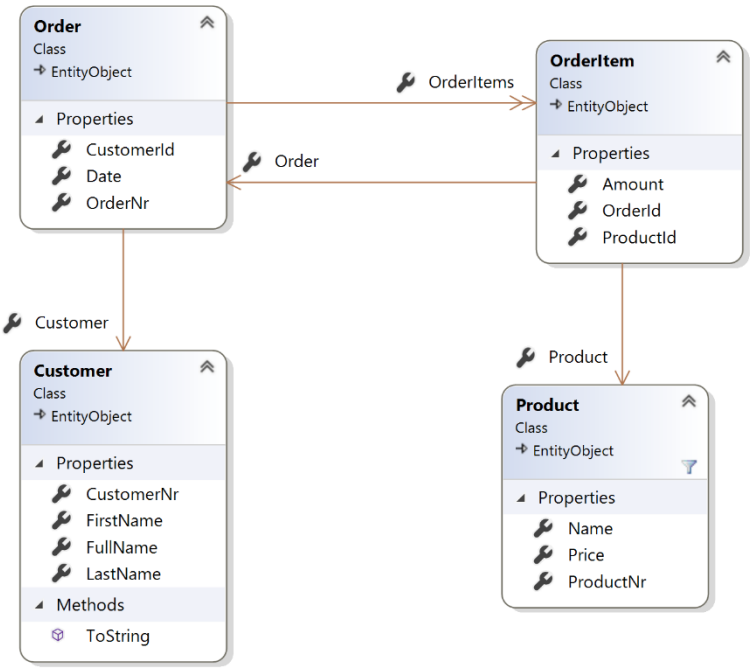
Bakery-Shop

Es soll eine Software für eine Bäckerei implementiert werden, die die Kundenbestellungen verwaltet. Eine Kundenbestellung (***Order***) ist einem Kunden (***Customer***) zugeordnet und besteht aus mehreren Bestellpositionen (***OrderItems***), wobei eine Position ein bestimmtes Produkt (***Product***) mit einer bestimmten Anzahl darstellt.



Aus den beiden vorgegebenen csv-Files ***Products.csv*** und ***OrderItems.csv*** sollen die Daten in vier Entitäten (Products, Customers, Orders und OrderItems) eingelesen und in eine SQL-Datenbank (mittels Code-First) übertragen werden.

In der Datei OrderItems.csv sind die Bestellpositionen aller Bestellungen enthalten.

Kundendaten und Bestelldaten sind in OrderItems.csv redundant (mehrfach) je Position eingetragen. Alle vorkommenden Kunden sollen (natürlich nur je einmal) in der Entität Customers gespeichert werden und je vorkommender Bestellung soll ein Datensatz in der Entität Orders erstellt werden. (Feldbezeichnungen lt. Überschrift in den csv-Files).

Die Datei Products.csv enthält alle Produkte, die zu importieren sind (auch solche, die noch keine Bestellungen aufweisen).

ImportConsole:

Wer den Import überspringen will, kann das beiliegende Batch-File Import/Import.csv ausführen

```
Import der Bäckerei-Daten in die Datenbank
Datenbank löschen
Datenbank migrieren
Daten werden von csv-Dateien eingelesen
  12 Produkte eingelesen
Produkte werden in Datenbank gespeichert (persistiert)
  Es wurden 12 Produkte gespeichert!
  Es wurden 4 Kunden gespeichert!
  Es wurden 8 Bestellungen gespeichert!
  Es wurden 14 Bestellpositionen gespeichert!

Beenden mit Eingabetaste ...
```

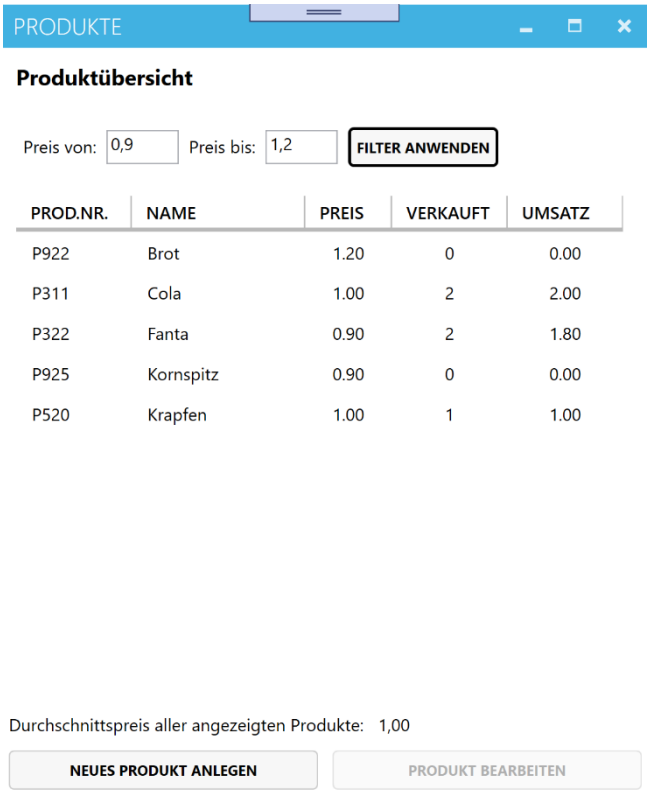
WPF mit MVVM

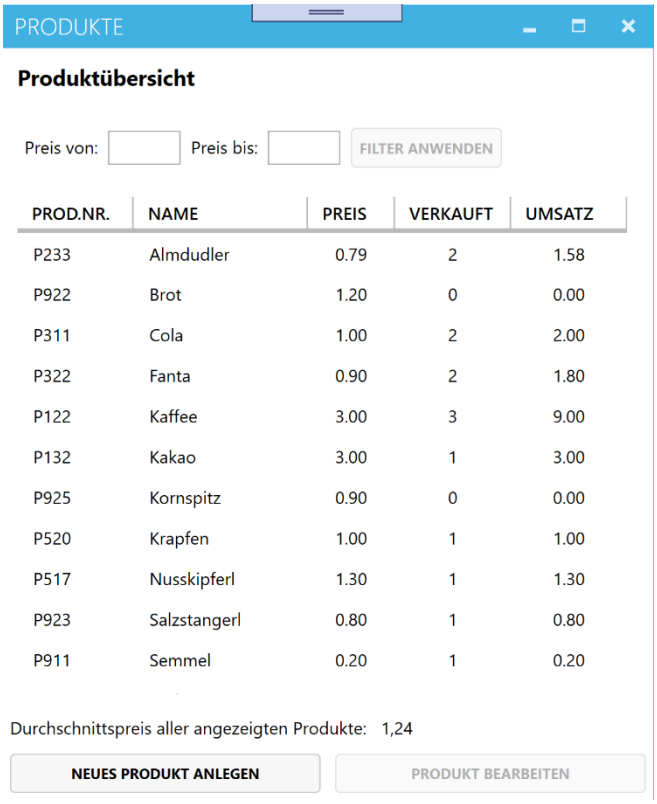
Die Produkte sollen in einer WPF-Applikation bearbeitet werden können. Weiters sollen neue Produkte angelegt und bestehende Produkte editiert werden können.

Startmaske: Produktübersicht

In der Startmaske werden alle Produkte in einer Liste dargestellt, inklusive der jeweiligen Verkaufszahlen laut Bestellungen.

Diese Liste kann nach „Preis von“ und „Preis bis“ gefiltert werden. Der Filter wird angewendet, sobald der entsprechende Button gedrückt wird. Dieser Button „Filter anwenden“ soll nur aktiviert sein, wenn ein gültiger Preis oder kein Preis in einem oder beiden Filterfeldern eingetragen ist. (D.h. bei ungültiger Eingabe z.B. „abc“ soll der Button deaktiviert werden -> nach jedem eingegebenen Zeichen prüfen!)



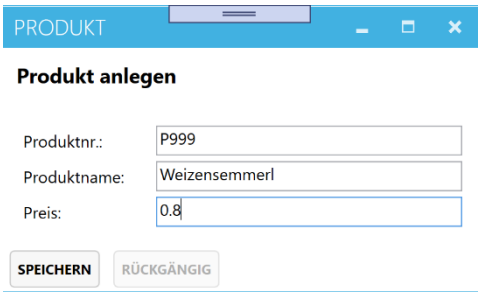


Wenn ein Filterfeld keinen Text beinhaltet, so gilt der Wertebereich in diese Richtung als unbeschränkt.

Unterhalb der Liste wird der Durchschnittspreis aller angezeigten Produkte ausgegeben. (Je nach Filter!)
Wieder darunter kann mittels zwei Buttons entweder ein neues Produkt angelegt oder das aktuell ausgewählte Produkt in einer Detailsmaske bearbeitet werden. Der „Bearbeiten“-Button ist nur aktiviert, wenn ein Datensatz ausgewählt ist.

Detailmaske: Neues Produkt anlegen / Produkt bearbeiten

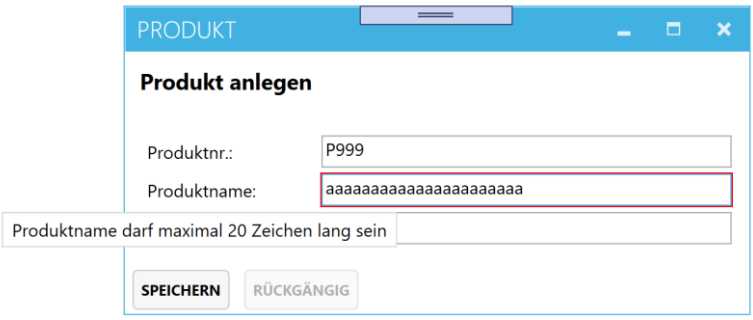
Tipp: Es wird empfohlen, für beide Funktionen nur eine Maske und ein ViewModel zu verwenden! (Überschrift entsprechend anpassen!)



Validierung in WPF

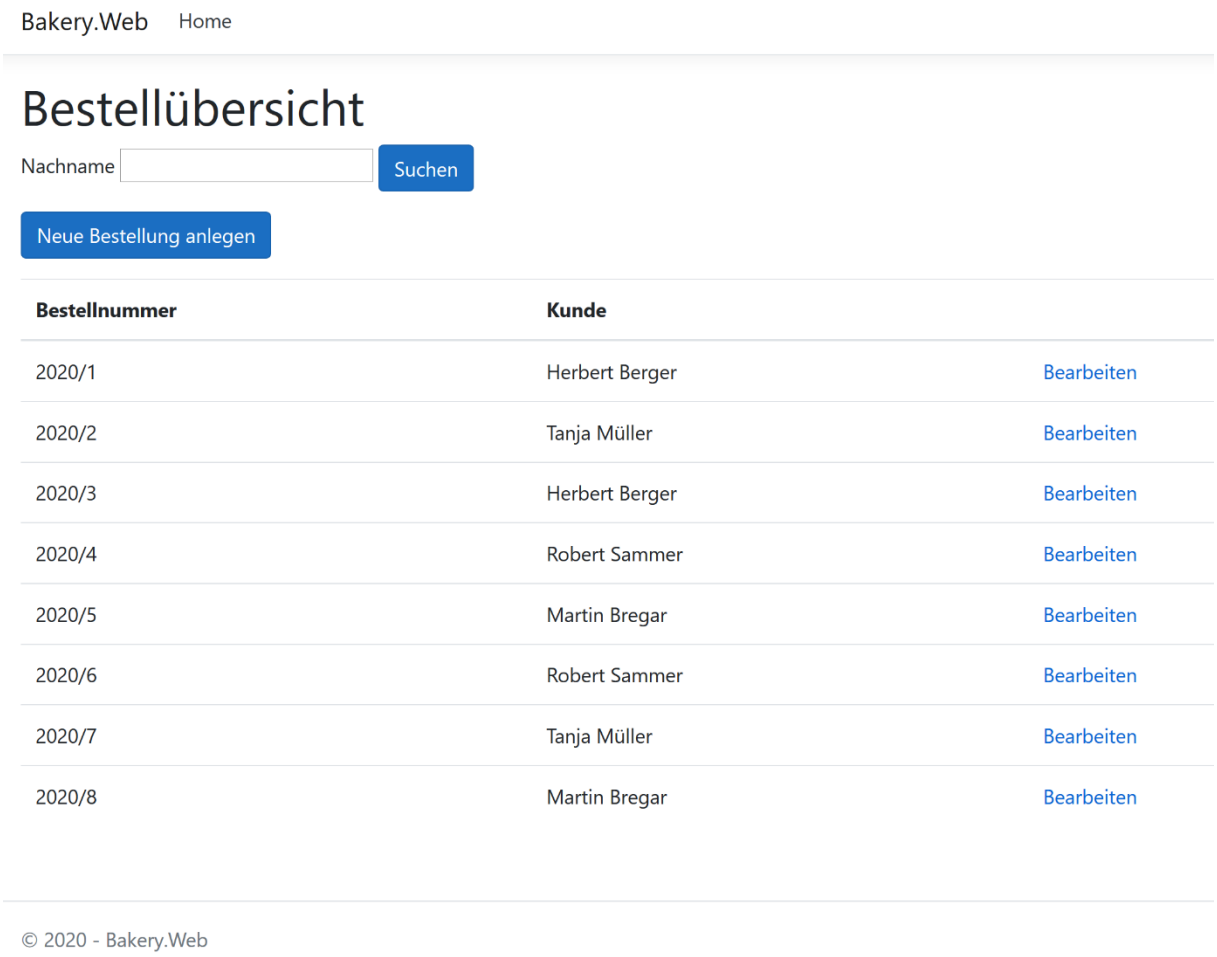
Validierungsmeldungen beim Speichern (z.B. Name existiert bereits) sind am Formular auszugeben.

Die Prüfung, ob der Produktname lang bzw. kurz (Mind. 1, maximal 20 Zeichen) genug ist, soll mittels WPF-Validierung bereits vor Speicherung direkt auf dem WPF-Fenster dargestellt werden!



MVC

In einer Razor Page-Webanwendung sollen Bestellungen verwaltet werden können. In der Startmaske werden alle Bestellungen sortiert nach der Bestellnummer in einer Liste dargestellt.



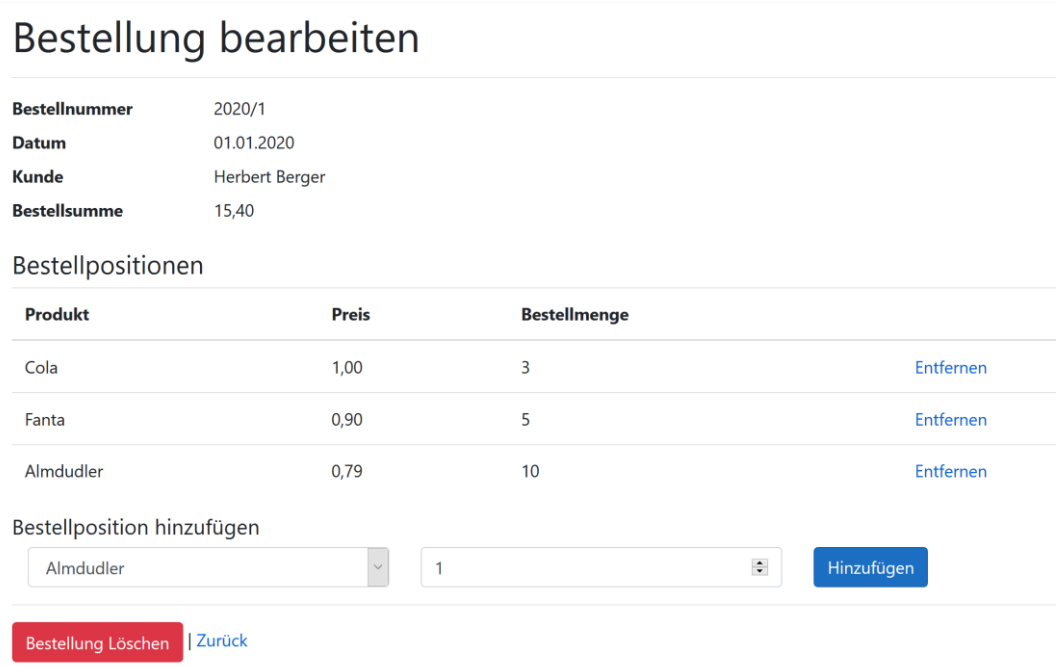
Neben jeder Bestellung soll ein Link zum Bearbeiten (inkl. Löschen) der jeweiligen Bestellung platziert werden, außerdem soll auf der Startmaske ein Link zum Neuanlegen einer Bestellung verfügbar sein.

Im oberen Bereich soll nach einem Nachnamen gesucht werden können. Dazu kann ein Teil eines Nachnamens eingegeben werden, wenn auf den „Suchen“-Button geklickt wird, wird die Liste entsprechend aktualisiert:

Beim Anlegen einer neuen Bestellung soll das Datum mit dem heutigen Tag vorbelegt sein und das Dropdown mit allen Kunden befüllt sein.

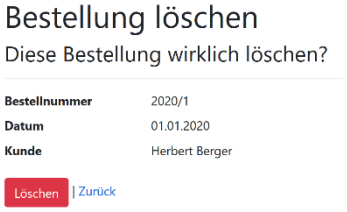
- Die Bestellnummer muss angegeben sein (client-seitige Validierung).
- Die Eingabe der Bestellnummer ist gegen die Datenbank zu validieren, da die Bestellnummer eindeutig sein muss.

Bearbeiten/Löschen einer Bestellung



In der Detailmaske zu einer Bestellung werden alle Bestelldaten inklusive der Bestellsumme über alle Bestellpositionen angezeigt (read-only). Weiters können beliebig viele Bestellpositionen hinzugefügt und gelöscht werden!

Vor dem endgültigen Löschen einer Bestellung soll noch eine Sicherheitsrückfrage angezeigt werden:



REST-Abfragen

Es soll zusätzlich die Möglichkeit geschaffen werden, folgende Daten per Rest-Services abfragen zu können:

- alle Bestellungen (/api/orders)
- alle Bestellungen zu einem bestimmten Kunden (KundeId wird übergeben) (/api/orders/ordersByCustomerId/5)

Request URL

```
http://localhost:64535/api/Orders
```

Server response

Code	Details
200	**Response body**


```
[
  {
    "orderedItems": [
      {
        "id": 1,
        "productName": "Cola",
        "productPrice": "1,00",
        "amount": 3
      },
      {
        "id": 2,
        "productName": "Fanta",
        "productPrice": "0,90",
        "amount": 5
      },
      {
        "id": 3,
        "productName": "Almdudler",
        "productPrice": "0,79",
        "amount": 10
      }
    ],
    "sales": 15.4,
    "orderId": 1,
    "orderNr": "2020/1",
    "date": "2020-01-01T00:00:00",
    "customerName": "Herbert Berger"
  },
  {
```


Request URL

```
http://localhost:64535/api/Orders/ordersByCustomerId/4
```

Server response

Code	Details
200	**Response body**


```
[
  {
    "orderedItems": [
      {
        "id": 13,
        "productName": "Kaffee",
        "productPrice": "3,00",
        "amount": 8
      }
    ],
    "sales": 24,
    "orderId": 7,
    "orderNr": "2020/5",
    "date": "2020-01-04T00:00:00",
    "customerName": "Martin Bregar"
  },
  {
    "orderedItems": [
      {
        "id": 14,
        "productName": "Kaffee",
        "productPrice": "3,00",
        "amount": 9
      }
    ],
    "sales": 27,
    "orderId": 8,
    "orderNr": "2020/8",
    "date": "2020-01-09T00:00:00",
    "customerName": "Martin Bregar"
```