Tatyana Merlo – Natalie Stalder – Nadja Stadelmann

Starbucks Manager

M120 – Benutzerschnittstelle entwickeln

# Einleitung

In einem vorherigen Projekt haben wir für Starbucks ein Menümanager geschrieben. Die Mitarbeiter können Produkte zum Menu hinzufügen, bearbeiten und wieder löschen. Weiter wird das Menü natürlich auch für die Kunden angezeigt.

Bisher wurde kein GUI eingesetzt, weshalb für die Klienten von Starbucks ein eher unschönes Menü angezeigt wurde. Weiter konnte das Programm nur über die Konsole bearbeitet werden, sodass eine spezielle Mitarbeiterschulung für den Gebrauch des Starbucksmanagers nötig war.

Dies wollen wir in diesem Folgeprojekt nun ändern.

## Dokument

### Aufbau

* Analyse
  + Beschrieb und erste Aufsplittung in einzelne Teilbereiche
* Vorgehen
  + Abhängigkeiten, Aufgabenübersicht
* Design
  + Detaillierte Analyse DB-Design
  + Detaillierte Analyse Programmstruktur
* Implementation
* Testing
  + Testcases basieren auf UseCases

Inhalt

[Einleitung 1](#_Toc496595033)

[Dokument 1](#_Toc496595034)

[Aufbau 1](#_Toc496595035)

[Vision 3](#_Toc496595036)

[Analyse 3](#_Toc496595037)

[Anforderungen 3](#_Toc496595038)

[Muss-Kriterien 3](#_Toc496595039)

[Kann-Kriterien 3](#_Toc496595040)

[Use Cases 3](#_Toc496595041)

[Design 4](#_Toc496595042)

[Filestruktur 4](#_Toc496595043)

[Programm 4](#_Toc496595044)

[Model 4](#_Toc496595045)

[View 4](#_Toc496595046)

[Controller 4](#_Toc496595047)

[Implementation 5](#_Toc496595048)

[Implementierung 5](#_Toc496595049)

[Testing 6](#_Toc496595050)

# Vision

Im Modul 326 haben wir ein Starbucks-Menümanager mit Java erstellt. Bisher konnte der jedoch nur textbasiert verwendet werden (mittels Konsole).

Unser Ziel ist nun, den Mitarbeitern das Mutieren des Menüs mit einem GUI zu erleichtern. Das GUI soll alle bisherigen Userinteraktionen abdecken, sprich:

* Menü anzeigen
* Produkt hinzufügen
* Produkt bearbeiten
* Produkt löschen
* Programm beenden

Weiter möchten wir eine Login- sowie Logoutfunktion einbauen, um die Sicherheit der Daten zu gewährleisten.

# Analyse

## Anforderungen

### Muss-Kriterien

1. Menüanzeige für die Kunden
   1. an Starbucksdesign angepasst
   2. gegliedert nach Produktkategorien
2. Menüanzeige für den Mitarbeiter
   1. Edit-Funktion zu jedem Produkt
   2. Delete-Funktion zu jedem Produkt
   3. selbe Gliederung wie bei der Kundenansicht
   4. Überbegriffe der einzelnen Felder sichtbar (bsp. Price)
3. Add-Funktion
   1. eigene Ansicht
   2. Auswahl der Produktkategorie
   3. Eingabe der Produktinformationen
      1. nur die relevanten Infos müssen eingegeben werden
4. Edit-Funktion
   1. Item bearbeiten
5. Delete-Funktion
   1. Item löschen
6. Layout
   1. Einheitlich
   2. Schlicht
   3. Gemäss Kapitel sowieso
7. Error-Handling
   1. genaue Informationen für den User

### Kann-Kriterien

1. Preis-Eingabe
   1. Dialogfenster mit Taschenrechnerlayout
2. Add-Funktion
   1. Speicherung relevante Daten beim Wechsel der Produktkategorie
3. Edit-Funktion
   1. Produktkategorie anpassen können
4. Help-Funktion
   1. Beschrieb der Funktionen
   2. Beschrieb des Files product.txt
5. Error-Handling
   1. Dialogfenster
      1. Message
      2. OK-Button als Userbestätigung

## Use Cases

# Design

## Filestruktur

Der bisherige Starbucksmanager hat keine Datenbank, welche er verwaltet. Die Daten werden in einem einfachen File abgespeichert. Dennoch ist die Struktur des Files natürlich extrem wichtig für die Datenverarbeitung, weshalb sie hier kurz deklariert ist.

Jede Zeile steht für einen Menü-Eintrag, der in einzelne Komponenten unterteilt ist. Je nachdem, welche Attribute der Eintrag mitführt, wird er in eine andere Kategorie unterteilt.

**Kategorie Recordstruktur**

Beverage Name | Preis | null | true (hot) / false (cold)  
beverage¦2.3¦null¦true

Coffee Name | Preis |Zutaten  
coffee¦1.4¦ingr

Extra Name | Preis  
extra¦2.0

Food Name | Preis |Zutaten | Vegi / Vegan / Glutenfrei / usw.  
food¦1.5¦Ingr¦Vegi

## Programm

### Model

Wie bereits erwähnt werden die Daten in einem einzigen File gehandhabt. Das File und sein Handling besteht bereits. Dennoch müssen am Datenmodel noch einiges angepasst werden, um die korrekte Zusammenarbeit mit dem GUI sicherzustellen.

### View

Das GUI ist komplett neu umzusetzen. Bisher waren nur Kommandozeileneingaben zur Steuerung des Programmes da. Dies soll neu über ein GUI geregelt werden, um den Mitarbeitern einen leichteren Einstieg zu ermöglichen.

So stellen wir uns das Ergebnis des GUIs vor:

Die Kundenansicht soll dem Starbucksdesign angepasst sein. In ihrem Logo ist ein spezielles Grün zu finden. Weiter zeigt es eine Weisse Figur auf schwarzem Hintergrund. Um den Kunden eine möglichst angenehme Ansicht bieten zu können, wird für das GUI das Grün als Hintergrund eingesetzt. Darauf positionieren werden die Artikel in weisser, serifenloser Schrift. Diese garantiert einen guten Lesefluss. Um dem Kunden die Auswahl möglichst zu vereinfachen, werden die Produkte in vier Kategorien unterteilt. Die Kaffees sind zu Oberst zu finden, da die meisten Kunden Starbucks wegen diesen Produkten besuchen. Danach folgen die Beverages (weitere Getränke), die Extras und zum Schluss noch die Muffins und Cookies unter der Kategorie Food.

Für die Mitarbeiteransicht wollen wir ein einheitliches Layout erstellen, damit der User sich schnell auf allen Seiten zurechtfinden. Es soll ausserdem übersichtlich sein, das heisst für uns, dass der User immer nur das angezeigt bekommt, was er wirklich braucht.

Für all diese Views verwenden wir CSS für JavaFX, um die verschiedenen Teile des GUIs einheitlich gestalten zu können.

### Controller

Die Controller bestehen teilweise, da bereits Usereingaben verarbeitet wurden. Dennoch müssen sie an das GUI angepasst werden.

# Implementation

## Implementierung

# Testing