Buchungssystem für Gastronomen

Moritz Großmann

4. Februar 2018

Inhaltsverzeichnis

1 Einführung

Im Zeitalter der Digitalisierung steigen auch viele Gastronomische Einrichtungen auf ein Elektronisches Buchungssystem um. So haben Gastronomen einen besseren Überblick über ihre Finanzen und sind vor allem Kreditwürdiger, da sie ihre Einnahmen und Ausgaben besser Nachweisen können. Leider habe ich selbst die Erfahrung gemacht, dass die meisten Systeme sehr umständlich und vor allem funktionsarm sind. Gerade für Kellnerinnen und Kellner, welche sich nicht unbedingt mit Technik auskennen müssen, werden von diesem Systemen in ihrem Arbeiten ausgebremst.

Ich verfolge schon länger die Idee, die Grundlage für ein einfacheres System zu entwerfen. Ein System, welches vor allem Intuitiv ist.

2 Aufgabenstellung

2.1 Funktionen

2.1.1 Stammdatenverwaltung

Der Benutzer soll alle seine Ressourcen verwalten können. Hierfür soll es eine Stammdatenverwaltung geben, in der er seine Räume, Tisch, Waren und Warengruppen pflegen kann. Tische sollen Räumen untergeordnet sein, sowie Waren den Warengruppen. Außerdem soll der Benutzer bei den Warengruppen eine Baumstruktur anlegen können. So dass es zum Beispiel die Übergruppen "Getränkeünd SSpeißen"gibt und die anderen denen jeweils untergeordnet sind. Ein Raum soll aus einem Namen und einer Liste von Tischen bestehen. Ein Tisch soll aus einem Namen und einer Anzahl Sitzplätze bestehen. Dadurch kann später eine Auslastung des Raumes angezeigt werden. Eine Warengruppe soll aus einem Namen und einer Übergruppe bestehen, welche wiederum eine Warengruppe ist. Waren Haben einen Namen, einen Preis und eine Warengruppe deren sie angehören.

Alle Ressourcen können angelegt, geändert und gelöscht werden. Für eine spätere Erweiterbarkeit sollen die Objekte nicht aus der Datenbank gelöscht werden, sondern nur ein Flag gesetzt werden.

2.1.2 Buchen

Der Nutzer soll als Hauptmaske eine Übersicht der definierten Räume und deren Tische erhalten. Er soll sehen, wie viele Plätze und Tische in einem Raum frei sind, um eventuelle Kurzfristige Reservierungsanfragen ohne Probleme beantworten kann. Zu jedem Tisch soll der Gesamtbuchungsbetrag angezeigt werden und wann das letzte mal auf diesen Tisch gebucht wurde. So kann man vermeiden, dass zum Beispiel in einem größeren Restaurant ein Tisch aus versehen vernachlässigt wird.

Durch Anwählen eines Tisches sollen die Details zu diesem gezeigt werden. Man soll jede einzelne Buchung sehen, sowie wann diese gebucht wurde. In dieser Übersicht soll es möglich sein, Waren auf einen Tisch zu buchen. Der Benutzer soll im ganzen einen guten Überblick bekommen. Will ein Tisch getrennt bezahlen, kann der Benutzer einzelne Buchungen auswählen und sieht sofort deren Gesamtpreis. Der/Die Kellner/in muss den Betrag nicht im Kopf zusammen rechnen. Buchungen sollen ausgewählt und als Bezahlt oder Storniert markiert werden.

2.1.3 Tagesübersicht erstellen

Dem Nutzer soll es möglich sein, sich für jeden Tag eine Tagesübersicht ausgeben zu lassen. Hier werden alle Buchungen des Tages aufgelistet. Man sieht wie viele Buchungen es an diesem Tag gab und wie viele Stornierungen. Der Nutzer soll sofort den Tagesumsatz sehen können.

2.1.4 Webservice (Experimentell)

Mit dem Programm soll es möglich sein, einen Webservice zu starten. Über diesen können Kellner/innen über ein Client-Gerät, wie zum Beispiel mit einer Android-App, Daten in das System einspeisen. So können Buchungen direkt am Gast mit dem Smartphone getätigt werden. Der Service soll als REST-Service implemetiert werden, da sich Buchungen usw. gut als Ressourcen eignen. Diese Funktion habe ich von Anfang an als Experimentell erachtet und sehe es in dieser Version des Programmes nicht als "MUSS-Feature".

2.2 Design

Das Programm soll für eine Anwendungen mit Touch-Bildschirm, wie z.B. auf einem Tablet-PC oder einem Terminal, optimiert sein. Hierfür sind vor allem neben großen Schaltflächen auch eine gute Übersicht nötig. Der Nutzer darf in seinem Handeln nicht dadurch beeinträchtigt werden, dass er sich durch zu kleine Schaltflächen vertippt. Es soll ein dem Nutzer vertrautes Design verwendet werden, sodass es keine lange Eingewöhnungszeit gibt.

3 Implementierung

3.1 Teilprojekte

Die Solution ist in 3 Projekte aufgeteilt. Die App, die Domain und das Repository. In der App selbst ist das eigentliche Programm angesiedelt. Hier sind alle Views sowie ViewModels implementiert. Außerdem sind hier alle Konverter, welche Daten von einem ViewModel zur View, und umgekehrt, Konvertieren implementiert. In der Domain werden alle Models sowie das Interface zum Datenbankzugriff deklariert. Der Datenbankzugriff selber ist im Repository implementiert. Hier wird abgehandelt, wie Daten gespeichert werden. Durch die 3 Projekte soll eine Struktur entstehen, dass die App nur auf die Domain zugreifen kann. Die App greift auf die Datenbank nur über ein in der Domain deklariertes Interface auf das Repository zu. So wird realisiert, dass das Repository-Projekt ohne Probleme gegen ein anderes Ausgetauscht werden kann, welches das Interface zur Datenpersistierung implementiert.

3.2 Datenspeicherung

Zur Datenspeicherung wird EntityFramework 6.2.0 mit einer SQL-Server Compact Edition genutzt. So ist eine leichte Portabilität gewährleistet. Wird das Programm portiert, und es ist keine Datenbank auf dem neuen System vorhanden, wird eine neue generiert. Der Datenbankzugriff erfolgt ausschließlich über das Interface <IPersistBookingsystemData>. In diesem sind alle Funktionen deklariert, die zum Datenaustausch mit der Datenbank benötigt werden. Implementiert wird dieses Interface im Projekt <Repository> von der Klasse <BookingSystemDataPersistence>. Diese ist als eine Art Database-Helper anzusehen. Die Models, welche in der Domain deklariert sind, werden nicht direkt in der Datenbank gespeichert. Hierzu gibt es für jedes Model ein Datenbank-Pendant. Zum Arbeitern mit der Datenbank werden Models in der <BookingSystemDataPersistenc> zur Speicherung erst in das Datenbankmodel geparst und umgekehrt.

3.3 Benutzerschnittstelle

Die Benutzerschnittstelle wurde mit WPF implementiert. Hierzu wurde das MVVM-Pattern strikt durchgezogen. Einzig im <MainWindow> (einziges Fenster der Applikation) muss einmal Code-Behind verwendet werden um einen Daten-Kontext herzustellen. Ansonsten werden alle Interaktionen des Benutzers mit der Oberfläche durch ViewModels abgehandelt.

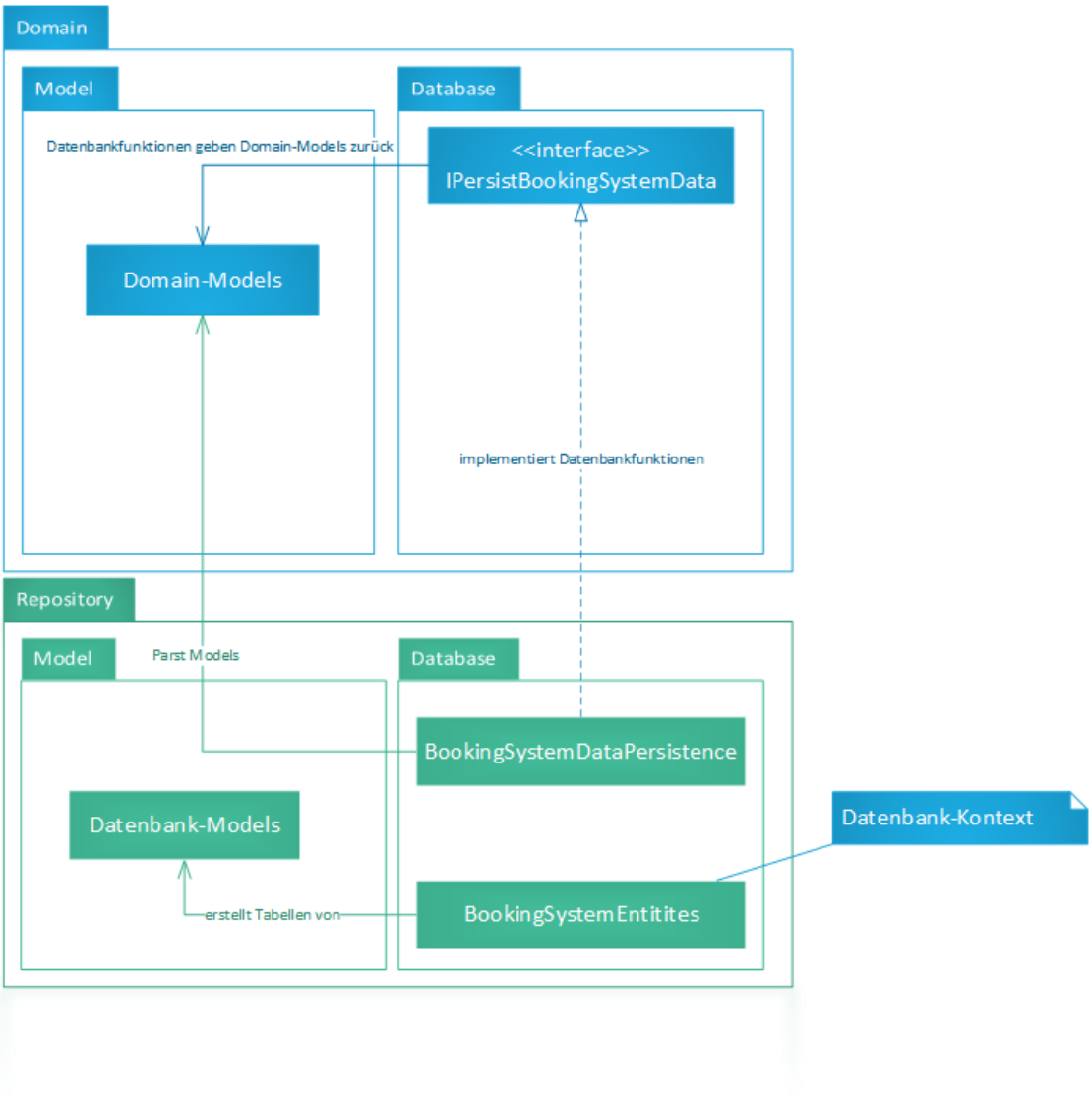

Abbildung 1: Zusammenhang zwischen Domain und Repository

Für das Layout wurde die Library MAHAPP.METRO verwendet. Dadurch wird leicht ein ansprechendes und flaches Design ermöglicht, welches dem Standarddesign von Windows ab Windows 8 stark ähnelt. Dadurch soll dem Nutzer ein Vertrautes Design vorgelegt werden.

Es wurde auf eine Menüleiste verzichtet, da diese zur Touch-Bedienung ungeeignet ist. Stattdessen wurde zur Navigation ein Hamburger-Menü genutzt.

3.3.1 Navigation zwischen verschiedenen Ansichten

Die Anzeige verschiedener Seiten im <MainWindow> wird ausschließlich durch UserControls umgesetzt. Hierfür wird eine bestimmte UserControl mit einem View-

Model verknüpft. Ändert sich in einem ViewModel eine Property, welche ein anderes ViewModel darstellt, ändert sich auch die View demensprechend. Als Beispiel: Das <MainWindow>, ausschnitt daraus zu sehen in Abbildung 3, ist mit dem <MainViewModel>, ausschnitt zu sehen in Abbildung 2, verknüpft. Das <MainViewModel> besitzt eine Property <CurrentViewModel> vom Typ <BaseViewModel>. Ist das BaseViewModel vom Typ <BaseDataManagementViewModel> wird im dafür vorgesehenen Bereich die <BaseDataManagementView> angezeigt. Ist das <BaseViewModel> ein <RoomListViewModel> wird die <RoomListView> angezeigt. Dies wird, auch verschachtelt, in diesem Projekt angewandt. So wird automatisch durch Austausch des ViewModels auch die View geändert.

MainViewModel.cs

```
1  internal class MainViewModel : BaseViewModel
2  {
3      private BaseViewModel _currentViewModel;
4
5      /// <summary>
6      /// Das Aktuell angezeigte ViewMdoel
7      /// </summary>
8      public BaseViewModel CurrentViewModel
9      {
10         get => _currentViewModel;
11         set => SetProperty(ref _currentViewModel, value, nameof(CurrentViewModel));
12     }
13
14     private void ToBaseData()
15     {
16         CurrentViewModel = new LoadingViewModel(); // Ladeansicht wird angezeigt
17         TaskAwaiter<BaseDataManagementViewModel> awaiter = GetBaseDataManagementViewModel().GetAwaiter();
18
19         awaiter.OnCompleted(() =>
20         {
21             CurrentViewModel = awaiter.GetResult(); //Stammdatenverwaltung wird angezeigtm wenn alle Daten geladen wurden
22         });
23
24     }
25
26     private Task<BaseDataManagementViewModel> GetBaseDataManagementViewModel()
27     {
28         //_bookingSystemDataPersistence = Kontext zur Datenbank
29         return Task.Run(() => new BaseDataManagementViewModel(_bookingSystemDataPersistence));
30     }
31 }
```

Abbildung 2: Ausschnitt aus dem MainViewModel

Beispiel

MainWindow.xaml

```
<controls:MetroWindow
    <Window.Resources>
        <!--Verknüpfung der ViewModel mit dfen Views!-->
        <DataTemplate DataType="{x:Type roomView:RoomListViewModel}">
            <booking:RoomListView />
        </DataTemplate>

        <DataTemplate DataType="{x:Type bookings:BookingsFromDayViewModel}">
            <dailyOverview:DailyOverview />
        </DataTemplate>

        <DataTemplate DataType="{x:Type baseDataManagement:BaseDataManagementViewModel}">
            <baseDataManagementView:BaseDataManagementView />
        </DataTemplate>

        <DataTemplate DataType="{x:Type loading:LoadingViewModel}">
            <pages:LoadingScreen />
        </DataTemplate>

    <Grid>
        <!--Anzeige der View, welches mit dem ViewModel, welches das CurrentViewModel darstellt, verknüpft ist!-->
        <ContentControl Content="{Binding CurrentViewModel}" />
    </Grid>
</controls:MetroWindow>
```

Abbildung 3: Ausschnitt aus dem MainWindow

4 Benutzerhandbuch

4.1 Stammdatenverwaltung

4.1.1 Die Stammdatenverwaltung aufrufen

Die Stammdatenverwaltung rufen sie auf, indem Sie im Menü auf der rechten Seite auf das Schraubenschlüssel-Symbol klicken. Alternativ können sie auch das Menu ausklappen. Nun erscheint neben den Symbolen auch die Beschriftung.

4.1.2 Räume bearbeiten

In der Raum-Ansicht sehen sie auf der linken Seite eine Liste mit allen Räumen, die sie bereits definiert haben. Wenn sie einen neuen Raum anlegen möchten, Klicken sie auf den Button mit dem Plus-Symbol über der Raum-Liste. Nun öffnet sich auf der rechten Seite ein Formular. Tragen sie hier den Name des Raumes in das vorgesehene Textfeld ein. Klicken sie nun auf den Disketten-Button. Die Ansicht des Formulars ändert sich nun von der Bearbeitungs- auf die Leseansicht.

Um einen Raum zu editieren, klicken sie auf den Bearbeiten-Button unter dem Formular. Dadurch wechselt die Lese- auf die Editieransicht. Das Editieren schließen sie ab, indem sie mit dem Disketten-Button Speichern.

Um einen nicht mehr benötigten Raum zu löschen, wählen sie in der Raumliste einen Raum an und klicken Sie auf den Mülltonnen-Button unter dem Eingabeformular. Es öffnet sich ein Dialog, in dem sie das Löschen bestätigen müssen. Sie können keinen Raum löschen, welcher einen Tisch mit offenen Buchungen enthält.

Nach dem Speichern des Raumes können sie diesem Tische hinzufügen. Dies funktioniert auf die gleiche Art und Weiße wie bei einem Raum. Klicken sie auf das Plus-Symbol über der Tischliste, welche noch leer ist. Im Formular müssen sie einen Namen für den Tisch vergeben, sowie eine Anzahl von Sitzplätzen. Dies ist später im Betriebt wichtig, um einen schnellen Überblick über die Freien Tische und Plätze zu bekommen. Um den Tisch zu speichern klicken sie auf das Disketten-Symbol. Die Editieransicht wechselt dadurch auf die Leseansicht.

Um einen Tisch zu editieren, wählen sie einen Raum an und klicken sie dann auf das Edititeren-Symbol unter des Eingabeformulars. Das Editieren schließen sie durch Speichern mit einem klick auf den Disketten-Button ab.

Um einen nicht mehr benötigten Tisch zu löschen, wählen sie in der Tischliste einen Tisch an und klicken Sie auf den Mülltonnen-Button unter dem Eingabeformular. Es öffnet sich ein Dialog, in dem sie das Löschen bestätigen müssen. Sie können keinen Tisch löschen, auf dem sich offene Buchungen befinden.

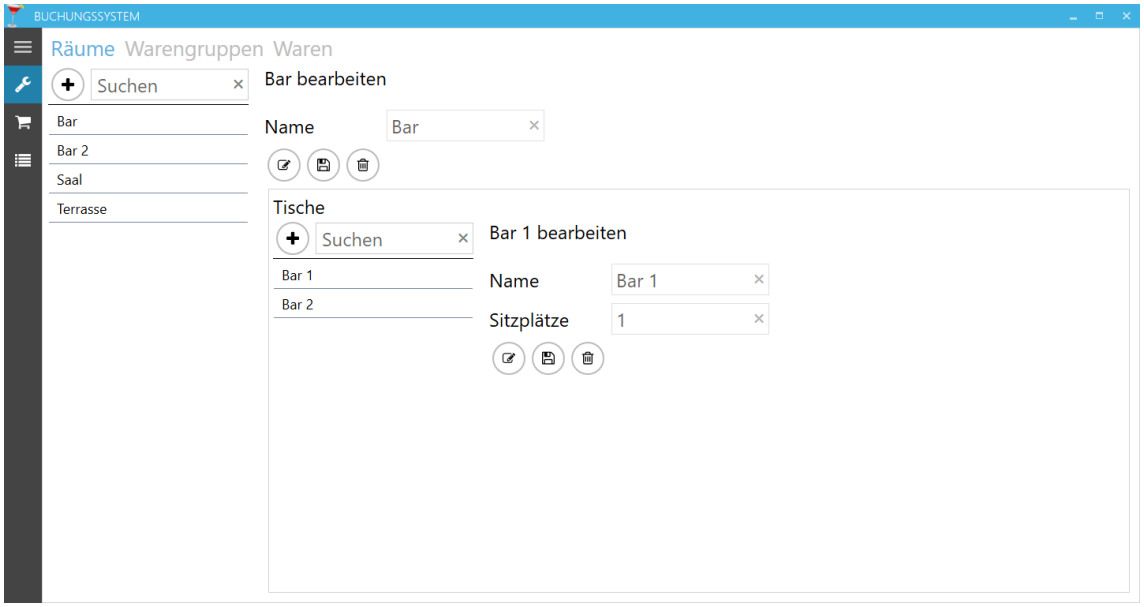

Abbildung 4: Übersicht der Raumverwaltung

4.1.3 Warengruppen bearbeiten

Um die Oberfläche für die Warengruppen aufzurufen, klicken sie in der Stammdatenverwaltung oben auf den Tab Warengruppen. Auf der linken Seite sehen sie eine Liste mit allen definierten Warengruppen.

Wenn sie eine neue Warengruppe anlegen möchten, klicken sie auf den Plus-Button über der Warengruppenliste. Nun öffnet sie das Eingabeformular. Hier müssen sie einen Namen vergeben, sowie die übergeordnete Warengruppe angeben. So können sie eine Baumstruktur von Warengruppen erzeugen. Soll ihre Warengruppe keine Übergruppe haben, stellen sie den Schalter bei 'Keine Übergruppe' auf 'Ja'. Zum Abschluss klicken sie auf den Disketten-Button um die Warengruppe zu speichern.

Um eine Warengruppe zu Editieren, wählen sie in der Warengruppenliste eine Warengruppe aus. Öffnen sie die Editieransicht durch Klick auf den Editieren-Button. Sie können den Namen sowie die Übergruppe ändern. Beachten sie, wenn sie die Übergruppe ändern, werden auch alle Waren, welche sich in der Warengruppe befinden, verschoben.

Um eine nicht mehr benötigte Warengruppe löschen, wählen sie in der Warengruppenliste eine Warengruppe an und klicken Sie auf den Mülltonnen-Button unter dem Eingabeformular. Es öffnet sich ein Dialog, in dem sie das Löschen bestätigen müssen.

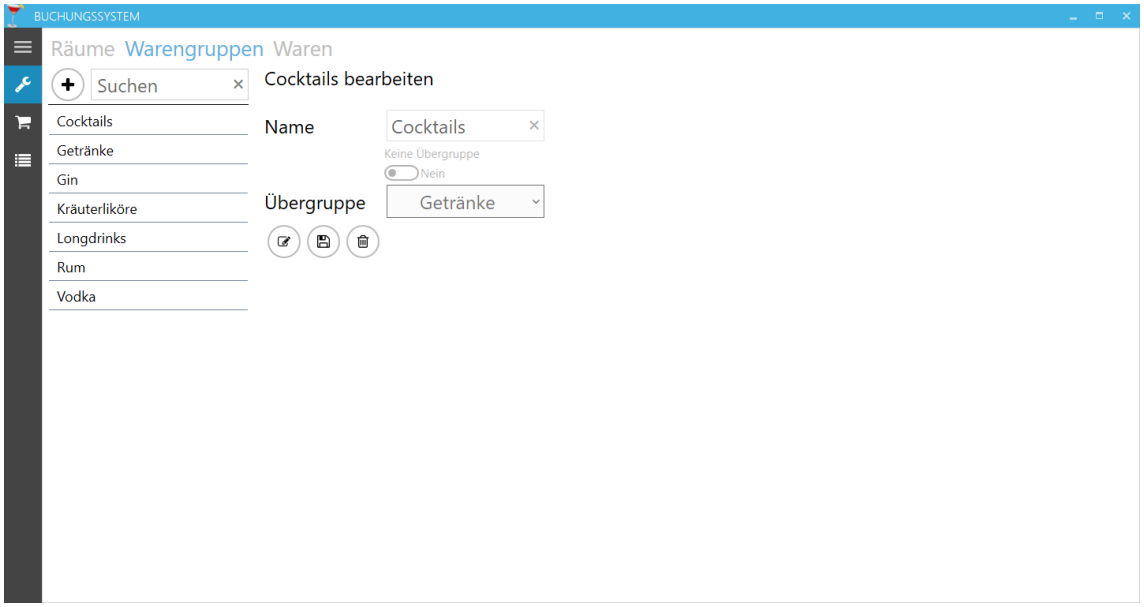

Abbildung 5: Übersicht der Warengruppenverwaltung

4.1.4 Waren bearbeiten

Um die Oberfläche für die Waren aufzurufen, klicken sie in der Stammdatenverwaltung oben auf den Tab Waren. Auf der linken Seite sehen sie eine Liste mit allen definierten Waren.

Wenn sie eine Ware anlegen möchten, klicken sie auf den Plus-Button über der Warenliste. Nun öffnet sie das Eingabeformular. Hier müssen sie einen Namen vergeben,einen Preis, sowie die übergeordnete Warengruppe angeben. Sie können nur Warengruppen auswählen, die keine weiteren Warengruppen enthalten. Zum Abschluss klicken sie auf den Disketten-Button um die Ware zu speichern.

Um eine Ware zu Editieren, wählen sie in der Warenliste eine Ware aus. Öffnen sie die Editieransicht durch Klick auf den Editieren-Button. Sie können den Namen, den Preis und die Übergruppe ändern.

Um eine nicht mehr benötigte Ware zu löschen, wählen sie in der Warenliste eine Ware aus und Klicken Sie auf den Mülltonnen-Button. Es öffnet sich ein Dialog, in dem sie das Löschen bestätigen müssen.

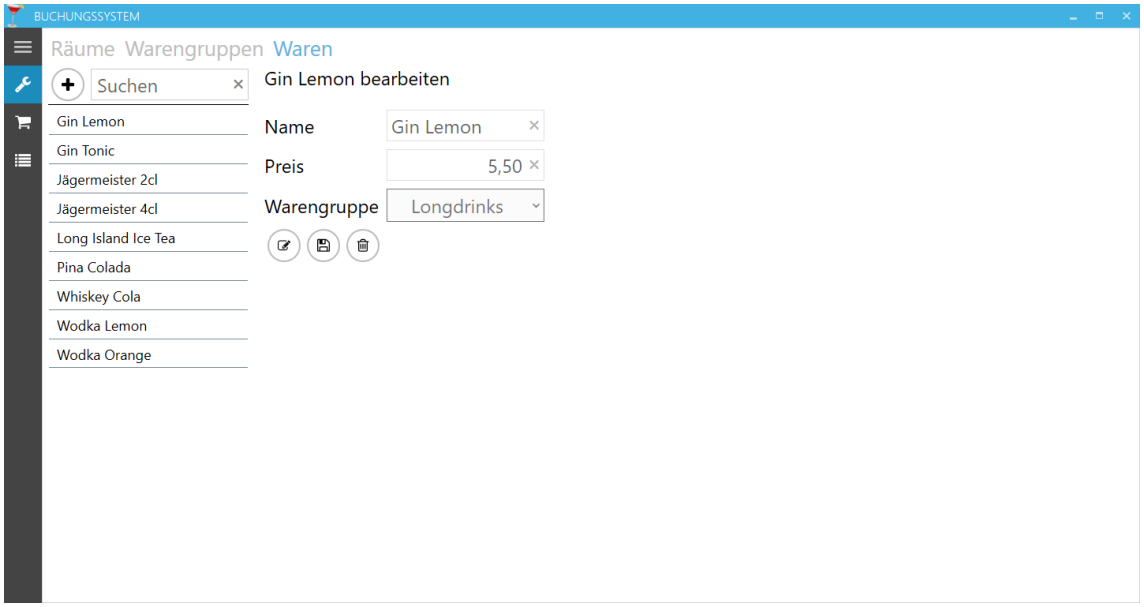

Abbildung 6: Übersicht der Warenverwaltung

4.2 Buchungsverwaltung

4.3 Die Buchungsverwaltung aufrufen

Die Buchungsverwaltung rufen Sie auf, indem Sie im Menü auf der linken Seite auf das Einkaufswagen-Symbol klicken. Alternativ können sie auch das Menu ausklappen. Nun erscheint neben den Symbolen auch die Beschriftung. Die Buchungsverwaltung öffnet sich automatisch beim Start des Programmes. Hier finden Sie eine Übersicht aller Räume mit deren Tischen. In der Fußzeile sehen sie, wie viele Tische, und Plätze in dem ausgewähltem Raum Frei sind.

4.4 Status eines Tisches ändern

Um einen Tisch zu besetzen, klicken sie mit der rechten Maustaste (bei einem Touch-Gerät alternativ lang mit dem Finger) um das Kontext-Menü zu öffen. Klicken Sie nun auf 'Status Ändern'. Der Tisch wird nun als Besetzt markiert. Auf die gleiche Weise machen Sie die Rückgangig.

4.4.1 Ein Ware auswählen und Buchen

Wählen durch Klick einen Tische an um die Tischübersicht zu sehen. Im linken Teil sehen sie die Warenauswahl. Im rechten die Buchungen auf diesem Tisch. Wählen sie in der linken Liste durch Klick eine Ware aus. Sie können beliebig viele Waren

Abbildung 7: Raumübersicht

auswählen. Diese werden in einer Liste daneben zwischengespeichert. Um eine Ware zu deselektieren, klicken sie in der Liste der Ausgewählten Waren auf diese. Um die ausgewählten Waren zu Buchen, klicken sie auf den 'Buchen'-Button unten links. Nun erscheint in der Liste der Buchungen für jede Ware eine Buchung.

In der Fußzeile der Buchungen sehen die den Gesamtbetrag, welcher auf dem Tisch gebucht ist.

4.4.2 Buchungen als Bezahlt oder Storniert markieren

Um Buchungen abzuschließen, wählen sie in der Tischübersicht eine oder mehrere Buchungen durch einen Klick auf diese aus. Diese werden dann in die Liste 'Ausgewählte Buchungen verschoben'. Diese können sie wiederum durch Klick darauf deselektieren. Unter dieser Liste wird der Gesamtbetrag der ausgewählten Buchungen angezeigt. Klicken sie zum Schluss auf den Button 'bezahlen' um die Buchungen zu Bezahlen oder auf 'Stornieren' um die Buchungen zu Stornieren.

4.5 Tagesübersicht

Die Tagesübersicht rufen Sie auf, indem Sie im Menü auf der linken Seite auf das Listen-Symbol klicken. Alternativ können sie auch das Menu ausklappen. Nun erscheint neben den Symbolen auch die Beschriftung.

Hier können sie durch klicken auf den Kalender ein Datum auswählen. Haben

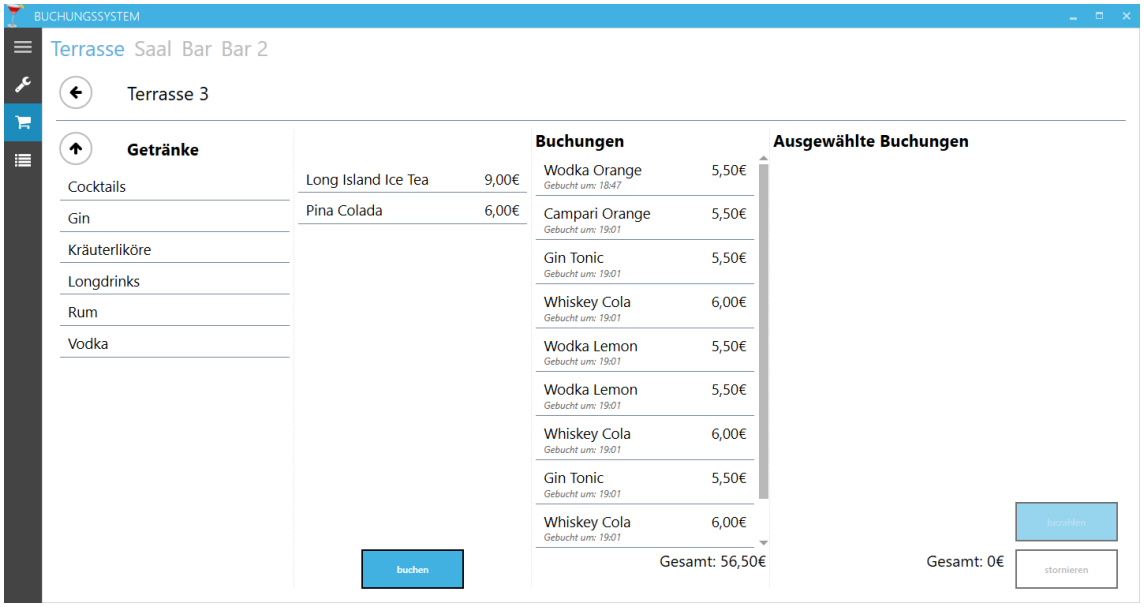

Abbildung 8: Tischübersicht

Sie ein Daten ausgewählt, werden alle Buchungen von diesem Tag angezeigt. In der Fußzeile ist der Tagesumsatz zu sehen. Beachten Sie, dass Offene und Stornierte Buchungen nicht in den Tagesumsatz fließen. Sie können die Liste durch Klick auf das Entsprechende Attribut sortieren.

5 Diskussionen

5.1 Fazit

Die Kernfunktionen des Projektes wurden Implementiert und getestet. Einzig der Webservice wurde aus Zeitgründen nicht implementiert. Für einen Produktiven Einsatz müssen allerdings noch einige Funktionen folgen. Ein paar davon sind im nächsten Abschnitt zu lesen.

5.2 Mögliche Erweiterungen

Das Programm wurde für eine möglichst große Erweiterbarkeit programmiert. Beim Löschen von Objekten wird nur ein Flag in der Datenbank gesetzt. Die Daten werden nicht wirklich gelöscht. So könnte man zum Beispiel in der Stammdatenverwaltung ein Papierkorb-System implementieren.

Der Oben genannte Webservice wäre eine wichitge Erweiterung.

Auch ein Mehrbenutzer-Betrieb ist denkbar. So könnte zum Beispiel ein Kellner/in auf dem Clientgerät angemeldet sein. Dadurch könnte man nachverfolgen, von welchem Kellner welche Buchung ist.

Dies waren nur ein paar Möglichkeiten. Möglich und Denkbar ist noch viel mehr.

6 Anhang

6.1 Verwendeter fremder Quelltext

Die Klassen BaseViewModel und RelayCommand im Namespace Buchungssystem.App.ViewModel.Ba
wurden aus den Folien der Vorlesung entnommen (49 Codezeilen). Des weiteren wurde im Namespace Buchungssystem.App.Properties sowie Buchungssystem.Domain.Properties von Re-Sharper jeweils eine 205-Zeilige Annotations-Klasse angelegt. Diese sind ebenfalls nicht von mir. Im Namespace Buchungssystem.App.Util ist die Klasse GridViewSort der Website https://www.thomaslevesque.com/2009/03/27/wpf-automatically-sort-a-gridview-when-a-column-header-is-clicked/ entnommen.

6.2 Anzahl Codezeilen

C#	2198
Fremdcode C#	-516
XAML	1684
Gesamt	3336

7 Eidesstattliche Erklärung

Hiermit erkläre ich, Moritz Großmann, dass ich die vorliegende Studienarbeit im Modul .NET-Programmierung mit C# selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Studienarbeit, die anderen Quellen im Wortlaut (ebenfalls Quellcode betreffend) nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht.