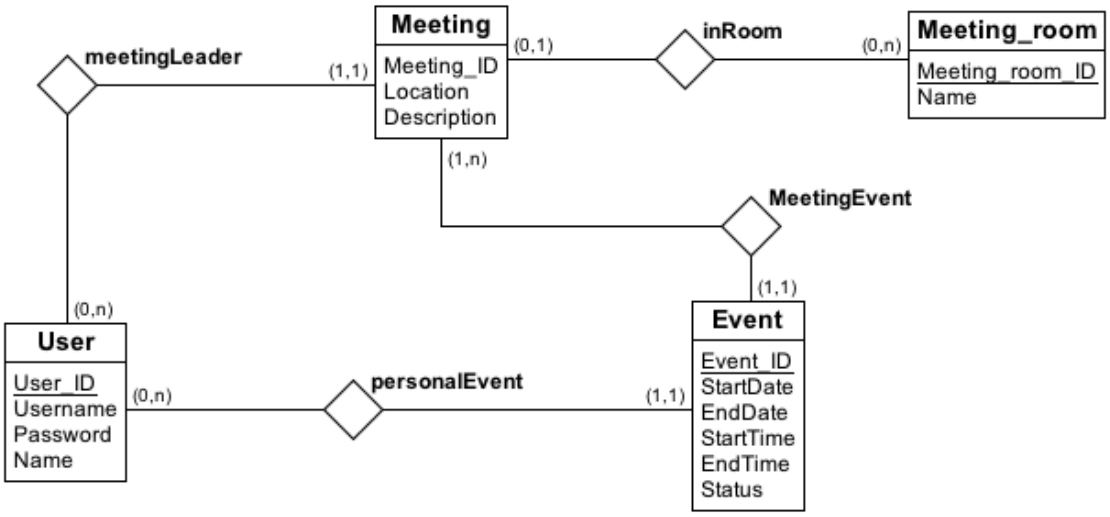
Database 1 - Conseptual model

GROUP 38

In the following document, we will explain how we plan to implement the database in use with our calendarsystem. We have made an EER diagram, and we will use this to explain how we plan to fulfill the given requirements.



System Requirements

Requirement 1 - Logging in

We have a user entity that has a unique user_ID and username , a name and a password. This enables us to have employees log into their unique account using their username and password. The user is also associated with a real name, so the other employees can easily identify who the user is.

Requirement 2 - Create new appointment

When a user creates a new appointment, a new event is created and is linked to the user by the personalEvent relation. This is what will be shown in the users personal calendar. When creating an event, a meeting entity is also created and the event is linked to the meeting with the MeetingEvent attribute. The meetingLeader is set to be the user who created the event. An appointment as explained in the system requirements is here defined as a meeting with only one linked event. The status attribute is used by the system to identify if an event is accepted, declined, not responded or cancelled. Since the user is the one who made the event himself, the status here defaults to accepted.

Requirement 3 - Delete appointment

We look for the correct event in the personalEvent relation for the given user by querying for the event_ID. When deleting an event we must also make sure to delete the corresponding meeting if the user is the only one in the MeetingEvent relation.

Requirement 4 - Edit appointment

Here we simply update the users chosen event based on event_ID. Alternatively, if we want to edit location or description, we simply edit the correct meeting if we're the one defined as the meeting leader.

Requirement 5 - Create new meeting

This works exactly like making an appointment/event, except now the MeetingEvent table contains multiple appointments, making it a meeting. The status attribute will be marked as accepted for the meeting leader and not responded for everybody else.

Requirement 6 - Receive invitation to meeting

When someone creates a meeting which contains another user, a new event will automatically be created for the invited user and linked with the users personalEvent. The client will check the contents of the personalEvent table based on what user that is logged in at the moment. If it finds any events that have a status as not responded, it will interpret this as a new invitation and add it to the users messages list. Any invitations that are marked with an accepted status are loaded into the calendar system UI. If a user decides to accept an invitation, then it will be loaded on the next server refresh. The meeting leader, aswell as anyone else, will be able to watch the status of all the invited users by clicking on the event. The database will then provide a list of the status of all the invited users.

Requirement 7 - Edit meeting invitation

When a meeting leader makes changes to a meeting, only his own event is affected at first. All the events related to the meeting, except the leader, will have their status attribute reset to not responded, regardless of previous input. As soon as the other users clients are updated with the server, the database will compare their personalEvent to the personalEvent of the meeting leader. If there is a difference, the event that is stored in the users personalEvent will be shown until the user either accepts the change or declines. The user will also get a new message informing them of the change made. As before, any user, including the leader, may click on the details of the meeting to see the status of all the invited users.

Requirement 8 - Cancel meeting

This works similarily to how edit meeting is handled. The status of the meeting leaders event is changed to cancelled. This change will be visible to all the other users the next time they update with the server. The users will receive a new message informing of the deletion and asked to confirm reading it. Only after confirming will their event be changed aswell, removing it from their personal calendar.

Requirement 9 - Cancel meeting confirmation

This works similarily to requirement 8, only this time only the user who deleted the invitation gets his status changed to cancelled. The leader will be informed of this the next time he updates with the server.

Requirement 10 - Book meeting room

The meeting entity has a inRoom relation that links to a meeting room. If he wants to, the meeting leader may choose to book a meeting room. The database will then check the inRoom relation to see if another meeting uses any of the meeting rooms at the time, and will return the ones that are not in use.

Requirement 11 - Show calendar

The server returns the personalEvent information from a database query for a given user and a given week.

Requirement 12 - Track meeting invitations

The server returns a database query using the aggregate function COUNT to count the number of cancelled, not responded and accepted on MeetingEvent.

Requirement 13 - Show other calendars

In the GUI, a user is able to use a checkbox to check all the users calendars he/she wants to view. The server then returns a database query that returns all the accepted events for a given time for the given users.