Dokumentation

Nandini Hilger, Elisa Popp,

Dana Pluciennik und Patrick Vollstedt

Inhaltsverzeichnis

Vorwort

Das Projekt, indem wir einen Prototypen für unser Kinoticketreservierungssystem anfertigen, wurde von vier Gruppenteilnehmern durchgeführt – Nandini Hilger, Elisa Popp, Dana Pluciennik und Patrick Vollstedt.

Der angefertigte Prototyp beschränkt sich auf eine lokal erstellte Webseite für unser Kino MovieMaxx.

Im Namen aller Teilnehmer wird zunächst darauf hingewiesen, dass wir keine Kenntnisse im Umgang mit HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) und JavaScript aufgewiesen haben. Ebenso bereitete uns die Verbindung und Kommunikation zwischen dem Front- und Backend zunächst Komplikationen. Mithilfe sorgfältiger Recherche haben wir es schlussendlich geschafft eine Webseite für unser Kino MovieMaxx zu erstellen, auf der es unseren Kunden möglich ist beliebige Kinotickets online zu buchen.

Die Dokumentation der Durchführung dieses Projektes findet sich auf den folgenden Seiten wieder.

Das User-Interface

Die Erstellung des Grundgerüsts im Frontend

Wir haben uns für ein schlichtes Design der Website entschieden, um bei den Nutzern keine Reizüberflutung zu verursachen.

Dazu gehört die Beschränkung auf drei Hauptfarben:

dunkelgrau: rgb (37, 36, 36)

magenta/dunkelrot: rgb (88, 21, 32)

weiß: #fff

Das Dunkelgrau haben wir als Hintergrund unserer Website genutzt, wie auch für kleinere Akzente (s. z.B. Footer). Das dunkle Magenta dient zur Hervorhebung der wichtigen Sachen (Buttons, Checkboxen, …) und ist ebenfalls der Hintergrund des Footers. Es wird immer der gleiche Rotton verwendet, damit die Website harmonisch erscheint. Weiß haben wir durchgehend als Schriftfarbe benutzt, da es sich sowohl vor dem dunklen Hintergrund als auch vor dem Magenta deutlich abhebt und gut zu erkennen ist.

Das Design ist so insgesamt sehr einheitlich und findet sich auch in unserem selbst designten Logo wieder.

Der Header und Footer sind auf jeder Seite identisch, bis auf den Seiten, die zum Buchungsprozess gehören. Dies liegt daran, dass während diesem bestimmte Attribute im Browser zwischengespeichert werden. Wird der Prozess nun abgebrochen, weil auf eine andere Seite geklickt wird, so sind diese gespeicherten Attribute noch immer vorhanden, da es zu umständlich wäre, für jeder der 24 Verlinkungen in Header und Footer eine Funktion zu erstellen, die die Attribute löscht.

Wir haben uns daher dazu entschieden, beim Ticketbuchungsprozess eine „abgespeckte“ Variante von sowohl Header und Footer zu erstellen. Hier kann man nur noch über das Logo, sowie den „Startseite“-Button den Buchungsprozess abbrechen. Im Footer befinden sich keine Verlinkungen mehr.

Klickt man nun auf das Logo oder den „Startseite“-Button, so erhält man durch eine Java Script-Funktion eine Warnung, das alle bisher eingegebenen Daten verloren gehen, wenn man die Seite verlässt. Drückt man auf „Ok“, so werden alle eingegebenen Daten gelöscht.

Der Body jeder Seite wird von einem div der Klasse „wrapper“ umschlossen, das ausschließlich bewirkt, dass zwischen Body und Header nicht unnötig viel Platz ist. Dies wird durch ein einziges CSS-Attribut bewirkt:

```
.wrapper{ margin-top: -50px; }
```

Große Überschriften werden immer mit <h2> gekennzeichnet und etwas kleinere Überschriften mit <h3>. Als Schriftart wird (fast) durchgehend „Lato“ genutzt.

Footer:

Der Footer besteht aus vier aneinander gereihten Containern, in denen sich jeweils die Verlinkungen zu entsprechenden Seiten befinden. Diese Verlinkungen sind durch einen Hover-Effekt im CSS animiert, d.h. sie verschieben sich um 8px nach rechts.

Der rote Hintergrund wird nach oben hin verblendet, damit er sich besser in den Rest der Website einfügt.

Eine besondere Aufmerksamkeit haben die Social Links im Footer erhalten: Hier wurde erneut die Font-Awesome-Bibliothek für die Logos der sozialen Networks genutzt, welche in den bekannten Farben hinterlegt werden, sobald man mit der Maus über ihnen hovert (Bsp.: rot für Youtube, hellblaue für Twitter, …). Zusätzlich werden dann ihre Namen als speziell designter Tooltip angezeigt.

Beim Ticketbuchungsprozess wird der Footer komplett reduziert: Ausschließlich die rote Hintergrundfarbe bleibt erhalten. Aller Verlinkungen werden entfernt, nur eine abgewandelte Version des Logos ist zu sehen. Dies dient dazu, dass der Ticketbuchungsprozess nur über die dafür vorgesehenen Buttons abgebrochen werden könnte (s. Erklärung oben).

Das User-Interface

Die einzelnen Seiten im Frontend

Startseite:

Eine Übersicht unserer Filme soll unseren Kunden eine einladende und anziehende Startseite bieten. Durch einen angewendeten Hover-Effekt wird den Kunden die Information bereitgestellt, dass die bekannte Filmfortsetzung „007 – Keine Zeit zu sterben“ demnächst ins Kinoprogramm aufgenommen wird. Wenn unsere Kunden die Maus über die aufgelisteten Filme bewegen, wird ein Hover-Effekt ausgelöst und stellt erste Filminformationen auf einen Blick bereit.

Eine kurze Beschreibung des MovieMaxxs und der Ausstattung des Kinos rundet die Startseite ab.

Filmseiten:

Die Filme, die auf der *Startseite* und im *Programm* abgebildet sind, weisen jeweils eine Verlinkung auf, die die Benutzer zu einer neuen Seite führt – die jeweilige Filmseite. Für jeden Film wurde eine separate Seite angelegt, die neben der Filmlänge, dem Genre und der Angabe der Altersfreigabe auch die Zusammenfassung des Films bereitstellt. Die schriftliche Filminhaltsbeschreibung wird durch den jeweiligen Trailer ergänzt.

Auf derselben Seite werden dem Benutzer alle wöchentlich gleichbleibenden Vorstellungszeiten angezeigt, die in Form von Buttons auswählbar sind.

Events:

Wir, das MovieMaxx, bieten unseren Kunden zwei regelmäßig stattfindende Events an. Die Events werden durch zwei passend ausgewählte Bilder auf dem Reiter *Events* demonstriert. Auch hier wurde mit Hover-Effekten gearbeitet, um alle relevanten Information anzeigen zu lassen, wenn der Mauszeiger über das jeweilige Bild bewegt wird. Eine Weiterleitung zu einer zusätzlichen Seite ist somit nicht notwendig. So werden unseren Kunden alle wichtigen Informationen auf einen Blick bereitgestellt.

Preise:

Den Kunden wird die Preisberechnung unserer Kinokarten auf der jeweiligen *Preis-Seite* offengelegt. Alle relevanten Informationen werden in zwei verschiedenen Tabellen festgehalten. Das Rabattsystem, das wir verwenden, wird ausführlich offengelegt, damit unsere Kunden über unsere Rabattstufen und anderweitige Rabattmöglichkeiten informiert sind. Somit könne alle Benutzer nachvollziehen, wie der schlussendlich festgelegte Preis für Kinotickets zustande kommt. Die Tabellen wurden zu der farblichen Gestaltung der Webseite angepasst.

Anfahrt & Öffnungszeiten:

Selbstverständlich werden unseren Kino-Besuchern die Öffnungszeiten offengelegt und die Anfahrt durch die Verlinkung unseres Kino-Standorts in Mannheim auf Google-Maps vereinfacht.

Covid-19-Infos:

Alle aktuellen Corona-Informationen werden auf dieser Seite abgebildet und regelmäßig aktualisiert, damit unseren Kunden nichts für ein atemberaubendes Kino-Erlebnis im Weg steht.

Gutscheine:

Die Gutschein-Seite informiert und motiviert die Benutzer zum Kauf von Kinogutscheinen als Geschenk für Familienangehörige oder Freunde. Allerdings findet der Gutscheinverkauf nur im Kino vor Ort statt.

FAQ:

Die *FAQ-Seite* bietet unseren Kunde eine Übersicht der meistgestellten Fragen von Kunden an das Kino. Mittels aufklappbarer Antworten können die Benutzer die jeweilige Antwort zur Frage auf- und wieder zuklappen. Mithilfe von CSS wird die ausgewählte Frage mit dunkelrotem Hintergrund markiert.

Die Funktion der ausklappbaren Antworten schafft ein dynamisches Gesamtbild.

Kontakt:

Das Kontaktformular ist auf der *FAQ-Seite* verlinkt, um besondere Anlässe, wie z.B. einen Kindergeburtstag zu buchen oder eine fehlgeschlagene Buchungsbestätigung zu korrigieren.

Wenn doch noch Fragen oder andere Anmerkungen aufkommen, können sich die Kunden ebenfalls gerne über ein Kontaktformular an uns wenden.

Die verschiedenen Textfelder ermöglichen den Kunden das Eingeben einer beliebigen Nachricht. Durch die Angabe des Namens und der E-Mail-Adresse können wir den Kunden dann persönlich antworten.

Aus datenschutzrechtlichen Gründen wird der Kunde selbstverständlich dazu aufgefordert, die Verarbeitung der eingegebenen Daten im Zwecke der Kontaktanfragebearbeitung zu genehmigen.

Newsletter:

Hier können unsere Kino-Fans unseren Newsletter abonnieren, um immer up-to-date zu sein und exklusive Rabatte zu erhalten.

Karriere:

Auf dieser Seite werden offene Stellen in unserem MovieMaxx-Betrieb angezeigt, um neue und begeisterte Kinofans mit ins Team zu holen.

Derzeit sind folgende Stellen offen:

- Aushilfe auf 450€-Basis
- Ausbildung zur Servicekraft für das Jahr 2022
- Servicekraft in Vollzeit

Die Bewerbungsunterlagen werden vor Ort im Kino herzlich empfangen.

Logo:

Das Logo für unser Kino-Unternehmen wurde frei und kreativ entworfen und mit einem Slogan ergänzt, der bei unseren Kunden ganz einfach im Gedächtnis bleibt: **Wir sind Kino.**

Rechtliches (Impressum, Datenschutz, AGB und Hausordnung):

Wer sich im Internet über eine Website geschäftsmäßig präsentiert, muss gewissen Informationspflichten einhalten. Dazu gehört auch das Vorhandensein eines Impressums, das gewisse Informationen über die Firma bereitstellt. Aufgrund dessen wird ein Impressum, sowie eine Datenschutzverordnung und die Allgemeinen Geschäftsbedingungen zur Verfügung gestellt. Die rechtliche Erfüllung der Informationspflichten wird in unserem Fall durch unsere Hausordnung ergänzt.

Der Ticketbuchungsprozess

Saalplan:

Im Saalplan wird zuerst die Legende eingefügt, in der gezeigt wird, welche Farben ausgewählte, freie und belegte Sitze haben. Hierfür werden wieder die bereits bekannten Farben verwendet: weiß für ausgewählte, rot für belegte und grau (etwas heller als der Hintergrund, damit es sich abhebt) für freie Sitze.

Hierbei werden auch direkt die drei Klassen eingeführt: „seat“ für freie Sitze, „seat occupied“ für belegte Sitze und „seat selected“ für ausgewählte Sitze.

Für die Kinoleinwand wird ein rechteckiges div erstellt, welches durch den Neon-Effekt leuchtend erscheint. Hierfür wird der Rahmen einfach geblurred.

Der Sitzplan an sich besteht aus 8 Sitzen auf 5 Reihen, wobei die jeweils zwei äußeren Sitze immer etwas weiter von den vier inneren Sitzen weggerückt sind, um den Schein eines Gangs zu erwecken.

Die Sitze selbst sind ebenfalls schlichtweg divs, die heller sind als der Hintergrund und an den oberen Ecken abgerundet sind. Das sieht als Form ansprechender aus als einfache Quadrate. Sie werden größer durch ein Scale-Attribut im CSS, sobald man mit der Maus darüber hovert - aber nur, wenn die Sitze entweder frei oder ausgewählt sind. Sind sie bereits belegt, so wird kein Hover-Effekt ausgelöst.

Der Saalplan ist eine der wenigen Seiten, die tatsächlich mit JavaScript animiert wurde, abgesehen natürlich von den Seiten, die Informationen an den Server senden.

JavaScript wird hier zum einen dafür genutzt, dass der Sitz weiß wird, wenn er ausgewählt wird, oder dementsprechend wieder grau, wenn ein ausgewählter Sitz erneut angeklickt wird. Dafür wird ein Event Listener genutzt, der auf die Aktion des Anklickens reagiert.

Zum anderen werden auch die ausgewählten Sitze gezählt und die Anzahl darunter angezeigt. Dafür wird einfach die Länge des Arrays der ausgewählten Sitze angezeigt.

Unter dem Sitzplan befindet sich ein weiterer div mit Checkboxen, welche, sobald sie angeklickt werden, ein Input-Feld anzeigen. Auch dies wird mit Java Script gemacht. Hier

wird abgefragt, ob und für wie viele Personen gebucht wird, die sich für einen Rabatt qualifizieren.

Damit Checkboxen und Inputfelder nebeneinander angezeigt werden, werden sie in eine Tabelle mit zwei Spalten eingefügt.

Um nun zu verhindern, dass jemand keine Sitze auswählt und die Buchung trotzdem fortsetzen kann, wird eine JavaScript Funktion geschrieben, die in einer If-Abfrage prüft, ob die Anzahl an ausgewählten Sitzen gleich Null ist. Ist dies der Fall, erhält der Buchende ein Alert, dass ihn darauf hinweist, dass mindestens ein Sitz ausgewählt sein muss, um die Buchung fortzusetzen. Die Funktion wird aufgerufen, sobald man auf den „Bestätigen"-Button drückt.

Außerdem soll natürlich verhindert werden, dass jemand unten in die Inputfelder mehr Personen eingibt, als Sitzplätze ausgewählt wurden.

Hierfür werden den Eingaben in die Inputfelder in JavaScript Konstanten zugeordnet, die addiert werden und dann wird in einer If-Abfrage verglichen, ob die eingegebene Personenanzahl größer ist, als die Anzahl an ausgewählten Sitzen. Ist dies der Fall, so erhält der Buchende ein Alert, dass ihn darauf hinweist, dass mehr Personen eingeben wurden als Sitzplätze. Trifft keiner dieser beiden Fälle ein, so kann die Buchung fortgesetzt werden.

Registrierung, Login & Gastlogin:

Registrierung, Login & Gastlogin sind vom Aufbau her sehr ähnlich, nur findet man auf der Login-Seite sehr viel weniger Inputfelder.

Zuerst wird die Registrierung erstellt. Diese befindet sich in dem div mit der Klasse „registrierung", welches alle Inhalte der Seite in einen Container mit etwas hellerem Hintergrund als die restliche Seite packt. Außerdem hat der Container abgerundete Ecken. Dies dient dazu, dass dieser Teil der Website dem User direkt ins Auge sticht und hervorgehoben wird.

Alle Inputfelder (bis auf Alter und Passwort) sind durch korrekte Tags mit der Auto-Fill-In Funktion von Google kompatibel, was das Ausfüllen sehr einfach macht. Auch wird die weiße Umrandung rot, sobald ein Feld angeklickt wird oder ausgefüllt ist.

Alle Felder sind als „required“ getagged, d.h. sie müssen ausgefüllt werden, damit man zur nächsten Seite gelangt. Dies trifft aber nicht auf die Radio-Buttons zu, falls jemand sein Geschlecht nicht angeben möchte, da dies auch nicht relevant für die Buchung ist, sondern nur für eventuelle personalisierte Werbung per E-Mail.

Unter dem „Bestätigen“-Button befinden sich zwei Verlinkungen: Einmal die Verlinkung zum Login, falls man bereits ein Konto besitzt, und dann noch die Verlinkung zum Gastlogin, falls man sich nicht registrieren möchte. Hiermit werden bereits 3 unterschiedliche Personas erfolgreich abgedeckt.

Als Gast muss man seinen vollständigen Namen, eine E-Mail-Adresse und das Alter eingeben, um die Buchung fortsetzen zu können.

Für den Login muss man dann schlichtweg seine E-Mail-Adresse und das zugehörige Passwort eingeben. Sollte man dieses vergessen haben, so kann man, unter dem E-Mail-Inputfeld auf „Passwort vergessen?“ klicken und sich das Passwort per E-Mail zuschicken lassen.

Die E-Mail-Adresse nutzen wir in der Datenbank immer als eindeutigen Key zur Zuordnung der Konten, daher muss diese immer eingegeben werden.

Preisberechnung:

Die Berechnung ist in mehrere Schritte gestaffelt:

1. Eine Differenzierung zwischen den verschiedenen Filmen findet **NICHT** statt! Es wird für jedes Ticket ein **Standardpreis von 20 Euro** angenommen.

2. Jeder Kunde besitzt als Attribut eine Rabattstufe (zwischen 0 bis 3), welche durch häufige Buchungen gestaffelt ansteigt. Der Wert der Rabattstufe wird mit dem Faktor 0.1 mit dem Standardpreis berechnet.
   - ➔ Ab 5 Buchungen steigt die Rabattstufe von 0 (Neukunde) auf 1.
   - ➔ Ab 10 Buchungen steigt die Rabattstufe von 1 auf 2.
   - ➔ Ab 20 Buchungen steigt die Rabattstufe von 2 auf 3.

   Sollte das Kundenkonto verifiziert sein, dann erhält der Kunde einen zusätzlichen Rabatt von fünf Prozent.

   Beispiel: Rabattstufe 2, verifiziertes Kundenkonto:

   Neuer_Preis = 20 Euro * (1 - (2 * 0,1 + 0,05)) = 15 Euro

3. Außerdem wird die aktuelle Auslastung einer Vorstellung berücksichtigt. Je mehr ausgelastet eine Vorstellung ist, desto weniger Rabatt wird gewährt. Die Staffelung findet wie folgt statt:
   - ➔ Bis zu einer prozentualen Auslastung von 10% wird ein Rabatt von 12% gewährt.
   - ➔ Bei einer prozentualen Auslastung von 10,01% bis 25% wird ein Rabatt von 5% gewährt.
   - ➔ Bei einer prozentualen Auslastung von 25,01% bis 50% wird ein Rabatt von 2% gewährt.

   Beispiel: Die Vorstellung morgen früh ist nicht stark besucht (Gesamtauslastung 7%):

   Neuer_Preis = 15 Euro * (1 - 0,12) = 13,20 Euro

4. Als letztes Kriterium für die Berechnung vom Ticketpreis wird das Alter des Kunden verwendet.
   - ➔ Sollte der Kunde jünger als 10 Jahre alt sein, wird ein zusätzlicher Rabatt von 50% gewährt.
   - ➔ Sollte der Kunde zwischen 10 bis 17 Jahre alt sein, wird ein zusätzlicher Rabatt von 25% gewährt.
   - ➔ Sollte der Kunde zwischen 18 bis 26 Jahre alt sein, wird ein zusätzlicher Rabatt von 10% gewährt.

   Beispiel: Der Kunde ist 19 Jahre alt:

Neuer_Preis = 13,20 Euro * (1 - 0,1) = 11,88 Euro

5. Zuletzt wird der Preis immer auf den vollen Euro abgerundet.
   Beispiel: 11,88 Euro als momentaner Preis:
   Neuer_Preis = 11 Euro

Es ist zu erkennen, dass vor allem das Alter entscheidend für den Preis ist. Dadurch soll die junge Generation gefördert werden.

Wichtig ist, dass der Rabatt nicht addiert wird, sondern es handelt sich hierbei um einen iterativen Prozess. Der Rabatt wird jeweils auf den zuvor bereits genannten ausgerechneten neuen Preis angewendet.

So würde die vollständige Rechnung des Beispiels wie folgt aussehen:

20 Euro * (1 - (2 * 0,1 + 0,05)) * (1 - 0,12) * (1 - 0,1) = 11,88 Euro

Die Preise variieren zwischen:

P_{max} = 20 Euro

und

P_{min} = 20 Euro * (1 - (3 * 0,1 + 0,05)) * (1 - 0,12) * (1 - 0,5) = 5,72 Euro ➔ 5 Euro

Fazit: Somit können speziell treue junge Kund*innen bei einer nicht stark besuchten Vorstellung (oder als Frühbucher*innen) 75% sparen!

Bezahlung:

Hier wird abgefragt, wie der Kunde bezahlen möchte, folgende Möglichkeiten stehen zur Auswahl: Paypal, Kauf auf Rechnung, Kreditkarte oder Barzahlung im Kino.

Hierfür werden Checkboxen durch CSS zu containerartigen Boxen verändert, die man auswählen kann. Es ist standardmäßig immer die erste Box (Paypal) ausgewählt, wenn man auf die Website kommt. Es ist nicht möglich, keine Box auszuwählen.

Für die Logos der entsprechenden Bezahlmöglichkeiten wird die Font-Awesome-Bibliothek verwendet.

Selbstverständlich wurde der Vorgang der Bezahlung an sich nicht programmiert, da es sich hier nicht um ein echtes Kino handelt.

Kasse:

Für die Kasse wird eine abgewandelte Version des Containers aus der Registrierung verwendet. Der div nennt sich hier „kasse“ und ist schlichtweg etwas kleiner als der Container aus der Registrierung, hebt den Text aber trotzdem mit einer helleren Hintergrundfarbe hervor.

Die entsprechenden Daten, die hier angezeigt werden (also z.B. der ausgewählte Film, die eingegebenen Kontaktdaten, …), werden über den Server aus der Datenbank gezogen.

Die Vorgehensweise hierzu wird später genauer erklärt.

Die Kasse dient zur Zusammenfassung der Bestellung, damit der Kunde seine eingegebenen Daten noch einmal überprüfen kann.

Ticketversendung und Buchungsbestätigung per E-Mail:

Nachdem der Kunde seine beliebigen Sitzplätze ausgewählt hat und seine Kontaktdaten bei uns hinterlassen hat, folgt nach der Auswahl der Bezahlart auch schon die Versendung des Kinotickets per E-Mail. Die E-Mail ist so aufgebaut, dass wir unsere Kunden auf das Ticket im Anhang aufmerksam machen und sie dann noch an die tagesaktuellen Corona-Einlassregeln erinnern, sodass dem Besuch bei uns im MovieMaxx nichts mehr im Weg steht.

Eine Übersicht der Bestelldetails wird unseren Kinokunden zur Verfügung gestellt, damit alle Bestelldaten auf einen Blick sichtbar sind und nochmals überprüft werden können.

Durch das Vorzeigen dieses Online-Tickets wird der Einlass ins Kino garantiert.

Die Kommunikation zwischen dem Front- und Backend

Um den Austausch von Daten zwischen Frontend und Backend zu ermöglichen, mussten sich zunächst auf einheitliche Befehle in Form von URI's (Uniform Ressource Identifier) und Attribute (z.B. E-Mail und Passwort beim Login) geeinigt werden. Die folgende Tabelle zeigt alle Kommunikationen, welche zwischen Backend und Frontend implementiert wurden. Außerdem kann man sehen, welche Anfrage über JavaScript gesendet wird und welche Antwort der Server an das Frontend sendet.

GET / POST / DELETE	**BENÖTIGTE URI**	**ANFRAGE: BODY (im JSON oder String-Format)**	**Antwort: Body (JSON-Format)**
POST	http://localhost:8080/testdaten/alle (alle Testdaten werden in der Datenbank automatisiert angelegt)	-	-
POST	http://localhost:8080/vorstellungen/filmbekommen (wenn ein Film ausgewählt wurde, werden die dazugehörigen Vorstellungen angefragt und im Frontend angezeigt)	Terminator	[{ "vorstellungsid": 1, "filmName": "Transformers", "kinosaalNummer": 1, "startuhrzeit": "12:00", "laengeDerVorstellungInMinuten": "175" }, { ... }] (verkürzt)

POST	http://localhost:8080/vorstellungen/sitzplaetze
	(wenn daraufhin eine Vorstellung ausgewählt wurde, werden die Sitzplätze und deren Status angefragt, um diesen im Frontend anzuzeigen)
		{
		"filmName":"Transformers",
		"startuhrzeit":"12:00"
		}
			[
			{
			"sitzplatzID": 161,
			"reihe": 1,
			"spalte": 1,
			"vorstellungsID": 1,
			"statusVomSitzplatz": "BESETZT"
			},
			{ … }
			]
			(verkürzt)

POST	http://localhost:8080/kunden/register
	(Wenn sich ein Kunde registrieren möchte, werden die Daten, welche im Frontend eingegeben werden an den Server gesendet und in der Datenbank gespeichert – Wenn dieser Vorgang gelingt -> Antwort = True)
		{
		"vorname":"Max",
		"nachname": "Mustermann",
		"email":"max@gmail.com",
		"alterInJahren": 20,
		"strasse":"Musterstrasse",
		"hausnummer":"7a",
		"postleitzahl":"50667",
		"ort":"Musterstadt",
		"passwort":"TestPasswort"
		}
			True oder False

POST	http://localhost:8080/kunden/login
	(Bei einem Login wird nur die E-Mail (eindeutiger Identifikator) und das Passwort eingegeben. Falls diese stimmen, erhält man das jeweilige Kundenobjekt aus der Datenbank)
		{
		"email":"max@gmail.com",
		"passwort": "TestPasswort"
		}
			{
			"id": 25,
			"vorname":"Max",
			"nachname": "Mustermann",
			"email":"max@gmail.com",
			"alterInJahren": 20,
			"strasse":"Musterstrasse",
			"hausnummer":"7a",
			"postleitzahl":"50667",
			"ort":"Musterstadt",
			"passwort":"TestPasswort"
			"verifiziert": **false**
			}

POST	http://localhost:8080/bestellungen
	(Hier wird ein eindeutiger Identifikator für die jew. Bestellung angefragt)
		{
		"email":"max@gmail.com"
		}
			1 -> entspricht der BestellID

POST	http://localhost:8080/tickets
	(Das jeweilige Ticket wird an den Server gesendet und für 15 Minuten auf den Status “reserviert” gesetzt)
		{
		"startuhrzeit": "15:00",
		"kinosaalNummer": 4,
		
		"filmName":"Transformers",
		"preis": "20",
		"sitzplatzreihe": 1,
		"sitzplatzspalte": 1,
		"bestellungID": 1
		}
			True oder False

POST	http://localhost:8080/bestellungen/emailsenden	{
		"bestellID": 4,
		"bezahlart":“ Bar an der Kasse",
		}
			True oder False

	(wenn die Bestellung abgeschlossen wurde, wird die BestellId und Bezahlart an den Server gesendet)		
POST	http://localhost:8080/kunden/passwortvergessen (wenn ein Kunde sein Passwort vergessen hat)	max@gmail.com	True oder False

Beispiel: Eine Anfrage an den Server senden

Für alle oben aufgelisteten Anfragen an den Server haben wir jeweils eine Asynchrone Funktion in JavaScript erstellt.

Zunächst wird eine neue Instanz der Klasse XMLHttpRequests erzeugt. Dann wird eine neue Anfrage von dem angegeben Typ (z.B. POST) mit der festgelegten URI geöffnet. Daraufhin wird die Anfrage mit den notwendigen Daten (JSON oder String-Format) an den Server gesendet. Wenn die Anfrage an den Server erfolgreich war, wird die success-Funktion in JavaScript geöffnet, welche nun mit der Antwort des Servers arbeiten kann. Falls die Anfrage nicht erfolgreich war, wird eine Error-Funktion aufgerufen, welche über das Scheitern der Anfrage informiert.

Anzeigen der Daten im Frontend

Die Daten, welche wir durch die Antwort vom Server auf eine unserer Anfragen erhalten, soll nun teilweise im Frontend angezeigt werden (z.B. der aktuelle Saalplan). In den meisten Fällen wie z.B. bei der Vorstellung wird nur auf ein bestimmtes HMTL-Element über dessen ID zugegriffen und der jeweilige Text angepasst. Bei den Sitzplätzen jedoch erhalten wir ein Array mit allen Sitzplätzen für einen ausgewählten Film und eine ausgewählte Vorstellung. Jedes Element dieses Arrays stellt einen Sitzplatz da. Durch eine for-Schleife werden alle Elemente durchlaufen. Zunächst wird die ID des jeweiligen Sitzplatzes durch die Zusammensetzung verschiedener Strings bestehend aus der Reihe und Spalte des jeweiligen

Sitzplatzes ermittelt. So kann identifiziert werden welcher Sitzplatz in dem Array zu welcher ID im Frontend gehört. Dann wird der Status des aktuellen Elements geprüft. Wenn dieser ungleich “FREI” ist, wird die Klasse des Elements mit der zuvor ermittelten ID auf “occupied” gesetzt. Da die verschiedenen Klassen in der CSS-Datei unterschiedliche Farben zugeordnet bekommen haben bewirkt eine Änderung der Klasse schlussendlich eine Veränderung der Farbe im Frontend.

Zwischenspeichern von Daten im SessionStorage

Da nicht alle Daten sofort an den Server gesendet und einige Daten erst zu einem späteren Zeitpunkt benötigt werden haben wir uns über die Möglichkeiten informiert Daten im Browser des Benutzers zu speichern. Dabei sind wir auf den SessionStorage und den Localstorage von JavaScript gestoßen. Der Unterschied liegt lediglich bei der Haltbarkeit und dem Geltungsbereich. Die Daten des Sessionstorage sind begrenzt auf eine Session, werden also gelöscht, sobald man das Browserfenster schließt. Der SessionStorage ist außerdem auf ein Individuelles Fenster beschränkt, was z.B. der Falle entgegentritt, dass während eines Einkaufs bei mehreren geöffneten Tabs oder Browserfenstern der Kauf aus Versehen zweimal ausgelöst wird. Aus diesem Grund haben wir uns für den Sessionstorage entschieden.

Unser Bestellprozess sieht es vor, dass der Benutzer ohne Gastkonto, Login oder Registrierung den Film, die Vorstellung und seine Sitzplätze auswählen kann. Dies ist notwendig, falls ein Benutzer sich nur über Vorstellungszeiten oder die aktuelle Auslastung einer Vorstellung informieren möchte. Zunächst wird also ein Film ausgewählt, dann eine Vorstellung und daraufhin die Sitzplätze und (optional) das alter der Personen (für z.B. Rabatte). Dann folgt das Eintragen der E-Mail in Form von einem Login, einer Anmeldung oder einem temporären Gastkonto (an Session gekoppelt). An diesem Punkt wird über das Senden der E-Mail eine eindeutige BestellID für die aktuelle Bestellung bei dem Server angefragt und in den Sessionstorage zwischengespeichert. Über diese BestellID können nun die Zwischengespeicherten Informationen an den Server gesendet werden. In diesem Zuge werden die mitgesendeten Sitzplätze für 15 Minuten auf den Status “reserviert” gesetzt. Die Sitzplätze sollen erst auf “reserviert” gesetzt werden können, wenn der Benutzer seine E-Mail angegeben hat, da ab diesem Moment die Bestellabsicht klar ist. Dann kann nur noch die Bezahlart ausgewählt werden. Zum Abschließen der Bestellung wird nur die BestellID an den

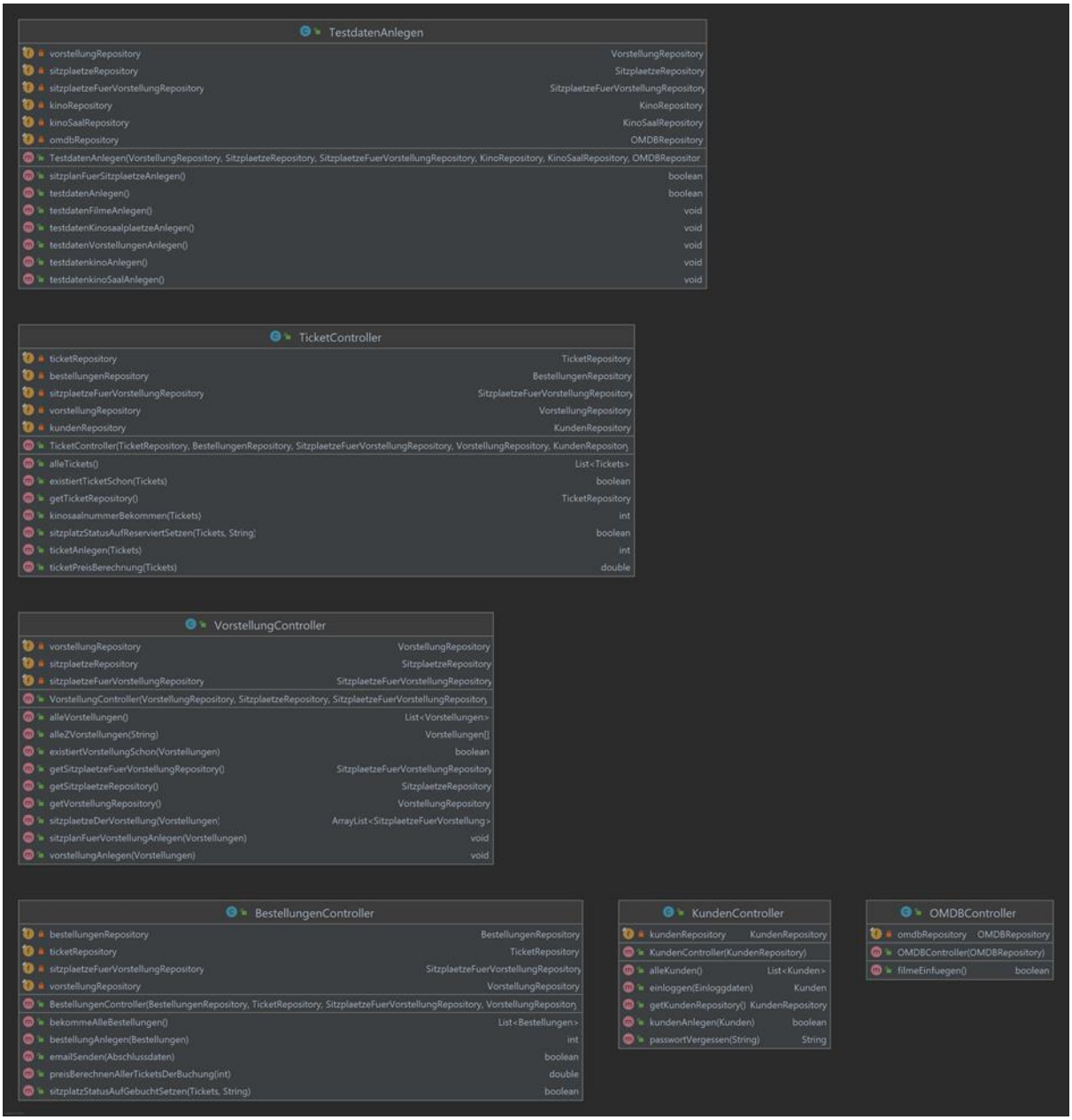
Server gesendet, welcher daraufhin den Status aller Sitzplätze mit dieser BestellID auf “besetzt” schaltet, sofern die 15 Minuten noch nicht abgelaufen sind.

Sitzplätze zwischenspeichern

Das Speichern der Sitzplätze im SessionStorage wurde durch ein Array realisiert. Da im SessionStorage nur Strings gespeichert werden können, musste das Array also über die Methode JSON.stringify() in einen String umgewandelt werden. Wenn dieses wieder benötig, wurde, konnte es durch die Methode JSON.parse() wieder in ein Array umgewandelt werden.

Anhang

Übersicht der Controller:



Übersicht über die Entitäten: