INF202 Meilenstein #4

Verantwortliche/r:

Teoman Turan, e180503040@stud.tau.edu.tr

Barış Emre Yaşar, e180503038@stud.tau.edu.tr

Kapitel 1.

Unser Projekt ist ein Urlaubsbuchungssystem, das mit JavaScript, HTML und CSS entwickelt wurde. Das System basiert auf einer Client-Server-Architektur. Auf der Client-Seite wird ein Webbrowser verwendet, über den Benutzer mit dem System interagieren können. Auf der Server-Seite nutzen wir die Node.js-Umgebung. Als Datenbank verwenden wir eine SQL-basierte Datenbank, die mit dem Code-First-Ansatz erstellt wurde.

Die Hauptkomponenten des Projekts sind:

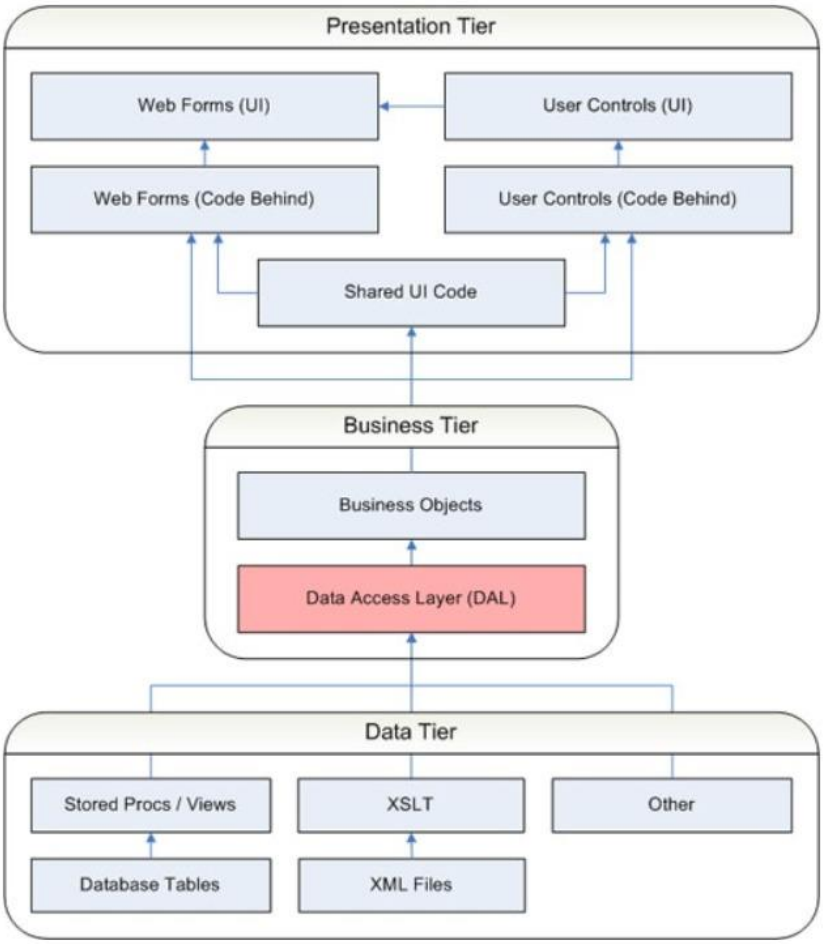
Benutzeroberfläche: Die Benutzeroberfläche wurde mit HTML und CSS gestaltet. Benutzer können über die Benutzeroberfläche Urlaubspakete oder Aktivitäten buchen.

Controller-Klassen: Die Controller-Klassen, die mit JavaScript erstellt wurden, nehmen HTTP-Anfragen entgegen, verarbeiten sie und führen die entsprechende Geschäftslogik aus. Zum Beispiel verwaltet die Klasse ReservationController die Buchung von Urlaubspaketen, während die Klasse ActivityController Buchungen für Aktivitäten verwaltet.

Datenbankzugriffsschicht: In unserem Projekt verwenden wir das Spring Framework, um auf die Datenbank zuzugreifen. Die Repository-Klassen definieren, welche Daten in der Datenbank gespeichert werden und wie auf sie zugegriffen wird. Die Klasse ReservationRepository speichert Reservierungsdaten in der Datenbank und ermöglicht den Zugriff darauf.

Es wird sich aus 3 Layers bestehen:

- Presentation Layer
- BusinessLayer
- Data Acces Layer
- Entity Layer



Kapitel 2

Die Controller-Klassen spielen in diesem Projekt eine wichtige Rolle, da sie die Verarbeitung von Benutzeranfragen und die Rückgabe von Ergebnissen steuern. Der Controller fungiert als Vermittler zwischen der View und dem Model und ist verantwortlich für die Koordination der Logik.

ReservationController: Diese Klasse verwaltet die Buchung von Urlaubspaketen. Sie nimmt HTTP-Anfragen entgegen, überprüft die erforderlichen Daten und speichert die Buchungsdaten in der Datenbank. Darüber hinaus ermöglicht sie Funktionen wie Bestätigung, Stornierung und Aktualisierung von Buchungen. Beispielcode:

javascript

```
class ReservationController {
  createReservation(request, response) {
    // Überprüfe die erforderlichen Daten und speichere die Buchung in der Datenbank
    // Sende eine Antwort mit dem Ergebnis
  }

  updateReservation(request, response) {
    // Überprüfe die erforderlichen Daten und aktualisiere die Buchung in der Datenbank
    // Sende eine Antwort mit dem Ergebnis
  }
```

```
  // Weitere Funktionen...
}
```

ActivityController: Diese Klasse verwaltet Buchungen für Aktivitäten. Sie nimmt HTTP-Anfragen entgegen, überprüft die erforderlichen Daten und speichert die Buchungsdaten in der Datenbank. Darüber hinaus ermöglicht sie Funktionen wie Stornierung und Aktualisierung von Buchungen. Beispielcode:

```
class ActivityController {
  createActivityReservation(request, response) {
    // Überprüfe die erforderlichen Daten und speichere die Aktivitätsbuchung in der Datenbank
    // Sende eine Antwort mit dem Ergebnis
  }

  cancelActivityReservation(request, response) {
    // Überprüfe die erforderlichen Daten und storniere die Aktivitätsbuchung in der Datenbank
    // Sende eine Antwort mit dem Ergebnis
  }

  // Weitere Funktionen...
}
```

Kapitel 3

KUNDE

Konto erstellen – Controller kontrolliert, ob diesen Kunde in der Datenbank schon erstellt wird.

Kontodaten aktualiesieren - Controller

Hotels suchen und filtern - Controller sucht die Datenbank und gibt die Benutzer,die Hotels diese Bedingen erfüllt.

- Bewertungen und Blogs von den Hotels sehen
- Vergangene Reservierungen ansehen

Reservierungen machen

- Reservierungen aktualisieren und absagen
- Die Urlaub bezahlen

Hotels bewerten und Blogs über ihre Reise schreiben.

Anforderung 1: Kunden sollen in der Lage sein, Urlaubspakete zu reservieren.

Controller: ReservationController

Beschreibung: Diese Anforderung zielt darauf ab, dass Kunden über die Web-Benutzeroberfläche Urlaubspakete reservieren können. Der ReservationController ermöglicht es den Kunden, Reservierungen zu erstellen, zu aktualisieren und zu stornieren.

Anforderung 2: Kunden sollen in der Lage sein, Aktivitäten zu reservieren.

Controller: ActivityController

Beschreibung: Diese Anforderung zielt darauf ab, dass Kunden über die Web-Benutzeroberfläche Aktivitäten reservieren können. Der ActivityController ermöglicht es den Kunden, Aktivitätsreservierungen zu erstellen und zu stornieren.

Anforderung 3: Kunden sollen Reservierungsbestätigungen und Benachrichtigungen erhalten.

Controller: ReservationController, ActivityController

Beschreibung: Diese Anforderung zielt darauf ab, automatische Bestätigungs- und Benachrichtigungse-Mails an Kunden zu senden. Der ReservationController und ActivityController verwalten den Versand von E-Mails an Kunden, die Reservierungsinformationen enthalten.

Jede dieser Anforderungen wird von den entsprechenden Controller-Klassen erfüllt. Die Controller-Klassen verarbeiten HTTP-Anfragen, führen erforderliche Überprüfungen durch und steuern die entsprechende Geschäftslogik. Sie interagieren auch mit den Repository-Klassen, um auf die Datenbank zuzugreifen.

Jede einzelne Anforderung und Controller-Klasse betont die Beziehung zwischen den grundlegenden Komponenten Ihres Projekts.

Kapitel 4

In unserer DB-Zugriffsschicht verwenden wir das Spring Framework, um auf die Datenbank zuzugreifen und die Datenmodelle zu verwalten. Hier sind die Details der Repository-Klassen und Datenmodelle, die in unserem Projekt verwendet werden:

1. ReservationRepository:

Das ReservationRepository ist verantwortlich für den Zugriff auf Reservierungsdaten in der Datenbank. Es verwendet das Reservation-Datenmodell, das die Informationen über eine einzelne Reservierung enthält. Das Datenmodell enthält Felder wie Kundenname, Reservierungsdatum, Anzahl der Gäste und weitere relevante Informationen.

```
@Repository
public interface ReservationRepository extends JpaRepository<Reservation, Long> {
   // Methoden zur Abfrage und Aktualisierung von Reservierungsdaten
}
```

2. ActivityRepository:

Das ActivityRepository ermöglicht den Zugriff auf Aktivitätsbuchungsdaten in der Datenbank. Es verwendet das ActivityReservation-Datenmodell, das Informationen über eine Aktivitätsbuchung enthält. Das Datenmodell enthält Felder wie Aktivitätsname, Buchungsdatum, Teilnehmerzahl und weitere relevante Informationen.

Beispielcode:

```
@Repository
public interface ActivityRepository extends JpaRepository<ActivityReservation, Long> {
   // Methoden zur Abfrage und Aktualisierung von Aktivitätsbuchungsdaten
}
```

Diese Repository-Klassen sind Teil der DB-Zugriffsschicht und verwenden das Spring Data JPA-Framework, um die Datenbankabfragen und -operationen zu vereinfachen. Sie erben die grundlegenden CRUD-Methoden von der JpaRepository-Klasse, die das Erstellen, Lesen, Aktualisieren und Löschen von Daten unterstützt. Darüber hinaus können benutzerdefinierte Abfragemethoden in den Repository-Klassen definiert werden, um spezifische Abfragen und Operationen auf den Datenmodellen auszuführen.