Applied Research

Individual Track Assignment

16/05/2022 Eindhoven
Version: 1.1

Esther Wolfs: 3329984

Tutor:
Tim Kurvers
Márcio Paixão Dantas

Version history

Version	Date	Author(s)	Changes	State
0.1	16/05/2022	Esther Wolfs	Move everything research related from design document to this document	Finished
1.0	17/05/2022	Esther Wolfs	Answering the research questions	Started
1.1	17/06/2022	Esther Wolfs	Add conclusion	Finished

Contents:

1. Introduction

This document will be about applied research. In the first paragraph of chapter 2 the main question will be explained. The main question will be divided into smaller sub questions, that together will answer the main question. To answer these questions I will be using certain research questions that will be specified in each sub-question paragraph. The last chapter will be the conclusion of my research, here you will find the answer to the main question.

2. How should the data for reservations be stored?

Reservations are a big part of this project. Users should be able to make reservations for rooms they select, after applying search filters. These reservations should then be stored in a database. These questions will give answers on how to store this information, so that it can be accessed by the relevant users.

2.1 What database to use

To start off this research it is important to figure out what kind of database to use, as there are many different kinds, all of which have different use cases. The relevant research methods for this question that will be used are literature study.

Relational databases store data using multiple related tables, the data is stored in rows and columns. SQL (Structured Query Language) is the most common language for performing CRUD operations. Relational databases work well with structured data and are very reliable. Some examples are Microsoft SQL Server, Oracle Database and MySQL (Matillion, 2020).

Non-relational databases are all databases that don't use SQL as its primary language. Data is a NoSQL database that does not have to conform to a predefined schema. It works well for organisations that want to store semi-structured or unstructured data. One advantage is that you can make changes to the database without it affecting the application. For example Apache Cassandra, MongoDB or CouchDB (Matillion, 2020)

A cloud database is any database that is designed to run in the cloud. It is low maintenance and flexible. (Matillion, 2020).

The best kind of database for this project is a relational database, because the data is structured and stored in different tables that relate to each other. A local MySQL database will be used.

2.2 What information is relevant

To find out what information is relevant, the problem needs to be stated. Problem analysis is a good research method for this. Why does the application need to store information, when does it need to retrieve this data.

For every reservation, there is a lot of necessary information. These include the date the reservation is made, the check in date, the check out date, what kind of room(s) the guest

wants to book. With this information we can determine the total price of the reservation. All of this needs to be linked to a guest. The status of the reservation is also important to store, so the availability of the rooms can be updated and the guest can get a refund if the reservation gets cancelled.

Every time a guest makes a search for a room the application needs to check for available rooms for the specified dates. Whenever a guest or employee wants to manage a booking the details of this specific reservation need to be retrieved.

2.3 What user data should be stored and how should it be stored

Stating the problem is a good method to find out what data should be stored related to the users of the application, to find out how to securely store account information literature study is used.

This application has two different kinds of users, both of which have different functionalities. The employee and the guest. The basic information about users that needs to be stored are their name and an email address. For employees the date of birth is also stored. Guests have a list of their reservations, this is done by a one to many relationship with the reservation table, because one guest can have multiple reservations.

To use the main functionalities of the application, the user needs to have an account. Employees need this to manage bookings to perform their daily work tasks. Guests are asked to log in whenever they want to make a reservation. When a new employee is hired and they are added to the system, a user account is automatically created for them that is linked to their data. A guest can sign up for an account.

To log in to the account, the software needs to authenticate the users. This process verifies that the person is whoever they say they are. Authorisation verifies that a user has the right permission to perform a specific task or access specific data. Authorisation is done after the authentication process. (Fernigrini, 2022)

Passwords are the most common methods to confirm the identity of a user. Storing a password in plain text is not a good idea, because everyone that has access to the database can see the password of the users. The data that needs to be stored for the users consists of the following:

- Account ID
- Username
- Encrypted password
- Email
- First Name
- Last Name

(Fernigrini, 2022)

In this application the information will be split into two seperate tables: one that has the personal information like the email and the name, and table that stores the username and

encrypted password. These tables are connected with a one-to-one relationship; one user account is linked to one person.

To differentiate between the users, a list of roles will be given to each user. These roles are stored in another table in the database, with a many to one relationship with the user account table. There are three role types in this application: a customer that can perform CRUD operations on their own reservation and user account, an employee that can perform CRUD operations on all reservations and their own user account and an admin user that can perform CRUD operations on all user accounts. An employee can have both the employee role, as well as an admin role. This is why a many to one relationship is necessary.

2.4 How to connect a room to the reservation

The method used to answer this question is problem analysis.

The two most important parts that are needed for this application are the availability of the rooms and the reservations. The question is when to connect these two together.

One option is giving a reservation a specific room the moment the guest makes the reservation, but this can give some problems. What if a guest made a reservation 6 weeks in advance and they have been assigned room 27 the moment they made the reservation, but the week before they are supposed to arrive the shower breaks and the room needs to be repaired. Or even worse, a crime has happened in that room and the police need to investigate. This could take weeks, even months, leaving the guest with no room to stay in.

A better option would be to assign a specific room to a reservation when the guest wants to check in. To do this the room type can be added to the reservation the moment it is made, like a single room or double room. Then whenever the guest arrives and the employee helps them check in, a specific room with a room number is then assigned to this reservation and given to the guest.

2.5 How to store the availability of a room

To answer this question the methods that will be used are problem analysis and community research.

The hotel only has a certain number of rooms available. Whenever a guest makes a reservation, they have to select the dates and a list of available rooms is shown. The guest then selects a room and completes their reservation. The availability of the rooms need to be stored in the database, so the application knows which rooms to show.

Because the guests make a reservation for specific dates, and all of these dates can vary and the hotel has different room types, it can be a bit difficult to keep track of the availability of all the rooms for every date.

To find out how to store the availability of a room, it is necessary to know when exactly in the booking process a specific room is connected to a reservation. We need a new table in the

database to store all the specific rooms. These rooms are connected to the “normal” room table, that has all basic information like the type of the room and the capacity. In this table the information specific to the room is stored, like the room number, the price per night and if the room is available.

One way to store the availability of each room type is to create a database table that has a row for every single day, for each room type, with a number of available rooms. But if the hotel has 6 different types of rooms and you can make a booking a year in advance this would mean that the table has (6*365) 2190 rows. Whenever a guest wants to search a double room for 4 nights the system needs to check for every single one of those dates if it still has a room available. If the guest decides to make the reservation, the number of available rooms for those dates need to be updated.

Another way is to count the reservations that have the specified room type and that have a check in date before the given check out and a check out date after the given check in date. This should return a list of rooms that are already reserved, making them unavailable. The amount of unavailable rooms can then be counted and subtracted from the total number of rooms. In the database this would be stored in 3 different tables, one table that has all the information regarding the reservation (check in, check out), one table that has the room type that stores the total amount of the rooms in the hotel, and another table that connects these two together: a table that has the reservation id and the room type. For every room in a reservation, there will be a row in this table connecting the room to the reservation. (stackexchange, 2018)

SELECT COUNT(r.id) FROM goldskye_hotel.reservation_room as rr INNER JOIN goldskye_hotel.reservation as r on rr.reservation_id = r.id WHERE rr.room_id = 2 AND r.check_in < '22-06-15' and r.check_out > '2022-06-10'

This is the query that can be used for this. It counts the amount of reservations that have a specific room reserved for a specific date. The rr.room_id is the room that is selected, for example a double room. The r.check_in should be less than the check out date and the r.check_out should be greater than the selected check in. The count of the reservations should be subtracted from the total amount of the selected rooms that are in the hotel.

3. Conclusion

In conclusion, there is quite a bit of data that is necessary to store a reservation. Starting off with the basic data, like the check in/out dates, the amount of guests, the room(s) and the total price. The right guest should also be stored and linked to the reservation.

Before anyone can make a reservation they need an account. The user data like the username and password are stored in the “user” table. The name and email of the guest are stored in the “guest” table. When a guest makes a reservation, the reservation date, check in date, check out date, amount of guests, total price, check in status and the guest id are stored in the “reservation” table. The guest id in this table is used to link the reservations to

the correct guest. The rooms that are added to the reservation are stored in a separate table, the “reservation_room” table. This links the correct rooms to the reservation. Each Reservation entity has a list of ReservationRoom entities that contain a Reservation and a Room entity. The Guest entity has a list of Reservation entities. These are joined by OneToOne, OneToMany or ManyToOne joins.

Unfortunately I didn’t have enough time to implement the functionality to check the availability for the rooms for the selected dates.

References

Fernigrini, L. (2022, February 3). *How to Store Login Data in a Database*. Vertabelo. Retrieved May 16, 2022, from https://vertabelo.com/blog/authentication-data-storage/

Polito, J. (2020, November 18). *The Types of Databases (with Examples)*. Matillion. Retrieved May 16, 2022, from https://www.matillion.com/resources/blog/the-types-of-databases-with-examples

sqlbot, m. (2018, January 14). *mysql - checking availability of rooms*. Database Administrators Stack Exchange. Retrieved May 17, 2022, from https://dba.stackexchange.com/questions/195342/checking-availability-of-rooms