können mit den Operatoren gleich/ungleich/größer/kleiner verglichen werden. Bei Zeichenfolgen wird ein alphabetischer Vergleich durchgeführt. Bei Wahrheitswerten (z.B. Aktiv: ja/nein) werden nur die Operationen gleich/ungleich angeboten, der Operand muss textuell definiert werden (z.B.: ja/y/true, nein/n/false).

Sind mehrere Filter hinzugefügt, ergibt sich eine UND- oder ODER-verknüpfte Kombination. Die Verknüpfungsart ist wie abgebildet angegeben. In diesem Beispiel werden alle Kunden im Alter von 18-25 durch dieses Filter-Set als Empfänger deklariert.

Vorhandene Filter

Gruppe: Kunde / Kriterium: Alter / Operation: größer / Wert: 18
Gruppe: Kunde / Kriterium: Alter / Operation: kleiner / Wert: 25

Verknüpfung: UND

ausgewählte löschen

Filter-Set speichern

Sind alle gewünschten Filter hinzugefügt, kann das Filter-Set mittels Button gespeichert werden.

Neue Filter-Sets erscheinen in der Liste der vorhandenen Filter-Sets auf der rechten Seite. Hier können existierende Filter-Sets gelöscht oder ausgewählt werden.

Vorhandene Filter-Sets

Verknüpfung: UND, Filter: 2
Verknüpfung: ODER, Filter: 2

Bei der Auswahl eines Filter-Sets werden die Filter dieses Filter-Sets zur Übersicht auf der linken Seite wieder angezeigt.

In diesem Beispiel sind zwei Filter definiert:

UND-Verknüpfung:

- Kunde / Alter größer 18
- Kunde / Alter kleiner 25

ODER-Verknüpfung:

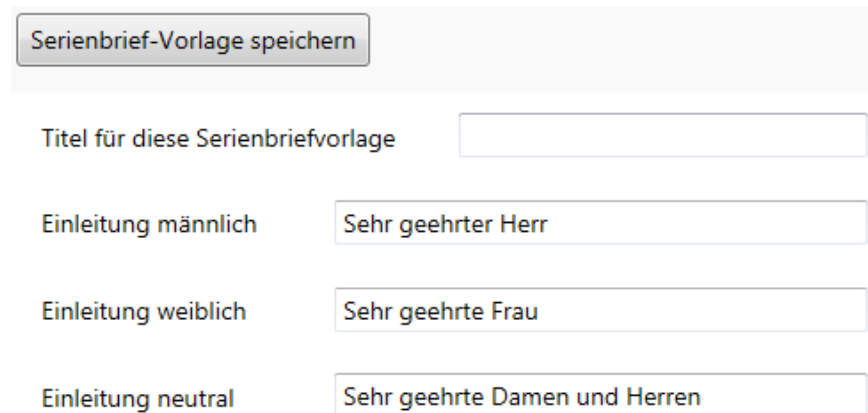
- Kunde / Wohnort gleich Augsburg
- Kunde / Wohnort gleich Bobingen

Aus dem ersten Filter-Set ergeben sich alle Kunden im Alter zwischen 18 und 25. Aus dem zweiten Filter-Set ergeben sich alle Kunden aus Augsburg oder Bobingen.

Schlussendlich wird aus den vorhandenen Filter-Sets die Schnittmenge gebildet, was folgender Aussage entspricht:

Alle Kunden im Alter zwischen 18 *und* 15 *welche* als Wohnort Augsburg *oder* Bobingen haben.

Tab Textfelder



Serienbrief-Vorlage speichern	
Titel für diese Serienbriefvorlage	<input type="text"/>
Einleitung männlich	<input type="text" value="Sehr geehrter Herr"/>
Einleitung weiblich	<input type="text" value="Sehr geehrte Frau"/>
Einleitung neutral	<input type="text" value="Sehr geehrte Damen und Herren"/>

In der Sektion Textfelder wird der Titel für diese Serienbriefvorlage und die Einleitungen angegeben.

Als Einleitung ist in diesem Modul die Zeile in einem Brief zwischen Betreff und Briefftext gemeint.

Da beim Anlegen einer Person das Geschlecht angegeben werden muss, kann im Serienbrief automatisch die passende Einleitung generiert werden. Mit diesen Textfeldern hat man die Möglichkeit, die Einleitung anzupassen. So kann beispielsweise bei einem Schreiben, das gezielt Kinder adressieren soll, als Einleitung „Lieber“, „Liebe“, und „Liebe(r)“ als persönlichere Anrede gewählt werden.

Tab Dateiverknüpfung

Im Textverarbeitungsprogramm (z.B. LibreOffice Writer) müssen in einem existierenden Serienbrief die Zuordnungen zu den Attributen der Datenquelle (Spalten der CSV-Datei) bereits existieren. Dabei entsprechen die Namen der verknüpften Attribute den Attributen der Datenquelle (Spalten der CSV-Datei).

Auf dieser Seite muss also eine existierende .odt Serienbriefvorlage ausgewählt werden.

Dieses LibreOffice-Dokument im OpenDocument-Format wird auf vorhandene Serienbrief-Feldverknüpfungen geparkt, das Ergebnis wird als Liste dargestellt:

Bitte wählen Sie die OpenOffice-Datei aus, der diese Vorlage zugeordnet werden soll

uments\serienbrief_vorlagen\firstFormletter.odt

Durchsuchen

C:\Users\Dietmar\Desktop\KöTaf\src\KöTaf.WPFAApplication\bin\Debug\documents\serienbriefvorlagen\firstFormletter.odt

Zuweisungen der Datenfelder zu den Serienbrief-Feldern

Titel	Kunde.Anrede ▼
Vorname	Kunde.Vorname ▼
Name	Kunde.Nachname ▼
Adresse	Kunde.Straße Hsnr ▼
PLZ	Kunde.PLZ ▼
Ort	Kunde.Wohnort ▼
Anrede	Kunde.Einleitung ▼

Im Serienbrief „*firstFormletter.odt*“ gibt es also die Verknüpfungen *Titel*, *Vorname*, *Name*, *Adresse*, *PLZ*, *Ort*, *Anrede*, welche zuvor an den entsprechenden Positionen im Dokument platziert wurden (linke Spalte).

Diese Felder müssen nun den entsprechenden Attributen zugeordnet werden (rechte Spalte).

So wird beispielsweise definiert, dass im Serienbrief an der Position von *Adresse* die Straße und Hausnummer des Kunden angezeigt wird.

Zur Auswahl stehen hier die Attribute von Kunden, Sponsoren und Mitarbeitern. Natürlich macht es keinen Sinn, ein Attribut von Sponsoren oder Mitarbeitern auszuwählen, da in unserem Beispiel alle Empfänger Kunden sind (siehe oben). Das entsprechende Feld im Serienbrief bliebe leer.

Serienbrief drucken

Im Menüpunkt *Druck* können die Serienbriefe gedruckt werden. Dabei muss nur die gewünschte Serienbrief-Vorlage ausgewählt werden. In der anschließenden Ansicht wird die Serienbrief-Vorlage noch einmal dargestellt, inklusive aller Empfänger-Regeln.

Nach dem Klick auf *drucken* öffnet sich LibreOffice mit dem Serienbrief-Dokument. Dort muss ganz normal gedruckt werden. Die Nachfrage, ob ein Serienbrief gedruckt

werden soll, muss mit Ja beantwortet werden. Im Druck-Dialog sieht man nochmal die Inhalte der CSV-Datei, welche in den Serienbrief gespeist werden.

Buchhaltung

Schnellbuchung

Um die im Berechtigungsnachweis per Hand eingetragenen Tageseinnahmen ins System zu übertragen, kann die Funktion *Schnellbuchung* verwendet werden. Diese bietet eine tabellarische Ansicht aller Kunden, welche nach der Gruppe gefiltert ist.

<div>Einnahmen erfassen</div> <div>Gruppe: 1</div>							
Gruppe	AusweisNr	Nachname	Vorname	Betrag	Ausweis gültig bis	Geburtstag	Letzter Einkauf
1	6	Greiner	Andrea		01.07.2013	24.01.1971	
1	7	Stork	Michael		01.07.2013	14.04.1971	
1	17	Maier	Michael		01.07.2013	09.08.1981	
1	18	Bäumle	Paul		01.07.2013	08.03.1965	
1	21	Sizmann	Michael		01.07.2013	23.01.1986	

Es kann für jeden ein Betrag eingegeben werden. Aus jeder Eingabe wird beim Erfassen der Einnahmen eine Buchung generiert, die mit dem jeweiligen Kunden verknüpft ist und in der Beschreibung den Text *Schnellbuchung* beinhaltet.

Für diese generierten Buchungen ist ein standardmäßig konfiguriertes Quell- und Zielkonto zwingend erforderlich. Diese müssen in der Konfigurationsdatei *config.ini* eingetragen sein, z.B.:

```
[ACCOUNTING]
defaultCustomerAccountNr=100
defaultCashBoxAccountNr=200
```

Der Eintrag `defaultCustomerAccountNr` steht für die Standard-Kundenkontonummer, der Eintrag `defaultCashBoxAccountNr` steht für die Standard-Kassenkontonummer.

Hier würden Buchungen mit Quellkontonummer 100 und Zielkontonummer 200 angelegt werden. Gibt es diese Konten nicht, funktioniert die *Schnellbuchung* nicht.

Buchungen

Im Bereich *Buchungen* können Buchungen per Hand eingegeben, durchgesehen und verändert werden.

Buchung erfassen		<< < 1 bis 12 von 14 > >>						
ID	Datum	Quellkonto	Zielkonto	Betrag	Beschreibung	Benutzer	Kunde	
10	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	3,00 €	Brot	ADMIN		
7	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	3,00 €	Brot	ADMIN		
8	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	5,00 €	Brot für 4 Kinder	ADMIN		

Existierende Buchungen werden in einer Tabelle angezeigt, die in mehrere Abschnitte eingeteilt ist. In obigem Beispiel werden von insgesamt 14 Buchungen nur die erste Seite mit jeweils 12 Buchungen angezeigt (1-12). Diese Anzahl kann in der Konfigurationsdatei im Abschnitt [PAGING] verändert werden.

Die Ansicht kann des Weiteren nach Quellkonto-Name/Nummer, Zielkonto-Name/Nummer, Betrag oder Beschreibung gefiltert werden.

Buchungen können so lange bearbeitet und gelöscht werden, bis ein Kassenabschluss gemacht wurde. Wenn Buchungen zu einem Kassenabschluss gehören, können sie nicht mehr geändert oder gelöscht werden, es ist aber noch eine Korrekturbuchung möglich.

Die Korrekturbuchung beeinflusst dabei nicht den Betrag der zur Buchung gehörenden Kassenabschlusses. Sie erstellt lediglich eine neue Buchung mit dem aktuellen Datum, die die Transaktion der zu korrigierenden Buchung umkehrt.

Buchung erfassen

Speichern
Zurücksetzen
Abbrechen

Quellkonto
Kunde

Zielkonto
Bargeldkasse

Betrag
4

Beschreibung
Brot für 4 Kinder

Beim Erfassen einer neuen Buchung ist das Quell- und Zielkonto anzugeben. Hierbei ist zu beachten, ob das jeweilige Konto ein Konto für Eigenkapital bzw. ein offizielles

Konto ist. Im obigen Beispiel ist das Konto *Kunde* kein Konto für Eigenkapital, das Konto *Bargeldkasse* jedoch schon.

Alle Transaktionen müssen aufgrund der Abrechnung semantisch logisch sein. So kann es beispielweise Buchungen von *Kunde* zu *Bargeldkasse*, von *Bargeldkasse* zu *Tresor* oder von *Bargeldkasse* zu *Pfarramt* geben, jedoch keine Buchungen von *Kunde* zu *Pfarramt* oder von *Benzingeld* zu *Mitarbeiterlohn*.

Das Bearbeiten von Buchungen geschieht analog.

Kontensummen

Im Punkt *Kontensummen* können Einnahmen, Ausgaben und Bilanz für ein spezifisches Konto gezeigt werden. Des Weiteren werden hier für das ausgewählte Konto die Buchungen angezeigt. Es kann ein Zeitraum für die angezeigten Buchungen angegeben werden.

Die Ansicht kann des Weiteren nach Quellkonto-Name/Nummer, Zielkonto-Name/Nummer, Betrag oder Beschreibung gefiltert werden.

<div> Zurücksetzen Konto: Kunde (100) Von: 25.03.2013 Bis: 21.07.2013 </div>							
Einnahmen: 0,00 €		Ausgaben: 20,00 €		Bilanz: -20,00 €			
ID	Datum	Quellkonto	Zielkonto	Betrag	Beschreibung	Benutzer	Kunde
10	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	3,00 €	Brot	ADMIN	
8	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	5,00 €	Brot für 4 Kinder	ADMIN	
7	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	3,00 €	Brot	ADMIN	
6	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	1,00 €	Brot	ADMIN	
5	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	2,00 €	Brot	ADMIN	
4	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	2,00 €	Brot	ADMIN	
3	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	1,00 €	Brot	ADMIN	
2	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	2,00 €	Brot	ADMIN	
1	03.07.2013 10:29	Kunde (100)	Bargeldkasse (200)	1,00 €	Brot	ADMIN	

Die anzuzeigenden Beträge werden Buchung für Buchung aufsummiert.

Entspricht das Zielkonto einer Buchung dem ausgewählten Konto, ist es aus Sicht des ausgewählten Kontos eine Einnahme. Entspricht das Zielkonto einer Buchung dem ausgewählten Konto, ist es aus Sicht des ausgewählten Kontos eine Einnahme. Aus diesen akkumulierten Beträgen wird die Bilanz errechnet.

Beispielsweise ist in obigem Beispiel das Konto *Kunde* ausgewählt. Da dieses Konto kein Eigenkapital-Konto ist und es nur Buchungen mit diesem Konto als Quellkonto gibt, werden die Buchungen alle als Ausgaben interpretiert. Bei der Auswahl des Kontos *Bargeldkasse* werden dieselben Buchungen als Einnahmen interpretiert.

Es besteht die Möglichkeit, die Bilanz aller Eigenkapital-Konten anzuzeigen. Dazu wird anstelle eines konkreten Kontos die Option *Alle Eigenkapitalkonten* ausgewählt. Dies zeigt alle Buchungen an, die ein Eigenkapital-Konto entweder als Quell- oder Zielkonto aufweisen.

Für die Berechnung der Einnahmen wird hier geprüft, ob das Zielkonto einer Buchung ein Eigenkapital-Konto ist. Für die Berechnung der Ausgaben wird geprüft, ob das Quellkonto einer Buchung ein Eigenkapital-Konto ist. Aus diesen akkumulierten Beträgen wird wieder die Bilanz errechnet.

In den Datumsfeldern stehen beim ersten Aufruf standardmäßig der genaue Zeitpunkt des letzten Kassenabschlusses im Feld *Von* und der aktuelle Zeitpunkt im Feld *Bis*. Dabei wird nur das Datum, nicht die Zeit angezeigt.

Wenn also nur Buchungen vorliegen, für die bereits ein Kassenschluss gemacht wurde und keine neuen Buchungen von danach vorliegen, ist die Liste der Buchungen daher standardmäßig leer. Daher muss unter Umständen zunächst ein passender Zeit-Bereich mit den Datumsfeldern definiert werden.

Kassenabschluss



Übersicht

Der Kassenabschluss wird am Ende eines Abrechnungstages gemacht und stellt die Fixierung der seit dem letzten Kassenabschluss erstellten Buchungen dar. Es wird eine Tagesbilanz errechnet und unveränderlich gespeichert. Zu einem Kassenabschluss gehören ein Kassenabschlussbericht und eine Kassenabrechnung.

Der Kassenabschlussbericht (im alten KöTaf-System „Kassensammler“) stellt eine Quittung dar, die zusammen mit dem gesammelten Geld ans Pfarramt überreicht wird.

Die Kassenabrechnung ist ein doppelter Beleg, von dem einer für die Unterlagen der Tafel bestimmt ist, der andere wird an die Caritas geschickt.

Folgende Abbildung zeigt einen kompletten Kassenabschluss:

Kassenabschluss erstellen								
Kassenabschluss	Benutzer	Eing.	Ausg.	Summe		Beleg gedruckt	Datum	Benutzer
03.07.2013 10:29	ADMIN	120,00 €	90,00 €	30,00 €	 	<input checked="" type="checkbox"/>	21.07.2013 13:21	ADMIN

...

Beleg erledigt	Datum	Benutzer	Abrechnung gedruckt	Datum	Benutzer	Kommentar
<input checked="" type="checkbox"/>	21.07.2013 13:22	ADMIN	<input checked="" type="checkbox"/>	21.07.2013 13:22	ADMIN	Erster Kassenabschluss
...						

Kassenabschlussbeleg und Kassenabrechnung können hier direkt ausgedruckt werden. Es kann nur eine Kopie ausgedruckt werden, Folgedrucke werden mit dem Vermerk „Nachdruck“ gekennzeichnet.

Folgende Tabelle dient zur Erklärung der Spalten zu den Kassenabschlüssen:

Kassenabschluss	Datum und Uhrzeit des Kassenabschlusses
Benutzer	Benutzer, der den Kassenabschluss durchführte
Eing.	Summe Einnahmen
Ausg.	Summe Ausgaben
Summe	Bilanz
(Druck-Buttons)	1. Button: Kassenabschlussbeleg, 2. Button: Kassenabrechnung
Beleg gedruckt	Zeigt, ob der Kassenabschlussbeleg im Original schon gedruckt wurde
Datum	Datum und Uhrzeit des Originaldrucks des Beleges
Benutzer	Benutzer, der den Originaldruck des Beleges durchführte
Beleg erledigt	Kann einmalig aktiviert werden, wenn die Quittung vom Pfarramt da ist
Datum	Datum und Uhrzeit der Erledigt-Markierung
Benutzer	Benutzer, der den Beleg als erledigt markiert hat
Abrechnung gedruckt	Zeigt an, ob die Kassenabrechnung im Original schon gedruckt wurde
Datum	Datum und Uhrzeit des Originaldrucks der Abrechnung
Benutzer	Benutzer, der den Originaldruck der Abrechnung durchführte
Kommentar	Kommentar zum Kassenabschluss

Kassenabschluss erstellen

Beim Erstellen eines Kassenabschlusses werden nur die Konten berücksichtigt, die als Eigenkapital-Konto/Offiziell markiert wurden. Dies sind typischerweise Konten, bei denen Geldfluss in beide Richtungen möglich ist. Im Einsatz an der Königsbrunner Tafel wird dies voraussichtlich nur bei der Bargeldkasse der Fall sein.

Kasse schließen
Abbrechen

Folgende Konten werden im Kassenabschluss berücksichtigt:

Kontonummer	Kontoname	Beschreibung	Nullzyklus	Altes Saldo	Einnahmen	Ausgaben	Neues Saldo
200	Bargeldkasse	Kassenbox für Bargeld	Kassenabschluss	0,00 €	56,00 €	20,00 €	36,00 €
700	Tresor	Konto für den hausinternen Tresor	Niemals	0,00 €	0,00 €	0,00 €	0,00 €

Einnahmen Gesamt 56,00 €

Ausgaben Gesamt 20,00 €

Summe Gesamt 36,00 €

Kommentar zu diesem Kassenabschluss:

In diesem Beispiel werden alle Buchungen der Konten, die im Kassenabschluss berücksichtigt werden, nach Einnahmen und Ausgaben aufsummiert und dargestellt. Es gilt:

$$\text{Neues Saldo} = \text{Altes Saldo} + \text{Einnahmen} - \text{Ausgaben}$$

Da beispielsweise beim Konto *Bargeldkasse* der Nullzyklus nach jedem Kassenabschluss erreicht wird, wird der neue Saldo für dieses Konto nach diesem Kassenabschluss auf 0,- gesetzt. Der Saldo wird hierbei nur für die Konten des Kassenabschlusses aktualisiert.













Konten, die beispielsweise einen jährlichen Nullzyklus aufweisen, werden anschließend im Saldo auf 0,- gesetzt, wenn zwischen dem letzten und dem aktuellen Kassenschluss ein Jahreswechsel stattgefunden hat. Dies geschieht mit einem monatlichen Nullzyklus analog.

Benutzerverwaltung

Übersicht

In der Benutzerverwaltung werden die Systembenutzer angezeigt. Dabei hat jeder Benutzer ein eindeutiges Kürzel und eine Administrator-Kennzeichnung.

Administratoren haben Zugriff auf die Verwaltung und die Funktionen darunter: Serienbrief-Verwaltung, Backup-Wiederherstellung, Benutzerverwaltung, Kontenverwaltung

Neuer Benutzer		
Benutzername	Admin	
MM	<input type="checkbox"/>	 
MK	<input type="checkbox"/>	 
KG	<input type="checkbox"/>	 
GS	<input type="checkbox"/>	 
BL	<input type="checkbox"/>	 
ADMIN	<input checked="" type="checkbox"/>	 

Neuer Benutzer

Beim Anlegen eines Benutzers muss der Benutzername, das Passwort und die Administrator-Kennzeichnung angegeben werden.

Optional kann ein Benutzerbild angegeben werden, das im Anmeldefenster bei Eingabe des Benutzernamens angezeigt wird.

Speichern
Zurücksetzen
Abbrechen

Benutzername

Passwort

Passwort wiederholen

Administratorrechte ☐

Benutzerbild

Das Benutzerbild darf dabei nicht kleiner als die in der Konfigurationsdatei angegebenen Maße sein. Diese sind in der Sektion [IMAGES] unter *imageWidth* und *imageHeight* angegeben und spielen eine Rolle bei der Einhaltung des Layouts des Anmeldefensters.

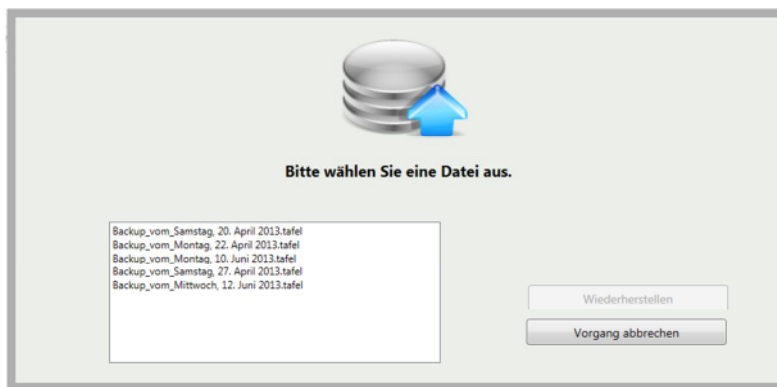
Das Benutzerbild wird beim Speichern ins Grafikverzeichnis der Anwendung geladen, das in der Konfigurationsdatei im Abschnitt [SETTINGS] definiert wurde. Der Anwendungspfad wird durch den Platzhalter %PROGRAMPATH%\Images gesetzt.

Die erlaubten Dateierweiterungen für Grafiken können ebenfalls in der Konfigurationsdatei im Abschnitt [IMAGES] eingestellt werden.

Die Bearbeiten-Funktion der Benutzerverwaltung geschieht analog zum Anlegen eines neuen Benutzers.

Wiederherstellung einer Sicherungsdatei

Durch drücken des Buttons „Backup-Wiederherstellung“ wird es einem Administrator ermöglicht, eine Sicherungsdatei von einem externen USB – Gerät wiederherzustellen. Dazu wird das USB - Gerät mit dem PC verbunden. Nachdem das USB - Gerät vollständig erkannt und geladen wurde, wird die nachfolgende Maske angezeigt.



Hier kann der Benutzer nun einen beliebigen „Zeit –Stand“ einer Datenbank wiederherstellen. Ein Neustart des Programms nach einer Wiederherstellung ist nicht erforderlich.

Kontenverwaltung

Übersicht

In der Kontenverwaltung können kaufmännische Konten eingerichtet werden, die für die doppelte Buchführung erforderlich sind.

Konto eröffnen		<< < 1 bis 9 von 9 > >>			
Kontobezeichnung	Kontonummer	Beschreibung	Kassenabschluss	Nullzyklus	Fixkonto
Kunde	100	Bargeld von Kunden	Nein	Kassenabschluss	Ja
Bargeldkasse	200	Kassenbox für Bargeld	Ja	Kassenabschluss	Ja
Benzinkosten	300	Ausgaben für Benzin und Diesel	Nein	Jährlich	Nein

In der Übersicht werden unter anderem der Nullzyklus und der Fixkonto-Vermerk angezeigt.

Der Nullzyklus gibt an, wann der Saldo für das jeweilige Konto auf 0,- gesetzt werden soll.

Der Vermerk *Fixkonto* gibt an, ob dieses Konto als Fixkonto angelegt wurde. Diese Konten sind weder löscht- noch änderbar.

Konto anlegen

Kontoname

Kontonummer

Kontobeschreibung

Offizielles Konto für Eigenkapital ☒

Wann soll der Saldo dieses Kontos auf den Betrag Null gesetzt werden?

Dieses Konto als unveränderbares Fixkonto anlegen ☐

Beim Anlegen eines Kontos müssen Kontoname, Kontonummer und Kontobeschreibung eingegeben werden. Handelt es sich um ein Konto für Eigenkapital, muss dies angegeben werden.

Für das Konto muss der Nullzyklus angegeben werden. Es gibt die Möglichkeiten *Kassenabschluss* (für jeden Kassenabschluss), *monatlich*, *jährlich* und *Niemals*.

Da beispielsweise die Bargeldkasse am Ende jedes Tages geleert wird, ist es sinnvoll, am nächsten Tag mit dem Saldo 0,- in der Bargeldkasse zu beginnen. Dies erfordert eine „Leerung“ bei Kassenabschluss. Wenn beispielsweise der jährliche Saldo des Benzinkosten-Kontos interessant ist, muss der Nullzyklus bei diesem Konto auf *Jährlich* gesetzt werden.

Wird das Konto als unveränderbares Fixkonto angelegt, kann dieses Konto niemals mehr verändert werden. Für die meisten Konten ist dies nicht erforderlich.

Druck

Generierte Dokumente

Formulare



Im Modul *Druck* können die gezeigten Dokumente ausgedruckt werden. Der Berechtigungsnachweis wird direkt mit *LibreOffice Calc* geöffnet, alle anderen Dokumente basieren auf *OpenOffice Writer*.

Der Punkt *Serienbrief-Druck* erzeugt noch eine Auswahl an den verfügbaren Serienbrief-Vorlagen.

Im Quellcode sind unter `KöTaf.WPFApplication.Helper.PrintForms` die Funktionen zum Druck der verschiedenen Formulare zu finden.

Notizen

Im Modul *Notizen* können simple Notizen angelegt, bearbeitet und gelöscht werden.

Name	Beschreibung	Datum	
Nach Kassenabschluss buchen	Nach nächstem Kassenabschluss unbedingt noch buchen: ...	21.07.2013	 

Beim Anlegen der Notiz ist darauf zu achten, dass aufgrund von Datenbankbeschränkungen eine Zeichenfolge maximal 4000 Zeichen lang sein darf.

Name:

Beschreibung:

Länge: 56

Verwaltung

Im Punkt *Verwaltung* sind die Serienbrief-Verwaltung, die Backup-Wiederherstellung, die Benutzerverwaltung und die Kontenverwaltung vorhanden.

Backup der Datenbank

Allgemein

Bei jedem Beenden des Kötaf – Programms wird der Benutzer aufgefordert, ein gültiges USB-Speichermedium an den PC anzuschließen. Ein USB – Speichergerät gilt als gültig, wenn die Volumenbezeichnung mit der, der Konfigurationsdatei übereinstimmt. Das Backup läuft durch die automatische USB – Erkennung völlig selbständig ab. Anschließend wird dem Benutzer durch eine Meldung signalisiert, ob das Backup vollständig oder fehlerhaft war. Das Backup wird einmal pro Tag ausgeführt.

Details

- Beim Programmende wird durch die Klasse CloseProgramm() wird die USB_Identification aufgerufen. Diese prüft als erstes auf den „Timestamp“, also ob an dem besagten Tag schon ein Backup durchgeführt wurde. Dieser Zeitstempel liegt als XML Datei im Root Verzeichnis des Programms unter dem Namen „TimeStamp.xml“ ab. Die zugehörige Funktionen werden aus der Klasse „/Kötaf.Utills/BackupRestoreDB/Timer.cs“ aufgerufen.
- Die Funktion dispatcherTimer_Tick() prüft ab, ob ein USB-Speichermedium eingelegt wurde.
- Anschließend wird in der Klasse USB_Identification.cs durch Aufruf der Klasse „/Kötaf.Utills/BackupRestoreDB/CheckForUSB.cs“ auf ein gültiges USB – Speichergerät (VolumenLabel) geprüft.
- Falls ein gültiges Speichermedium erkannt wurde, wird nun (durch Übergabe-Parameter) in der Klasse USB-Identification.cs entschieden, ob ein „Backup“ oder „Restore“ durchgeführt wird.
- In beiden Fällen wird die Klasse „/Kötaf.Utills/BackupRestoreDB/BackupRestoreDatabase.cs“ aufgerufen.
- In dieser „BackupRestoreDatabase.cs“ wird nun das „Backup“ oder „Restore“ durch Aufruf einer Switch – Anweisung ausgeführt. Bei einem „Restore“ wird zusätzlich durch die Funktionen() _DatabaseValidator und _CheckForEqualDB() geprüft, ob die zu wiederherzustellende Datenbank, die gleiche Struktur, wie die aktuelle Datenbank aufweist (Validierung).

Die Konfigurationsdatei

[APPSETTINGS]

Verschiedene Strings, die sich auf die Königsbrunner Tafel beziehen

`programTitle`

Der Titel für das Programm (Z.B. Königsbrunner Tafel)

`topBarColor`

Hintergrundfarbe für die Titelleiste oben

`toolbarColor`

Hintergrundfarbe der Toolbar

[RESTORE_TIMER]

`DAYS_UNTIL_DELETE_FILES` → Achtung dieser Wert gibt die Tage an, wie alt ein Backup (die gesicherte Datei) sein darf, bevor diese gelöscht wird. Angabe in Tagen. (Default = 365).

`TIME_FACTOR_TO_ENABLED_BACKUP_CANCEL_BUTTON`

Zeitfaktor, bis Backup abgebrochen werden kann. (Default =12)

[USB]

`VOLUME_LABEL`

Die Laufwerksbezeichnung für den Backup-USB-Stick

[PAGING]

`STEPSIZE=12`

Hiermit wird festgelegt, wie viele Elemente auf einer Seite des DataGrids zu sehen sind. Somit kann der Umbruch der Datensätze an den Monitor bzw. den Betrachter angepasst werden.

[VALIDATION]

In diesem Bereich sind alle Regexfunktionen für die Feldvalidierungen hinterlegt, bei fehlerhaften Regex kommt es standardmäßig zu einer falschen Validierung! Eingaben werden dann als inkorrekt markiert!

- `PHONENO=^((\+[0-9]{2,4}([-]?[0-9]+?[-]? ?\([0-9]+?\) ?))|(\(0[0-9]+?\) ?)|([0-9]+? ?(|-|\/)?))([0-9]+?[\\/-]?)+?[0-9]$`

Diese Funktion überprüft die Telefonnummern auf die gängigen Schreibweisen.

- MOBILENO= $^((00|\backslash+)49)?(0?1[5-7]\{1\}[0-9]\{1\})((\backslash-|\backslash/)?)([0-9]\{1,3\})(\backslash-|\backslash/)?([0-9]\{1,3\})(\backslash-|\backslash/)?([0-9]\{1,2\})\$$

Dient der Überprüfung von Mobilfunknummern, erlaubt auch Sonderzeichen nach der Vorwahl bzw. jeweils drei Zahlen („;“, „-“, „/“).

- EMAIL= $^\\w[-_\\w]^*\\w@\\w[-_\\w]^*\\w\\.\\w\\{2,3}\\$$
- PLZ= $^[0-9]\{5\\}\$$
- NUMBER= $^[0-9]\$$
- NAME= $^[a-zA-ZäöüÄÖÜ]\{2,\\}\$$
- DOUBLE= $^\\d+(,|.)?\\d\\{0,2\\}\$$

Anlegen und Editieren von Kunden

Im Untermenü Kunden ist es möglich neue Kunden anzulegen, bereits bestehende anzuzeigen oder diese zu bearbeiten.

Kundenverwaltung

Neuer Kunde Drucken << < 1 bis 12 von 100 > >> Name Suche ...

Anzeige: alle

Ausweisnr.	Name	Wohnort	Geburtsdatum	Nationalität	Gruppe
6	Greiner Andrea	10592 Königsbrunn, Herzogweg 168n	24.01.1971	italienisch	1

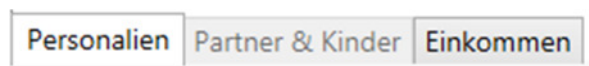
Durch einen Klick auf „Drucken“ lässt sich eine Kundenliste ausdrucken, im rechten oberen Bereich gibt es eine Suchfunktion nach diversen Kriterien, direkt darunter befindet sich der Anzeigefilter, hiermit können z.B. „aktive“ und „inaktive“ Kunden gesondert angezeigt werden.

Mit einem Klick auf „Neuer Kunde“ im oberen linken Bereich, öffnet sich die Maske zum Anlegen eines Neukunden, analog dazu können Kunden auch durch einen Klick auf den „Bleistift“ in der Zeile des Kundendatensatzes bearbeitet werden.

Die jeweiligen Druckbuttons erlauben das Drucken der benötigten Dokumente, durch einen Klick auf das „blaue Männchen“ lässt sich der Kundenstatus von aktiv auf inaktiv und andersherum setzen.

Personalien

Beim Anlegen eines neuen Kunden ist zu beachten, dass dieses Untermenü aus drei Tabs besteht welche nur unter gewissen Voraussetzungen freigeschaltet werden.



Der Tab „Personalien“ ist standardmäßig verfügbar um den Kunden selbst anzulegen. Möchte man nun einen Partner bzw. die Kinder des Kunden angeben gibt es hierzu zwei verschiedene Möglichkeiten.

Familienstand: LD - Ledig Anz. Kinder: 0 Kind hinzufügen

Wird der passende Familienstatus gewählt (Verheiratet, Lebenspartnerschaft, etc.) wird der Tab „Partner & Kinder“ freigeschaltet und die Felder für den Partner aktiv gesetzt. Durch einen Klick auf „weiter“ oder einen Klick direkt auf den Tab kann man dann in den jeweiligen Tab wechseln.

Die zweite Möglichkeit wäre es durch einen Klick auf den Button „Kind hinzufügen“ ein Kind hinzuzufügen und dadurch den Tab „Partner & Kinder“ freizuschalten. Dabei ist zu beachten, dass je nach gewähltem Familienstatus die Felder für den Partner inaktiv bleiben.

Sind weder Partner noch Kinder vorhanden kann mit einem Klick auf „weiter“ das Tab für Einkommen gewählt werden.

Um die Personendaten speichern zu können **müssen** folgende Felder zwingend, korrekt ausgefüllt werden:

- Anrede
- Vorname, Nachname
- Geburtsdatum
- Straße, Postleitzahl und Stadt
- Nationalität
- Alle weiteren Pflichtfelder werden standardmäßig belegt

Partner & Kinder

In diesem Tab werden der Partner und die Kinder des Kunden, insofern vorhanden, verwaltet. Bei den Eingabefeldern des Partners gelten die selbigen Regeln wie beim Kunden selbst! Das Anlegen der Kinder hingegen erfolgt mithilfe einer Liste welcher Elemente hinzugefügt bzw. Elemente entfernt werden können.

Kinder

Vorname	Nachname	Geburtsdag	Geschlecht	
Max	Mustermann	21.07.2013	Männlich	<input checked="" type="checkbox"/>

Hat man eine Zeile für ein Kind ausgefüllt kann dies durch einen Klick auf den Haken bestätigt werden, dadurch wird dieser Datensatz für die Überprüfung der Eingaben

vermerkt! Ist ein Datensatz vermerkt kann dieser durch einen Klick auf das rote „X“ wieder entfernt werden. Bei der Speicherung werden nur Datensätze berücksichtigt, die mittels des grünen Hakens als „bestätigt“ markiert wurden. Daten in einer unbestätigten Zeile werden nicht berücksichtigt.

Einkommen

Jeder Kunde weist in der Regel irgendwelche Einkommensarten auf. Um auch diese festzuhalten gibt es im letzten Tab die Möglichkeit, ähnlich wie bei den Kindern, diese Einkommen mittels einer Liste zu verwalten.

Einkommens-Art	Bezeichnung	von	bis	Betrag	
Waisenrente	Waisenrente	01.07.2013	22.07.2013	123	
		22.07.2013	22.07.2013	0	

Einkommen gesamt: 123

Wie in der Abbildung zu sehen ist, funktioniert diese Liste nach dem gleichen Prinzip wie die Liste zur Verwaltung der Kinder, einziger Unterschied ist, das Beitragsfeld, welches nur Ganzzahlen und Dezimalzahlen mit zwei Nachkommastellen aufweisen muss. Die Beträge aller Einkommen werden nach dem hinzufügen zur Überprüfung zusammen gerechnet.

Speichern der Daten

Wurden alle Eingabefelder gefüllt kann durch einen Klick auf „Speichern“ der Datensatz geprüft und demnach in der Datenbank abgelegt werden, dabei sind folgende Punkte zu beachten:

1. Die jeweiligen Tabs werden nacheinander auf Gültigkeit geprüft, nach jedem Tab gibt es zunächst eine aussagekräftige Fehlermeldung
2. Die Daten werden erst gespeichert wenn **alle** Felder korrekt ausgefüllt wurden und keine weiteren Fehler auftreten.
3. Bei einem Klick auf „Abbrechen“, in egal welchem Tab, werden alle Eingaben gelöscht und nicht gespeichert!

Editieren der Datensätze




Im Allgemeinen bleibt zu sagen, dass der Vorgang des Editierens von Datensätzen analog funktioniert. Der einzige Unterschied liegt in der Vorbelegung der Eingabefelder.

Anlegen und Editieren von Sponsoren

In diesem Menü ist es möglich neue Sponsoren anzulegen und bereits vorhandene zu bearbeiten.

Sponsorenverwaltung

Neuer Sponsor Liste drucken << < 1 bis 12 von 100 > >> Name Suche ...

Name	Wohnort	Typ	Firma	Firmenname
Bäurle Franz	53626 Königsbrunn, Am Roten Tor 144u	Finanziell	  	Bäcker Meier

In dieser Übersicht sind alle Sponsoren zu sehen, durch einen Klick auf „Neuer Sponsor“ wird die Maske zum Anlegen eines neuen Sponsoren geladen, durch einen Klick auf den „Bleistift“ kann ein bereits bestehender Sponsor bearbeitet werden. Im oberen rechten Bereich befindet sich das Suchfeld.

Neuer Sponsor

Dieses Fenster erlaubt es uns einen neuen Sponsor in die Datenbank zu speichern, dabei gilt es ein paar Punkte zu beachten. Sponsoren müssen nicht zwingend Privatleute sein, weshalb es wichtig ist zu unterscheiden ob ein Sponsor nun eine Firma ist oder nicht.

Hierfür dient die Schaltfläche „Firmen-Sponsor“. Ist der Haken gesetzt, erscheint ein zusätzliches Pflichtfeld „Firmenname“. Des Weiteren beziehen sich die Felder „Nachname“ und „Vorname“ dann auf die Kontaktperson der jeweiligen Firma, ebenso wie die Anschrift-Felder.

Firmen-Sponsor: ☐

Anrede:

Nachname: Vorname:

Ein weiterer wichtiger Punkt ist die Erlaubnis einem Sponsor Serienbriefe zu schicken, hierzu gibt es die Schaltfläche **Serienbrief:** ☒ welche Standardmäßig aktiv ist. Wird der Haken entfernt wird der Sponsor nicht mehr in den Serienbriefen berücksichtigt.

Bei der Sponsorenanlage sind folgende Felder zwingend auszufüllen:



- Vorname, Nachname
- Straße, Postleitzahl und Stadt
- Bei Firmenkunden: Firmenname

Editieren von Sponsoren

Grundsätzlich funktioniert die Bearbeitung von Sponsoren genauso wie das Anlegen dieser, der einzige Unterschied ist die Vorbelegung der Eingabefelder beim Laden der Maske.

Anlegen und Editieren von Teammitgliedern

Neben der Kunden- und Sponsorenverwaltung ist es auch notwendig das Team im Überblick zu haben, hierfür gibt es das Untermenü „Team“ welches es uns erlaubt Teammitglieder anzulegen oder bereits bestehende zu bearbeiten.

Teamverwaltung		
<div>Neuer Mitarbeiter Liste drucken << < 1 bis 12 von 15 > >> Name Suche ...</div>		
Name	Wohnort	Funktion
Banane Annabell	15355 Bobingen, Goetheplatz 33r	Ausgabe  

Diese Liste bietet einen Überblick über alle Teammitglieder, wobei inaktive Mitglieder ans Ende der Liste sortiert werden. Durch einen Klick auf „Neuer Mitarbeiter“ oder einen Klick auf den „Bleistift“ können Mitarbeiter angelegt bzw. bearbeitet werden.

Anlegen von Teammitgliedern

Anrede:	<input type="text" value="Herr"/>	Geburtstag:	<input type="text" value="Datum auswählen"/>
Nachname:	<input type="text"/>	Vorname:	<input type="text"/>
Straße:	<input type="text"/>		
PLZ:	<input type="text"/>	Ort:	<input type="text"/>
Telefon:	<input type="text"/>		
Mobil:	<input type="text"/>		
E-Mail:	<input type="text"/>		
aktiv:	<input checked="" type="checkbox"/>	Fahrer:	<input type="text" value="Fahrer"/>
Serienbrief:	<input checked="" type="checkbox"/>		

Möchte man ein Teammitglied anlegen öffnet sich obige Maske. Hierbei ist auch wieder zu beachten, dass Mitarbeiter von Serienbriefen ausgeschlossen werden können!

Um Mitarbeiter anzulegen müssen folgende Felder verpflichtend ausgefüllt werden:

- Geburtstag

- Nachname, Vorname
- Straße, Postleitzahl und Ort

Editieren von Teammitgliedern

Beim Editieren von Mitarbeitern liegt der Unterschied lediglich in der Vorbelegung der Eingabefelder.

Listen – Klassen – Hierarchie

Die pLists Klasse instanziiert die Klassen pAnyLists, pSavedLists und die pFormletterFilterSelection. Des Weiteren sorgt diese dafür dass beim Klick auf den „Listen“-Button die Tab-Selection mit den instanziierten Klassen angezeigt wird.

Die pDisplayedData-Klasse wird zur Auswahl der Anzuzeigenden-Daten in pAnyLists und pSavedLists verwendet.

Import der Bestandsdaten

Vorgehensweise beim Import:

1. Die alte Datenbank z.B. in XAMPP einfügen, darauf achten das der MySQL Server gestartet ist und eine Verbindung möglich ist
2. Darauf achten das in der Bestandsdatenbank der Benutzer ADMIN, Kennwort: ADMIN gelöscht wurde, da dieser zum späteren Zeitpunkt ggf. zweimal in der neuen Datenbank vorhanden wäre
3. Das Import Programm (KöTaf.DataImporter) starten und Benutzerdaten eingeben:
 Sofern in XAMPP nichts abgeändert wurde, müssen diese Daten eingegeben werden
 Datenbank: tisch
 Benutzername: root
 Kennwort: << KEINES EINGEBEN >>
4. Nachdem der Import Prozess abgeschlossen wurde, wird ein Verzeichnis geöffnet. Die generierte Datenbank (TischDB.sdf) kopieren und diese im eigentlichen KöTaf Programm reinkopieren.

ANMERKUNG: Der MySqlConnection Connector muss installiert sein, damit das Programm ordnungsgemäß funktioniert. Zu finden unter
KöTaf.DataImporter_\CONNECTOR\mysql-connector-net-6.6.5.msi

Arbeitsaufteilung

Florian Wasielewski

Projekt KöTaf.DataModel

- Entwurf der Datenbank
- Implementierung der Datenbank + Code (Entity Framework / komplette Logik (Add, Delete, Update, Get, ...))
- Import der alten Daten in die neue DB
- Erstellen von SQL-Skripts zum Einfügen von Test Daten in der Datenbank

Projekt KöTaf.WPFApplication

- Erstellen der Notizen + Implementierung
- Teilimplementierungen in mehreren Views (Suche, Datagrid XAML, etc.)
- Anpassen der DatagridHelper Funktion für unsere Zwecke
- Personen ausgrauen beim deaktivieren

Projekt KöTaf.Utils

- CSVExporter - Modul zum transformieren von dynamisch erzeugten Datagrids in CSV Dateien

Sonstiges

- diverse Programme zum erfolgreichen Start von KöTaf auf KöTaf-PC installiert
- Programm zu erzeugen von Testdatensätze erstellt

Michael Müller

Projekt KöTaf.WPFApplication

- Implementierung des „Loginfensters“
- Implementierung des „SplashWindow“
- Validierung der von der Konfigurationsdatei kommenden Werte.

Projekt KöTaf.Utils

- Implementierung des automatischen USB Backup und Restore
- Implementierung einer Kalender-Wochen-Berechnungsklasse für den Berechtigungsausweis

- Implementierung eines CSV-Readers/ Writers
- Implementierung zum Lesen und Schreiben von XML Dateien (Timestamp → Backup heute schon durchgeführt?)
- Validierung der von der Konfigurationsdatei kommenden Werte.
- Implementation von Klassen zur Generierung von „Excel-Sheet“ Dateien für den Druck. (Sponsor, Kunde, Team, Statistik, Berechtigungsausweis) in Form von *.xml Dokumenten.
→ Implementation von einer Druckschnittstelle
- Implementation zum „handeln“ der „Restore“ Dateien.
- Validierung „LoginBilder“ auf Größe

Sonstiges

- Erstellung eines VBA Scripts für LibreOffice (/src/LibreOffice) → Automatisierung LibreOffice.
- Grundinstallation des Kötaf PC's (Programm Installationen)
- Detailliertere Aufführung der Punkte siehe Datei (Wer_hat_Wo_dran_gearbeitet.txt)
- Erstellung der Klassendiagramme (/ _DOKUMENTATION/CLASS_DIAGRAM).

Antonios Fesenmeier

Projekt KöTaf.DataModel

- Entwurf der Datenbank

Projekt KöTaf.WPFApplication

- Entwurf diverser „Models“ zur Verarbeitung der Daten in den DataGrids
- Implementierung der Teamverwaltung
 - Insbesondere Designs der einzelnen Seiten
 - Anlage/ Editierung von Teammitgliedern
 - Grundlegende Logik
- Implementierung der Kundenverwaltung
 - Designs der einzelnen Seiten
 - Anlage / Editierung von Kunden, Kindern und Einkommen
- Implementierung der Sponsorenverwaltung
 - Designs der einzelnen Seiten
 - Anlage / Editierung der Sponsoren

Projekt KöTaf.Utils

- Implementierung des gesamten Validierungs-Automaten für Benutzereingaben

Sonstiges

- Erstentwurf des Programmdesigns

Georg Schmid

Projekt KöTaf.DataModel

- Entwurf der Datenbank

Projekt KöTaf.WPFApplication

- Entwurf und Implementierung der Paging-Funktionalität für die DataGrids inklusive Beachtung von Abhängigkeiten zwischen Filterung, Suche und Paging
- Implementierung der Teamverwaltung
 - Editierung von Teammitgliedern
- Implementierung der Kundenverwaltung
 - Anlage / Editierung von Kunden, Kindern und Einkommen
 - Insbesondere Abhängigkeiten zwischen den Tabs und Speicherung von Änderungen
- Implementierung der Sponsorenverwaltung
 - Editierung der Sponsoren
- Erweiterung der Toobar um Pagingfunktionalität und Suchfunktion
- Implementierung der Filterung (aktiv/inaktiv) bei Kunden und Sponsoren

Sonstiges

- Grobentwurf des Programmdesigns

Dietmar Sach

Projekt KöTaf.Utils

- INI-Parser für die Konfigurationsdatei
- Parser für die Windows-Registry
- OpenDocument Parsing / Manipulationsfunktionen
- Druckfunktion mit den OpenDocument Manipulationsfunktionen

Projekt KöTaf.WPFApplication

- Filter: Entwurf und Implementierung einer allgemeinen Datenstruktur
 - Filter: Tabellen, Spalten, Arten, Operationen, Werte
 - Filter-Sets: Aggregation von semantisch verknüpften Filtern
 - Datenbankstruktur

- Serienbriefe: Konzeptionierung und Implementierung des Moduls
 - Empfänger-Filter
 - Implementierung eines OpenDocument-Formats zur Verarbeitung von Serienbriefen
 - Zuweisung von Textfeldern
- Programmdesign (Template)
 - Erstellung eines einheitlichen Designs, das die Bedürfnisse aller Module erfüllt
 - Implementierung eines dynamischen Template-Systems mit eigenen Schnittstellen
 - Programmweite Anpassung von Klassen zur Übereinstimmung mit dem Design
 - Anlegen von globalen Stilen
- Buchhaltung
 - Schnellbuchungen
 - Buchungen
 - Kontensummen
 - Kassenabschluss
- Verwaltung
 - Benutzerverwaltung
 - Kontenverwaltung

Lucas Kögel

Projekt KöTaf.WPFApplication

- Statistik (pGeneralStatistic)
- Beliebige Listen (pAnyLists)
- Einbindung der Listen-Erstellung
 - Falls nicht als Admin eingeloggt -> (pAccessDenied)
- Anzuzeigende Daten Page (pDisplayedData)
- Listen initialisierungs Page (pLists)
- Redesign von OldpStatistiken zur Aufteilung in die einzelnen Klassen
 - Anpassungen an allen Lists-Klassen und GeneralStatistic die für das neue Design nötig waren

Sonstiges

- Use-Case-Diagramme zur Allgemeinen-Statistik, Beliebigen-Listen und zur Auswahl der Anzuzeigenden-Daten
- Designentwurf der Beliebigen-Listen (Filter, Anzuzeigenden-Daten und Gefundene Daten)

Björn Bittner

Projekt KöTaf.WPFApplication

- Statistik (pGeneralStatistic)
- Beliebige Listen Übersetzung
- Gespeicherte Listen (pSavedLists)
- Anpassungen der FilterSet-Erstellung von Serienbriefen für die ListenErstellung
- Filterung von Listen
 - Anschließend komplette Überarbeitung und Wiederverwendung von Filterung der Serienbriefe
- Komplette Absprache und Zusammenarbeit mit Serienbriefmodul(Dietmar) um möglichst hohe Wiederverwendbarkeit von Filterung zu erreichen

Sonstiges

- Use-Case-Diagramme zur Allgemeinen-Statistik, Beliebigen-Listen und zur Auswahl der Anzuzeigenden-Daten
- Designentwurf der Beliebigen-Listen (Filter, Anzuzeigenden-Daten und Gefundene Daten)

Patrick Vogt

- Entwurf der Buchhaltung
- GUI-Entwurf der Buchhaltung
- Implementierung der Buchhaltung
 - AccountManager
 - Auflistung der Konten
 - Anlegen, bearbeiten und löschen von Konten
 - Buchungen
 - Auflistung von Buchungen
 - Anlegen und bearbeiten von Buchungen
 - Korrekturbuchung
- Verwendung des IniParsers für Texte in C#-Files
 - Erstellen von Labels
 - Anwendung des IniParsers.GetSettings(...)
- Implementierung der Benutzerverwaltung
 - Erstellen eines neuen Benutzers mit optionalem Benutzerbild

- Beim Editieren eines Benutzerbildes wird das alte Bild aus dem Programmverzeichnis gelöscht und das neue in das Programmverzeichnis hineinkopiert
- Bearbeiten und aktivieren/deaktivieren-Funktion von Benutzern
- Anpassung des Logins (nur aktive Benutzer können sich anmelden)

Wartungshinweise

Modul Druck

Die Seite mit den Druck-Buttons kann in *KöTaf.WPFApplication.Views.Print* modifiziert werden.

Die Dateinamen der Dokumente werden aus der Konfigurationsdatei *config.ini* aus dem Abschnitt [DOCUMENTS] gelesen.

Die gleichnamigen Dokumente zum Druck werden im Verzeichnis *Anwendungsverzeichnis/documents/statische_dokumente* abgelegt.

ToDo-Liste

Sortierfunktion in den DataGrids

Momentan ist die Sortierfunktion in den Datagrids deaktiviert, da diese nur auf die Items zurückgreift die momentan dargestellt werden (Diese sind aber nicht vollständig, aufgrund des Pagings)

Hier müsste also noch eine Funktionalität geschaffen werden, die es ermöglicht einen Klick auf den Header einer Spalte abzufangen, zu interpretieren, in welcher Spalte man sich befindet und den kompletten Datenbestand zu sortieren und das Paging damit neu zu befüllen.

Abfangen des Klick-Events:

```
<DataGrid.ColumnHeaderStyle>
    <Style TargetType="DataGridColumnHeader" >
        <Setter Property="FontWeight" Value="Bold" />
        <EventSetter Event="Click"
Handler="columnHeader_Click" />
    </Style>
</DataGrid.ColumnHeaderStyle>
```

Hinzufügen eines Click-Events in den Header eines DataGrids

```
private void columnHeader_Click(object sender, RoutedEventArgs e)
{
```

```

        ...
    }

```

In der zugehörigen Administration, kann dieses Click-Event abgefangen werden und verarbeitet werden.

Hierzu müsste man noch „IComparer“ schreiben

z.B.:

```

public class TeamComparer : IComparer<Team>
{
    public int Compare(Team x, Team y)
    {
        Team team1 = x;
        Team team2 = y;
        int t = String.Compare(team2.LastName,
team1.LastName);
        return String.Compare( team2.LastName,
team1.LastName);
    }
}

```

Diese ermöglichen es Listen von Objekten miteinander zu vergleichen. Solche IComparer müssten für die verschiedenen Objekte der Datenbank, mit den jeweiligen gewünschten Eigenschaften, erstellt werden und im Click-Event angewendet werden.(Beachtung der Sortierrichtung fehlt auch noch)

z.B:

```

var header =
(System.Windows.Controls.Primitives.DataGridColumnHeader)sender;
String headerToSort = (String)header.Column.Header;

IEnumerable<Team> teams = _Teams;

if (headerToSort != null &&
!string.IsNullOrEmpty(headerToSort))
{
    switch (headerToSort)
    {
        case "Name":
            teams.ToList().Sort(new TeamComparer());
            break;

```

Zudem muss noch darauf geachtet werden, die Sortierpfeile im Header zu entfernen

```
//Entfernen der Sortierpfeile im Header
foreach (var column in dGTeamView.Columns)
{
    column.SortDirection = null;
}
```

und nach erfolgter Sortierung wieder richtig anzubringen.

Es war leider aufgrund der Prüfungssituation nicht möglich dieses Feature noch so einzubauen, damit es den Qualitätsansprüchen genügt.

Serienbriefe

Es kann noch eine Funktion zum nachträglichen Bearbeiten von Serienbrief-Vorlagen implementiert werden.

In der Filter-Auswahl können angelegte Filter-Sets zudem momentan nach dem Anlegen nicht mehr bearbeitet werden.

Notizen

Bei den Notizen wäre es sinnvoll, den Benutzer mit zu erfassen, der die Notiz angelegt hat. Dies benötigt eine Änderung der Datenbank.

Kundenmodul

Die Datumsfelder ermöglichen momentan keinen schnellen Eingaben von weit zurück liegenden Daten, z.B. 1960.

Backup

Eine manuelle Backup-Funktion könnte noch implementiert werden.

Globale Stile

Die Standard-Schriftgröße 14 könnte noch über die Konfigurationsdatei veränderbar gemacht werden.

Buchhaltung

Die Funktion, für einen Kunden alle Einnahmen anzeigen zu lassen, ist noch nicht implementiert.

Bei den Schnellobuchungen könnte schneller von Feld zu Feld mit Tabulator gesprungen werden.

Beim Anlegen einer Buchung sollte man einen Kunden auswählen können, dem diese Buchung zugeordnet ist.

Login-Fenster

Das Benutzer-Bild wird momentan nicht geladen. Der Fehler scheint in
LoginViewModel.cs / Zeile 88 zu liegen:

```
set { SetValue(() => UserImageSource, value); }
```

Der Grund, warum dieser Befehl nicht erfolgreich durchgeführt werden kann, ist
unklar.