Semesteraufgabe: Java EE Architektur SS 2017

Kurzbeschreibung der Anwendung

Die entwickelte Softwarelösung orientiert sich an der Model 2-Architektur. Die Komponenten der Anwendung lassen sich demnach unterschiedlichen Schichten zuordnen.

Die erste Schicht behandelt die Darstellung der Inhalte und wird daher im Folgenden als Präsentationsschicht bezeichnet. Zu ihr zählen die Views welche im Verzeichnis src/main/webapp/ liegen.

Die Schicht der Geschäftslogik hingegen beschäftigt sich mit der konkreten Implementierung jeglicher Funktionalitäten zur Realisierung der User-Stories. Zu ihr zählen daher die Controller- und Request-Komponenten.

Die letzte Schicht beschäftigt sich mit der Speicherung der Daten und wird daher als Persistenzschicht bezeichnet. Zu ihr zählen die Datenmodelle und die jeweiligen Service-Komponenten, welche die Persistierung in H2 realisieren.

Tabelle 1: Architektur der Anwendung

Anwendungsschicht	**Beschreibung**
Präsentation	Views für die Abbildung der User-Stories
Geschäftslogik	Steuerung der Präsentation und Abbildung von Funktionalitäten
Persistenz	Datenmodell und Persistierung in H2

Manager-Status

Manager nehmen eine gesonderte Position ein. Sie sind die einzigen Nutzer die Veranstaltungen anlegen, veröffentlichen und bearbeiten können. Zusätzlich können Sie auch die Reservierungen zu ihren jeweiligen Veranstaltungen ansehen. Allerdings sei zu erwähnen, das eine Registierung als Manager durch den Anwender selber nicht möglich ist. Über die Seite register.jsf kann sich ein Anwender lediglich als ein normaler Benutzer registrieren, nicht jedoch als Manager. Diese Designentscheidung wurde aufgrund der Annahme getroffen, das sich ein Manager im Normalfall verifizieren muss. Dies kann allerdings nicht über ein einfaches Anhaken eines Kontrollkästchens erfolgen, sondern müsste über einen Mitarbeiter des Webseitenbetreibers geschehen, welche nach erfolgreicher Verifizierung diesen dann bei der aktuellen Implementation noch manuell in der Datenbank anlegen müsste.

Testdaten

Über die Seite /initDatabase.jsf können Testdaten in der Datenbank erstellt werden. Beim Aufruf der Seite werden dabei zwei Buttons angezeigt. Der Button "Datenbank zurücksetzen" löscht zunächst alle bisherigen Daten in der

Datenbank, sofern welche vorhanden sind. Danach befüllt er die Datenbank mit Testdaten. Der Button "Datenbank leeren" löscht dagegen alle Daten. Beim erstmaligen Deployen der Anwendung sowie nach dem Betätigen von "Datenbank leeren" sei zu beachten, das wie unter dem Kapitel Manager-Status beschrieben, keine Registierung als Manager möglich ist und somit auch keine Veranstaltungen erstellt werden können. Falls trotzdem darauf verzichtet werden soll, die Testdaten zu verwenden, so muss manuell per Insert auf der Datenbank ein Nutzer mit Manager-Rechten erstellt werden.

Zusätzlich sei zu erwähnen, das die Seite initDatabase.jsf neben login.jsf und register.jsf die einzige Seite ist, die ein Anwender auch ohne Anmeldung erreichen kann. Ebenfalls wird nirgendwo in der Anwendung auf initDatabase.jsf verlinkt. Dies ist der Tatsache geschuldet, das diese Seite lediglich den Testprozess erleichtern soll und keinen eigentlichen Teil der Anwendung darstellt.

Damit die Testdaten auch genutzt werden können, hier eine Übersicht aller erzeugten Nutzer mit ihren E-Mail-Adressen, Passwörten und Statusen:

Tabelle 2: Übersicht der Testnutzer

E-Mail	Passwort	Manager
admin@admin.de	admin	x
foo@bar.de	admin	x
test@test.de	admin	x
user@user.de	user	
sonst@was.com	user	
keine@idee.de	user	

Umsetzung der User-Stories

> Liste aller beteiligten Komponenten und Klassen, sowie deren Aufgaben und Zugehörigkeit zu den Anwendungsschichten.
> Kurze Beschreibung der Schritte, die ein Nutzer für die Anwendung der Story ausführen muss.

1. Veranstaltung anlegen

Komponente	Aufgabe	Anwendungsschicht
createEvent.xhtml	Darstellung des Formulars	Präsentation
CreateEventRequest.java	Steuerung des Formulars	Geschäftslogik
SessionContext.java	Authentifizierung	Geschäftslogik
ManagementOperation.java	Autorisierung: Ist der Aufrufende ein Manager?	Geschäftslogik

EventService.java	Persistierung	Persistenz
Event.java	Datenmodell	Persistenz

> Der Manager wählt nach dem Login zunächst im linken Navigationsbereich den Eintrag Veranstaltung anlegen aus. Danach erscheint ein Formular in dem die nötigen Angaben zum Erstellen einer Veranstaltung abgefragt werden. Ist der Manager fertig mit der Eingabe der benötigten Daten, kann die Veranstaltung über den gleichnamigen Button erstellt werden.

2. Veranstaltung veröffentlichen

Komponente	**Aufgabe**	**Anwendungsschicht**
publishEvents.xhtml	Auflistung seiner noch nicht veröffentlichten Veranstaltungen	Präsentation
SessionContext.java	Authentifizierung	Geschäftslogik
ManagementOperation.java	Autorisierung: Ist der Aufrufende ein Manager?	Geschäftslogik
EventService.java	Durchführen der Veröffentlichung inklusive Persistierung	Persistenz
Event.java	Datenmodell	Persistenz

> Der Manager wählt nach dem Login zunächst im linken Navigationsbereich den Eintrag Veranstaltung veröffentlichen aus. Danach erscheint eine Liste der noch nicht veröffentlichten Veranstaltungen. Hier hat der Manager die Option eine Veranstaltung aus der Liste direkt zu veröffentlichen. Alternativ hierzu kann er sich auch zunächst die Details anzeigen lassen um vorher noch einmal die eingegebenen Daten näher zu kontrollieren und ggf. zu bearbeiten.

3. Veranstaltung bearbeiten

Komponente	**Aufgabe**	**Anwendungsschicht**
changeEvent.xhtml	Darstellung des Formulars	Präsentation
ProcessEvent.java	Steuerung des Formulars	Geschäftslogik
SessionContext.java	Authentifizierung	Geschäftslogik
SecurityContext.java	Autorisierung: Ist der angemeldete Nutzer auch der Ersteller der Veranstaltung?	Geschäftslogik
EventService.java	Persistierung	Persistenz
Event.java	Datenmodell	Persistenz

> Der Manager wählt nach dem Login zunächst im linken Navigationsbereich den Eintrag Meine

> Veranstaltungen aus. Aus der Liste der Veranstaltungen wählt (Details) er dann eine aus, die er gerne bearbeiten möchte. Hier bekommt der Manager zunächst eine Übersicht über die aktuellen Eigenschaften der Veranstaltung und hat die Option, die Veranstaltung zu bearbeiten. Die Anpassungen bestätigt der Manager mit einem Klick auf den Button Änderungen speichern, dadurch wird er zurück auf die Übersicht seiner Veranstaltungen geleitet.

4. Veranstaltung suchen

Komponente	Aufgabe	Anwendungsschicht
search.xhtml	Darstellung des Formulars	Präsentation
SearchRequest.java	Steuerung des Formulars	Geschäftslogik
SessionContext.java	Authentifizierung	Geschäftslogik
EventService.java	Durchführen der Abfrage	Persistenz
Event.java	Datenmodell	Persistenz

> Der Anwender wählt nach dem Login zunächst im oberen Navigationsbereich den Eintrag Suche aus. Dort gibt er einen Suchbegriff ein, dies kann beispielsweise der *Ort* Berlin sein. Nach der Bestätigung seiner Eingabe werden ihm alle Veranstaltungen die zu seiner Suche passen aufgelistet. Dabei wird der *Name*, die *Beschreibung* und der *Ort* der Veranstaltung berücksichtigt.

> Alternative: Darüber hinaus kann der Anwender bei jeder Auflistung von Veranstaltungen über die Filter unter den Spaltenüberschriften nach spezifischen Veranstaltungen suchen. Hierzu werden Filter für den *Namen*, die *Art* und den *Ort* der Veranstaltung angeboten.

5. Veranstaltung ansehen

Komponente	Aufgabe	Anwendungsschicht
events.xhtml	Auflistung der Veranstaltungen	Präsentation
event.xhtml	Darstellung der Details	Präsentation
SessionContext.java	Authentifizierung	Geschäftslogik
EventService.java	Durchführen der Abfrage	Persistenz
Event.java	Datenmodell	Persistenz

> Der Anwender erhält nach dem Login eine Übersicht der veröffentlichten Veranstaltungen, die in Zukunft stattfinden werden. Hierbei wird ihm *Name*, *Art*, *Ort*, *Datum* und das noch zur Verfügung stehende *Kontingent an Tickets* der Veranstaltung angezeigt. Nach einem Klick auf Details neben der Veranstaltung wird ihm zusätzlich eine weiterführende *Beschreibung* der Veranstaltung angezeigt.

6. Ticketreservierung

Komponente	Aufgabe	Anwendungsschicht
bookEvent.xhtml	Darstellung des Formulars	Präsentation
BookEventRequest.java	Steuerung des Formulars	Geschäftslogik
SessionContext.java	Authentifizierung	Geschäftslogik
ReservationService.java	Persistierung	Persistenz
Reservation.java	Datenmodell	Persistenz

> Der Anwender hat nach dem Login eine Veranstaltung aus der Übersicht ausgewählt, die er gerne besuchen würde. In der Detail-Ansicht erhält er die Option Ticket reservieren. Hierbei wird ihm eingeblendet wieviele Karten aktuell noch zur Verfügung stehen. Nach einer Eingabe der gewünschten Anzahl an Tickets, kann der Anwender die Reservierung bestätigen oder den Vorgang abbrechen. Bestätigt er seinen Reservierungswunsch, so wird - wenn noch genügend Tickets zur Verfügung stehen - die Reservierung bestätigt.

7. Reservierungsbestätigung

Komponente	Aufgabe	Anwendungsschicht
bookEvent.xhtml	Darstellung des Formulars aus 6.	Präsentation
myReservations.xhtml	Darstellung der Reservierungsbestätigung	Präsentation
BookEventRequest.java	Steuerung des Formulars aus 6.	Geschäftslogik
SessionContext.java	Authentifizierung	Geschäftslogik
ReservationService.java	Persistierung	Persistenz
Reservation.java	Datenmodell	Persistenz

> Bei dem Bestätigen einer Reservierung wird noch einmal geprüft, ob die vom Anwender gewünschte Anzahl an Tickets nach wie vor zur Verfügung steht. Danach erhält der Nutzer eine dementsprechende Meldung. Stehen nicht mehr genügend Tickets zur Verfügung kann der Anwender seine Eingabe anpassen.
>
> Sobald eine Reservierung erfolgreich durchgeführt wurde, wird dem Anwender ein Reservierungscode mitgeteilt. Hierzu gelangt er zu einer Übersicht seiner aktuellen Reservierungen, bei der oben die Nummer der neuen Reservierung angegeben ist.

8. Reservierungsübersicht

Komponente	Aufgabe	Anwendungsschicht

myReservations.xhtml	Auflistung der persönlichen Reservierungen	Präsentation
BookEventRequest.java	Auf Wunsch Stornierung einer Reservierung	Geschäftslogik
SessionContext.java	Authentifizierung	Geschäftslogik
ReservationService.java	Persistierung	Persistenz
Reservation.java	Datenmodell	Persistenz

> Der Anwender wählt nach dem Login im linken Navigationsbereich den Eintrag Meine Reservierungen aus. Auf dieser Seite wird im eine Übersicht seiner aktuellen Reservierungen dargestellt. Wenn er sich entscheiden sollte eine Veranstaltung doch nicht besuchen zu wollen, hat er hier die Möglichkeit die entsprechende Reservierung zu stornieren.

9. Noch reservierbare Tickets

Komponente	**Aufgabe**	**Anwendungsschicht**
events.xhtml	Auflistung der Veranstaltungen	Präsentation
event.xhtml	Darstellung der Details	Präsentation
SessionContext.java	Authentifizierung	Geschäftslogik
Event.java	Datenmodell	Persistenz

> Der Anwender wird an allen Stellen bei denen das freie Kontingent für ihn wichtig ist, über den aktuellen Stand informiert. Sowohl auf der Übersichtsseite der Veranstaltungen, als auch bei den Details einer Veranstaltung und der letzlichen Reservierung wird ihm daher die Anzahl noch reservierbarer Tickets angezeigt.

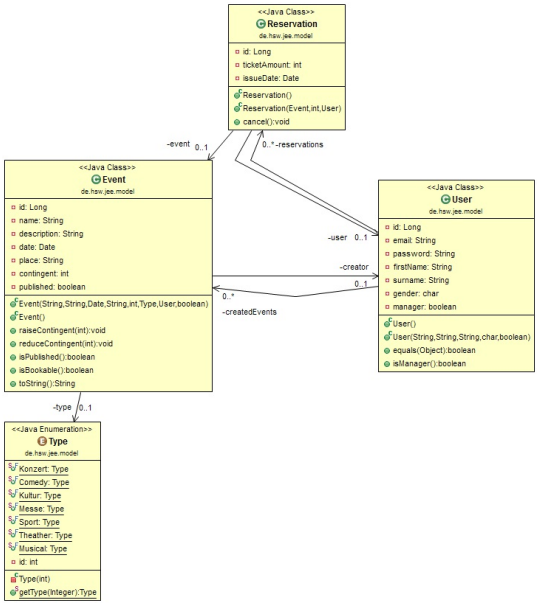
Tabelle 3: Sprint-Backlog

Task	**Beschreibung**	**Story**	**Entwickler**
Projekt-Setup	Git, Gradle, Eclipse, Primefaces, Templating	-	Florian
Datenmodell	Konzipierung der benötigten Entitäten	-	*Gemeinsam*
Authentifizierung	Registrierung & Anmeldung an der Anwendung	-	Arthur
Autorisierung	Zugriffsschutz für Seiten- und Methodenaufrufe, die Managern vorbehalten sein sollen	-	Florian
Erstellung Testdaten	Erstellung der Testdaten und Einbindung in die Anwendung	-	Arthur

Veranstaltung anlegen	Erstellen von unterschiedlichen Veranstaltungen	1	Arthur
Veranstaltung veröffentlichen	Veranstaltungen werden erst nach der Freigabe durch ihren jeweiligen Manager für andere sichtbar	2	Florian
Veranstaltung bearbeiten	Ändern von Eigenschaften einer erstellten Veranstaltung durch den jeweiligen Manager	3	Arthur
Veranstaltung suchen	Kontextbasiertes Durchsuchen zzgl. konkreter Filter in den Auflistungen	4	*Gemeinsam*
Veranstaltung ansehen	Einblenden von Detailinformationen zu einer Veranstaltung	5 & 9	Arthur
Ticketreservierung	Reservieren von Tickets und Möglichkeit zur Stornierung	6 & 9	Florian
Reservierungsbestätigung	Vergabe und Mitteilen einer eindeutigen ID für jede Reservierung	7	Florian
Auflistung von Reservierungen	Auflistung aller Reservierungen zu den Veranstaltungen des angemeldeten Managers	8	Florian

Datenmodell

Klassendiagramm



Beschreibung der Datenbankstruktur

Die Datenbankstruktur wird beim Deployment der Webanwendung automatisch erstellt. Die einzelnen Datenbanktabellen werden von der Java Persistence API anhand der Klassen, die als Entity gekennzeichnet sind

abgeleitet und automatisch generiert. Diese Persistence API übernimmt zusätzlich zum ableiten der Datenbankstruktur noch einige weitere Aufgaben. So sorgt sie auch für das Erstellen einer eindeutigen ID für die jeweiligen Instanzen der Klassen, kümmert sich um die Integrität der Datenbank und das Verwalten der Beziehungen zwischen den einzelnen Entitäten.
Insgesamt sind drei Entitäten für die Webanwendung entscheidend:

- Zunächst wäre hier der User zu nennen. Er repräsentiert einen realen Nutzer der Webanwendung und wird mit der Kombination aus E-Mail-Adresse und einem Passwort gekennzeichnet. Ein User kann zusätzlich als Manager markiert werden. Dies gibt im die Möglichkeit Veranstaltungen zu erstellen, diese frei zu geben, zu bearbeiten und zugehörige Reservierungen einzusehen.
- Eine weitere Entität ist das Event. Sie stellt eine Veranstaltung dar, bestehend aus einem Namen, einer Beschreibung, einem Veranstaltungsort, einer Datum mit Uhrzeit und einer Anzahl verfügbarer Karten. Zusätzlich kann jede Veranstaltung noch einer Kategorie zugeordnet werden. Eine Veranstaltung kann dabei entweder unveröffentlicht oder veröffentlicht sein. Eine noch nicht veröffentlichte Veranstaltung ist nur vom jeweiligen User der sie erstellt hat einsehbar und auch nur dieser kann sie veröffenlichen. Der Ersteller der Veranstaltung ist in dem Feld Creator_ID gespeichert.
- Als Letztes gibt es noch die Reservation. Sie bildet eine Reservierung für eine Veranstaltung von einem User ab. Als solche speichert sie diese Veranstaltung und den Nutzer in den entsprechenden Feldern Event_ID und User_ID ab. Zusätzlich besitzt sie noch das Attribut ticketamount, welches die Anzahl an reservierten Tickets abspeichert, und das issuedate, welches das Datum mit Uhrzeit speichert, an welchen die Reservierung erfolgte.