Vertiefung der Programmierung

Projektbericht im Rahmen der Portfolio-Prüfung SoSe 2020

**1. Zeitraum: 13.07.2020 – 09.08.2020**

**Gruppe 11**

**Gruppenmitglied: Kjell Treder**

**Gruppenmitglied: Marcel Sauer**

1. **Spezifikation**

Mit der Anwendung kann der Nutzer Plätze für ein Autokino reservieren.

Ein Film und eine der verfügbaren Zeiten werden ausgewählt, um eine Ansicht der verfügbaren Plätze zu erhalten. Der Nutzer kann dann einen oder mehrere Plätze für die Reservierung auswählen. Es gibt Stellplätze für normalgroße Autos und Strandkörbe, sowie beide Varianten als VIP mit besserer Sicht für einen Aufpreis. Stellplätze für SUVs findet man hinten.

Nach der Auswahl optionale Angebote für Essen und Trinken kann die Reservierung bestätigt werden.

* 1. **Benutzergruppen**

Die Anwendung wird hauptsächlich von Nutzern verwendet, die gerne ins Autokino gehen und alternativ die Reservierung manuell vor Ort beim Autokino gemacht hätten.

Zu dieser Nutzergruppe gehören hauptsächlich Pärchen, Ehepaare und Familien mit Kindern, aber auch Studenten (ohne Auto), welche die Strandkörbe besetzen

* 1. **Funktionen**

Auswahl eines Kinofilms und der Vorstellungszeit

Der Nutzer kann einen von mehreren angebotenen Kinofilmen auswählen.

Der Nutzer kann eine der möglichen Vorstellungszeiten des ausgewählten Kinofilms auswählen.

Auswahl der Stellplätze/Strandkörbe

Der Nutzer kann einen oder mehrere Stellplätze und/oder Strandkörbe auswählen.

Der Nutzer muss das Kennzeichen des Autos für jeden ausgewählten PKW/SUV-Stellplatz angeben.

Auswahl an Essen und Trinken

Der Nutzer kann je 0-9 Bestellungen von unterschiedlichen Optionen von Essen und Trinken zu tätigen.

Reservierung bestätigen und abschicken

Der Nutzer kann, nachdem er seine Reservierung überprüft hat, diese endgültig abschicken.

Navigation

Der Nutzer kann mit „Zurück“, „Abbrechen“ und „Fortfahren“ den vorherigen und nächsten Bestellschritt einsehen und die Reservierung abbrechen.

Der Nutzer kann einen vorherigen spezifischen Tab einsehen, indem dieser in der Ansicht oben angeklickt wird

1. **Entwurf**
   1. **Klassendiagramm**
   2. **MVC**

Das Model

Das Model ist dafür verantwortlich die vom User eingegebenen Daten zu speichern, zu löschen und zu managen. Dazu gehören:

* Der ausgewählte Film
* Die ausgewählte Zeit
* Die ausgewählten Plätze
* Die eingegebenen Kennzeichen
* Die Menge für jede Essens/Trinkens-Option

Zusätzlich speichert und berechnet das Model Daten für die View, wie:

* Die verfügbaren Zeiten anhand des ausgewählten Filmes
* Die verfügbaren Plätze anhand der ausgewählten Zeit
* Die Anzahl an PKW/SUV-Stellplätzen (damit die View die richtige Anzahl an Textfeldern für die Kennzeichen zeigen kann)

Das Model kann anhand der Daten einen Preis kalkulieren und eine fertige Bestellung erstellen.

Das Erstellen einer Bestellung besteht aus:

* Reservieren aller Plätze
* Einem Update der Showtime, als Check ob das Reservieren der Plätze die Show ausverkauft hat
* Dem Erstellen und Hinzufügen (in die Datenbank) eines neuem Bestellungs-Objekts, welches eine Bestellnummer generiert bekommt, die in der Datenbank gespeichert wird.
* Schreiben eines neuen Dokuments mit der Beschreibung der Bestellung

Durch das Reservieren eines Platzes werden für Strandkörbe Tickets generiert, und PKW/SUV-Stellplätze Kennzeichen zugewiesen. Diese Daten werden auch in der Datenbank gespeichert.

Der Controller

Der Controller reagiert auf alle Inputs des Users und gibt die Informationen an das Model weiter. Dazu gehören die Übergabe folgender Informationen an das Model:

* Der ausgewählte Film aus dem ItemEvent
* Der Index der ausgewählten Zeit aus dem Action Command des ActionEvents
* Die x-y-Position des geklickten Platzes aus dem Action Command des ActionEvents und ob dieser ausgewählt ist
* Eine Liste der Eingaben aller Textfelder für die Eingabe von Kennzeichen
* Eine Liste aller Mengen der verfügbaren Essens/Trinkens-Optionen

Der Controller löst daraufhin das Update bei der View aus. Der Controller weist die View auch zur Navigation auf, sollte der „Zurück“ oder „Fortfahren“ Button gedrückt werden. Der Controller weist das Model darauf hin, dass das Programm geschlossen werden soll, wenn der „Abbrachen“ Button geklickt wird.

Die View

Die View besteht aus sechs unterschiedlichen Tabs, die alle unterschiedlich „gebuildet“ und „geupdated“ werden. Jeder Tab besitzt eine kurzen Anweisungstext gefolgt von Komponenten, die für jeden Tab unterschiedlich sind und zuletzt drei universellen Buttons für die Navigation, welche im abstrakten Tab definiert sind:

* Zurück: Der Zurück Button kann auf jedem Tab angeklickt werden, um auf den vorherigen Tab zu gelangen (Ausgeschlossen ist der erste Tab (StartTab).
* Abbrechen: Der Abbrechen Button kann auf jedem Tab angeklickt werden, um das Programm zu verlassen.
* Fortfahren: Der Fortfahren Button wird immer dann genutzt, um auf den folgenden Tab zu gelangen. Dadurch wird der folgende Tab zwangsläufig „gebuildet“. Die View switcht dann auf daraufhin auf diesen Tab, damit man diesen einsehen kann. Der Zugriff auf alle weiteren Tabs wird deaktiviert.

Der Fortfahren Button kann bei (den meisten) Tabs erst genutzt werden, nachdem eine bestimmte Aktion abgeschlossen wurde:

- Der Nutzer kann erst die Zeiten einsehen, wenn ein Film ausgewählt wurde

- Der Nutzer kann erst die Plätze einsehen, wenn eine Zeit ausgewählt wurde

- Der Nutzer kann erst die Essens/Trinkens-Optionen einsehen, wenn min. ein Platz ausgewählt wurde und alle erforderlichen Kennzeichen zureichend und korrekt eingegeben wurden.

Der Check, ob der Fortfahren Button für den Nutzer verfügbar ist geschieht in der update()-Methode jedes Tabs. Die Update Methode wird aufgerufen, wenn mit Komponenten auf dem Tab interagiert wurde. Eine Update des Tabs bedeutet zwangsläufig auch, dass alle folgenden Tabs deaktiviert werden, denn:

Beispiel: Der Nutzer wählt einen Film aus und schaut sich die Zeiten an. Da ihm die Zeiten nicht passen, geht er einen Tab zurück und wählt einen anderen Film aus. Da das Bauen des folgenden Tabs nur durch das Switchen durch den Fortfahren Button ausgelöst wird, d.h. dass der Tab nur die aktuellen und richtigen Informationen anzeigt, wenn der Fortfahren Button betätigt wurde, müssen alle folgenden Tabs deaktiviert werden.

Die build()-Methode jedes Tabs baut Komponenten abhängig von den Daten im Model auf. Die Build Methode wird jedes Mal aufgerufen, wenn zu einem Tab mit dem Fortfahren Button geswitcht wird.

Beispiel: Der Nutzer wählt einen Film und Zeiten aus und klickt auf Fortfahren. Dadurch wird der Tab aufgebaut, der dafür verantwortlich ist, die Plätze dieser Zeit anzuzeigen. Wie viele und welche Plätze die Zeit besitzt, d.h. wie viele Auswahlmöglichkeiten dem User angezeigt werden müssen ist abhängig von der ausgewählten Zeit (welche abhängig von dem ausgewählten Film ist). Sollte sich also die ausgewählte Zeit oder Film ändern, muss der folgende Tab neu gebaut werden um die neuen, richtigen Informationen anzuzeigen. Daher wird auch der Zugriff auf die folgenden Tabs deaktiviert, da diese bei einem Update möglicherweise nicht mehr die richtigen Informationen anzeigen würden.

Merke: Keine der Filme(-objekte) haben die gleichen Zeiten(-objekte) und keine Zeiten(-objekte) haben die gleichen Plätze(-objekte). Jeder Film hat individuelle Zeiten, welche je individuelle Plätze haben.

Die View kann ebenfalls Dialoge erstellen, wenn eine Exception auftritt oder die Bestellung abgeschickt wird.

Error Dialoge werden erstellt, wenn das „Bauen“ eines Tabs nicht gelingt. Dies kann in zwei Fällen auftreten:

* Der ausgewählte Film hat keine Zeiten, da diese null sind
* Die ausgewählte Zeit hat keine Plätze, da diese null sind oder die Anzahl an Reihen oder Spalten dieser Plätze gleich 0 sind

In einem solchen Fall wird der Dialog mit einer zugehörigen Error-Message angezeigt. Der Nutzer kann den Dialog schließen und weiterhin die Software nutzen und z.B. einen anderen Film/eine andere Zeit auswählen

Der Finishing-Dialog erscheint, wenn der User die Bestellung abschickt. Dieser Dialog beinhaltet einige Dank-Nachrichten und (falls vorhanden) Tickets für Strandkörbe. Das Abschicken einer Bestellung bringt den Nutzer wieder zum ersten Tab, sodass eine weitere Reservierung möglich wäre.

1. **OOP**
   1. **Umsetzung der Grundelemente der OO Entwicklung**
   2. **Umsetzung der Prinzipien des OO Entwurfs 1 und 4**
2. **GUI/Screenshots**
3. **Zusammenfassung**
   1. **Aufgabenteilung**
   2. **Ausblick**