Simulator (MC)

Thema: Java

Dokumentinformationen

Dateiname: McSimulator

Speicherdatum: 30.01.2022

Autoreninformationen

Autor: Alessio, Dominic

E-Mail: -

Tel: -

Inhaltsverzeichnis

[1 Einleitung 3](#_Toc94478717)

[1.1 Sinn und Zweck 3](#_Toc94478718)

[2 IPERKA 4](#_Toc94478719)

[2.1 Informieren 4](#_Toc94478720)

[2.2 Planen & Entscheiden 4](#_Toc94478721)

[2.3 Realisieren 5](#_Toc94478722)

[2.4 Kontrollieren 6](#_Toc94478723)

[2.5 Auswerten 7](#_Toc94478724)

# Einleitung

## Sinn und Zweck

In unserem Projekt geht es um ein Programm, welches uns ermöglich einen Tag als Mitarbeiter bei McDonalds zu simulieren. Es ist möglich Burger zusammenzubauen und das Ganze in vielen verschiedenen Arten. Das Ganze zahlt sich in Punkten aus. Das Ziel ist es so viele Punkte wie möglich zu holen. Durch einen Burger mit den richtigen Zutaten werden dir Punkte zugeschrieben, wenn du jedoch falsche Zutaten für den Kunden zusammenstellst, werden dir Punkte von deinem Konto abgezogen.

# IPERKA

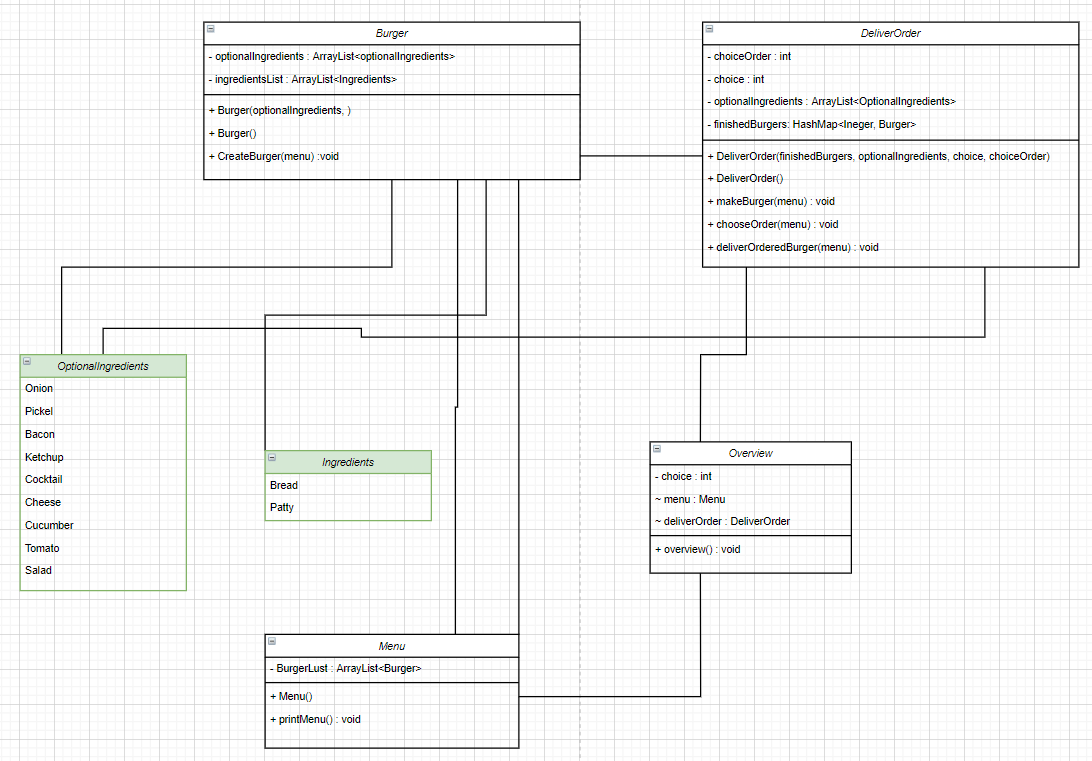
## Informieren

Wir haben uns zuerst zusammengesetzt und uns Gedanken darüber gemacht was wir gerne machen würden. Wir kamen mit der Hilfe von den Vorschlägen recht schnell auf die Idee, dass wir einem McDonalds-Simulator machen wollen

## Planen & Entscheiden

Danach haben wir unser UML und unsere Users-Cases erstellt. Diese haben wir kontrolliert und sichergestellt, dass alles vorhanden ist.

**UML (Davor)**

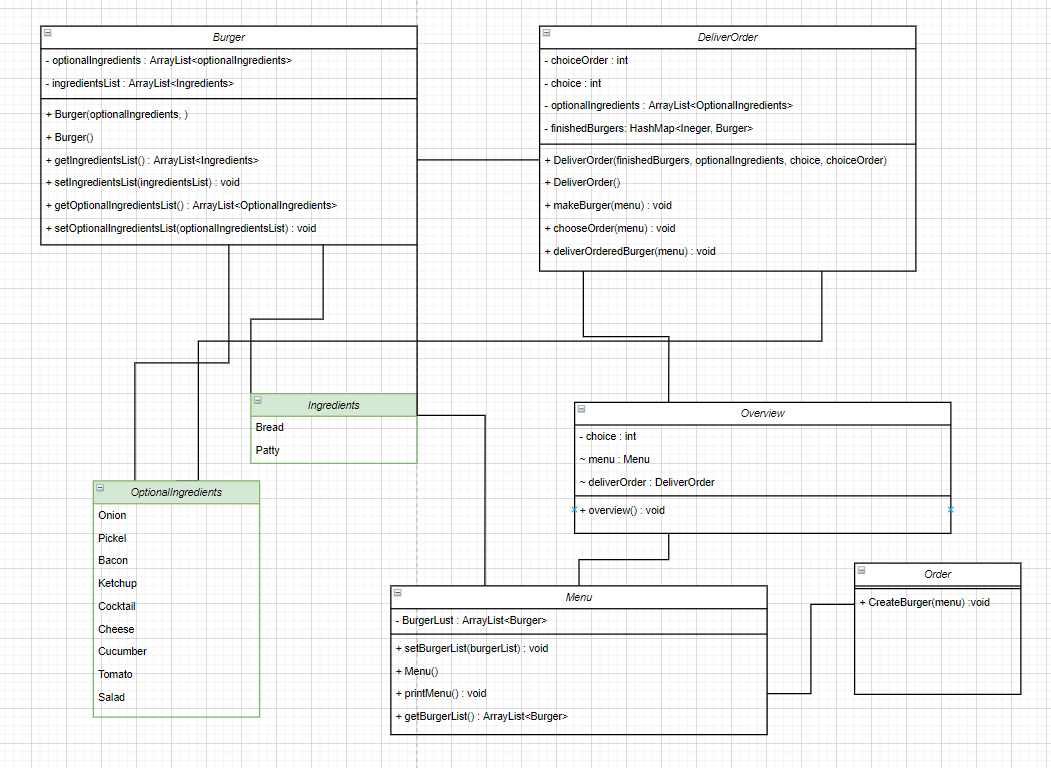


|  |  |  |
| --- | --- | --- |
| Use-Cases | Beschreibung | Vorhanden |
| Neue Bestellung erfassen | Es können mehrere Bestellungen erfasst werden. | Ja |
| Erhaltene Bestellungen erstellen random Burger | Burger für die Bestellungen werden verschieden generiert, so dass sie verschiedene Zutaten enthalten. | Ja |
| Überprüfe alle Bestellungen | Der Mitarbeiter kann alle Bestellungen managen. | Ja |
| Bestellung kann vorbereitet werden | Die Bestellung kann vorbereitet werden. Indem man die Burger selber zusammenstellen muss, so das sie mit der Bestellung überein stimmen. | Ja |
| Bestellung kann abgegeben werden (ausgeliefert) | Wenn man eine seiner Bestellungen fertig hat, kann man diese de Kunden übermitteln. Für jede übermittelte Bestellung erhält man Punkte | Ja |
| Simulation kann beendet werden | Wenn man mit der Bestellung fertig ist, kann der User, wenn er möchte die Simulation abbrechen. | Ja |

## Realisieren

Die meiste zeit haben wir danach in unser Programm gesteckt, weil dies am meisten Punkte gibt. Wir haben zuerst das Programm so aufgebaut, dass wir alle Punkte der Aufgabe abgedeckt haben. Danach haben wir verschiedene Erweiterungen vorgenommen und unser Programm verbessert.

**UML (Danach)**



## Kontrollieren

Als unser Programm so weit bereit war, haben wir mit dem Testen begonnen. Diese haben uns auch etwas zeit gekostet. Wir haben nicht nur unsere Tests gemacht, sondern auch unseren Code mit Kommentaren versehen**.**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Erwartetes Ergebnis | Erhaltenes Ergebnis | Fazit |
| Recieve new Order | Random Burger | Random Burger | Right |
| Check all open Orders after Receive new Order | Burger is shown | Burger is shown | Right |
| Prepare order after first 2 steps | Choose your order  Burger with index, ingredients and optional ingredients is shown | Choose your order  Burger with index, ingredients and optional ingredients is shown | Right |
| Choose Order after first 3 steps | Burger is chosen and you need to add Ingredients | Burger is chosen and you need to add Ingredients | Right |
| Add ingredients after first 4 steps | Adding ingredients (Cocktail, Pickle, Onion, Salad) | Adding ingredients (Cocktail, Pickle, Onion, Salad) | Right |
| Finish ordered Burger after first 5 steps | Type number 9 and made Burger is shown | Type number 9 and made Burger is shown | Right |
| Deliver ordered burger after first 6 steps | Ordered Burger and made Burger is shown with Points | Ordered Burger and made Burger is shown with Points | Right |
| End simulation | Typed number 9 and program has exited | Type numberd 9 and program has exited | Right |
| Deliver order (no order recieved) | “You first need to create a Burger to deliver one!” | “You first need to create a Burger to deliver one!” | Right |
| Prepare order (no order recieved) | “You first need to recieve a new order to make a Burger!” | “You first need to recieve a new order to make a Burger!” | Right |
| Check all open orders (no order recieved) | “There are no orders!” | “There are no orders!” | Right |
| Receive new Order and then Deliver Burger | “You first need to create a Burger to deliver one!” | “You first need to create a Burger to deliver one!” | Right |

## Auswerten

Schlussendlich haben wir alles in unsere Dokumentation ausgewertet. Wir haben alles zusammengefasst und die wichtigsten Informationen aus dem Programm aufgeschrieben.