**Hotel Booking Application Project**

**CSIS 23323**

**Group members:**

Ewa Ferens

Melanie Hammermaster

Alisher Matianiu

1. **Project description**

The “Hotel Booking System” is the web-based system, designed for managing rooms reservations in the hotel. This system replaces the manual method of booking rooms, and can be used by customers (guests) and authorized hotel staff.

The objective of the project is that customers can easily go online and make room reservation and receptionist can access customer’s information and reservation list, as well as edit or delete reservations. Create an account option will secure user data. The following in the project is ensured:

* Overall system quality is good and it provides a smooth experience for both customers and hotel staff.
* The system is easy to maintain and navigate. Object oriented programming techniques are used in the project.
* Customer’s details are secured in the profile module.
* Data validation and the security checks are done when it is necessary.
* System is integrated with hotel data base.

**Features:**

* Login and logout to user account.
* User authentication system with password encryption. The main aim of this feature is to maintain confidentiality, authenticity and integrity of stored information. The user information is verified with the user information stored in the database tables and if information matches, they should be able to access system. Once entered in the account, based on the user type and access rights they can use system functionality.
* Different sessions for customer and hotel staff. In the customer session members can make room reservation, access their user information, and their current reservations. In the staff session, members can access any user information and edit or delete reservations.

**Implemented technical concepts:**

* HTML: Page layout has been designed in HTML, Bootstrap
* CSS: CSS has been used for the layout design
* PHP: Logic has been implemented in PHP
* MySQL: MySQL database has been used as database for the project
* Wampserver: Project runs on Wampserver
* The number of entities in database: 4
* The application support CRUD for the entities
* There are HTML forms for: User registration, Booking room
* Statistics are shown for reservations and users.
* During user registration data are validated.
* Users are able to login and their passwords are encrypted

**Installation manual:**

* Source file for database is included in: sql/hotel.sql
* **Folder structure:**
  + **css**
  + **img**
  + **inc**
    - **Entities:** 
      * Page.class.php, Reservation.class.php, Room.class.php, Service.class.php, User.class.php
    - **Utility:** LoginManager.class.php, PDOService.class.php, ReservationDAO.class.php, RoomDAO.class.php, ServiceDAO.php, Validate.class.php
    - config.inc.php
  + **js**
  + **log**
  + **sql**

**main files:**

* + hotelLogin.php
  + hotelRegistration.php
  + staffView.php
  + userView.php

**Responsibilitiesof group members:**

Ewa Ferens: Database design: modeling diagram, ER-Model Mapping, SQL, Project report, Entities( Reservation, Room, Service, User, Page), Utility(PDOService, ReservationDAO, RoomDAO, ServiceDAO, UserDAO), config.inc, hotelReservation, hotelViewReservation, hotelUserProfile

Melanie Hammermaster: HTML, CSS

Alisher Matianiu:

**2. Database**

**2.1. EER Modeling Diagram**

Diagram 2.1 presents database schema which has been created for this project. The schema shows 4 entities: User, Room, Reservation, Service and their cardinal relationships and participation.

User

1

Room

Reservation

1

M

Included in

1

Included in

Service

M

N

Makes

**2.2. ER-Model Mapping to Database Relational Schema:**

User(email, password, userType, full\_name, address, postalCode, city, country, phone)

Reservation (reservationNr, email\*, startDate, endDate, paymentDate, roomNr\*)

Room (roomNr, type)

Service (serviceNr, type, price)

Reservation\_includes\_service (serviceNr\*, reservationNr\*, price, type)

*Notes: primary key , \* foreign key*

* 1. **Data Types (Domain) and Constraints:**

email: VARCHAR(4 0)

password**:** VARCHAR(250)

userType: VARCHAR(20)

full\_name: VARCHAR(40)

address: VARCHAR (50)

postalCode: VARCHAR(8)

city VARCHAR(20)

country: VARCHAR(20)

phone VARCHAR(20)

roomNr VARCHAR(4)

roomType: VARCHAR(20)

reservationNr: INT()

startDate: DATE

endDate: DATE

paymentDate: DATE

serviceNr: VARCHAR (4)

Service.type: VARCHAR(20)

price: FLOAT (4,2)

*Notes: All data constrained: NOT NUL*L

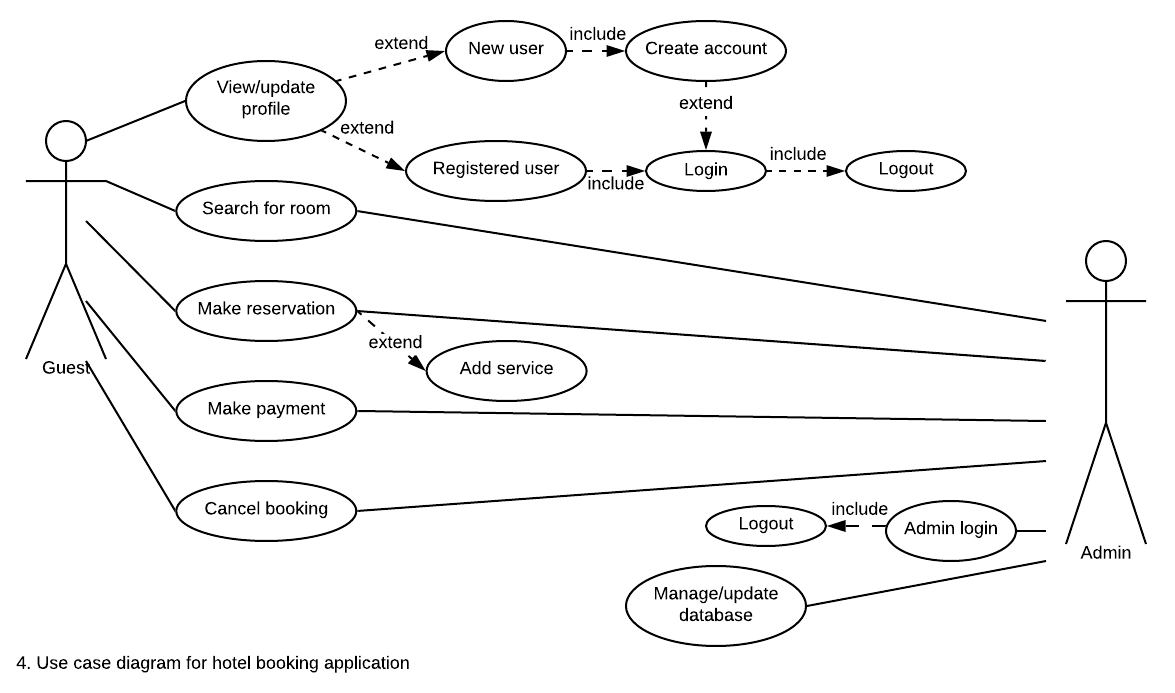
**3. UML diagrams for Hotel Booking Application**

**3.1. Use case diagram**

Diagram 3.1 presents use cases of actors in the application.

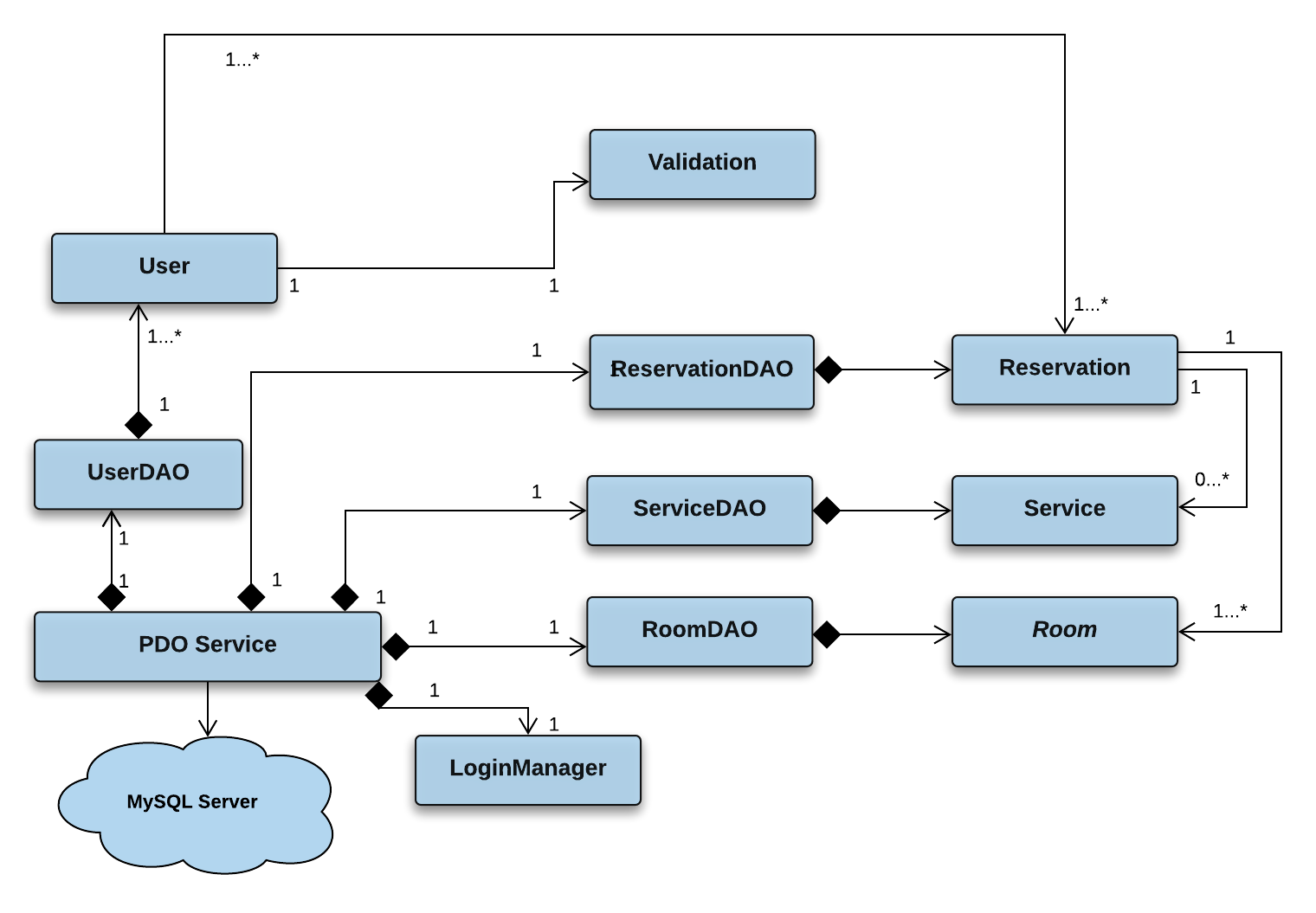
**Guest:** can login to the guest Account or register by entering his personal data. Then he can search for the room by choosing the room type and date, and make the reservation. The payment date is the reservation date. User can also cancel his reservation.

**Admin:** in case of this project it is authorized member of hotel staff. He can login to his account check the reservations and is able to delete and manage reservations.

****

**3.2. Class diagram with navigability arrows**

Diagram 3.2 shows classes diagram with navigabilityarrows.

****

*Notes:* For better readability *the diagram does not include Page class*

**3.3. Class diagrams with attributes and methods**

|  |
| --- |
| **Page** |
| + title: string |
| + header()  + footer()  + displayLogin()  + displayRegistration()  + displayUserDetails |

Diagram 3.3 presents classes with all attributes and method

|  |
| --- |
| **User** |
| - email: string  - password: string  - userType: string  - full\_name: string  - address: string  - postalCode: string  - city: string  - country: string  - phone: string |
| + setEmail()  + setPassword()  + setUserType()  + setFullName()  + setAddress()  + setPostalCode()  + setCity()  + setCountry()  + setPhone()  + getEmail()  + getPassword()  + getUserType()  + getFullName()  + getAddress()  + getPostalCode()  + getCity()  + getCountry()  + getPhone()  + verifyPassword() |

|  |
| --- |
| **Validate** |
|  |
| + validateInput() |

|  |
| --- |
| **ReservationDAO** |
| - db |
| + initialize()  + createReservation()  + getReservation()  + getReservations()  + getUserReservations  + updateReservation()  + getReservationList() |

|  |
| --- |
| **Reservation** |
| - reservationNr: int  - roomType: string  - email: string  - roomNr: int  - startDate: date  - endDate: date  - paymentDate: date  - finalPrice: float  + SALES\_TAX: float |
| + setEmail()  + setRoomType()  + setStartDate() |
| + setEndDate()  + setPaymentDate()  + setTotalPrice()  + getReservationNr()  + getRoomType()  + getEmail()  + getRoomNr()  + getStartDate()  + getEndDate()  + getPaymentDate()  + noOfDays()  + calculateTotalPrice() |

|  |
| --- |
| **RoomDAO** |
| - db |
| + initialize |

|  |
| --- |
| **ServiceDAO** |
| - db |
| + initialize()  + getService()  +getServices() |

|  |
| --- |
| **Service** |
| - serviceNr: string  - type: string  - price: float |
| + setType()  + setServiceNr()  + setPrice()  + getServiceNr()  + getType(  + getPrice() |

|  |
| --- |
| ***Room*** |
| - roomNr  - roomType |
| + setType()  +getType() |

**4. Meeting minutes**

2h/week

**5. User manual**

**References:**

Deitel P. Deitel H. Java how to program, 9th edition: Chapter with UML Class Diagrams

<https://www.w3schools.com/php>

[CSIS-3280](https://douglascollege.blackboard.com/webapps/blackboard/execute/courseMain?course_id=_81000_1) Course materials.