System Design: Design Family Calendar

Requirements based on the website originally by jagrutigoswami402
www.geeksforgeeks.org/system-design-design-google-calendar/amp/

A family calendar *is like a special calendar that you can use on your computer. You can use it to remember important things, like birthdays, appointments, and events. You can also set reminders to help you remember when something is coming up soon. You can also share your calendars with your family. It's like having your own special schedule that helps you keep track of everything that you need to do. With a family calendar, you can quickly schedule events and get reminders about upcoming activities, so you always know what's next.*

Requirements of the System:

Functional Requirements:

- **1) Book an event:** In a calendar, you can add an event with other family members. It can be a single user or multiple users.
- **2) Check availability of family members:** While you are scheduling an event, you want to figure out in that particular time family members are available or not.
- **3) Invite family members to the event:** The user should be able to invite other family members to the event. The number of family members invited for a scheduled event can be one or more.
- **4) Request RSVP:** Before sending invitations to family members, you want to know if that particular family member is already blocked or available. These types of responses can be handled by requesting RSVP.
- **5) Event reminders:** Getting a notification before the time for a particular event with a particular family member about the event description and all those details would help prepare for the event beforehand.
- **6) Modify an event:** This includes making changes to an existing event that you have created. If you want to change an event that you have been invited to, you should ask the event creator for the change.
- **7) Cancel an event:** This includes deleting or removing an event that you have created or have been invited to. If you have not created the event, the event will be deleted from your calendar only.
- **8) Lookup for all events in the calendar:** This includes viewing a list of all events that have been designated as events.
- **9) View Calendar:** This allows the user to see his or her scheduled events and appointments in a visual format which can be a monthly, weekly, or daily view, and it includes details such as the date, time, location, and description of each event.

Data Flow

Data flow for a particular user viewing the calendar:

1. The user opens the calendar application on his or her computer.

2. The calendar application sends a request to the database server to authenticate the user and establish a connection.

3. The database server verifies the user's account and sends a response to the calendar application.

4. The calendar application sends a request to the database server to retrieve the user's calendar data.

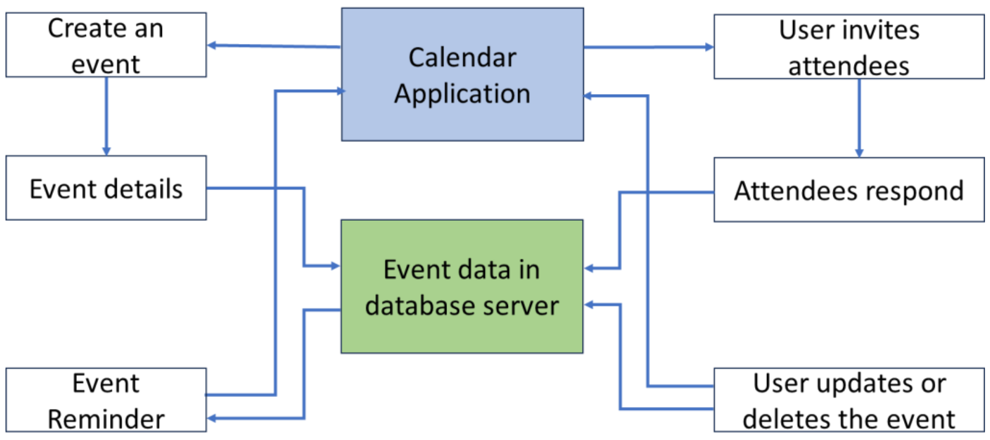
5. The database server retrieves the user's calendar data from the calendar database.

6. The calendar application displays the calendar data to the user through an interface.

7. The user interacts with the calendar by adding, modifying, or deleting events or changing settings.

8. The database server processes the requests and updates the user's calendar data in the calendar database.

Data flow diagram of the entire system



Data flow diagram of the entire system

1. The user creates an event in his or her calendar by providing event details such as the event name, date, time, description, and location.

2. The event data is stored in his or her calendar database, which is hosted on the database server.

3. The user invites attendees: If the event has attendees, the user can send out invitations to those attendees using their calendar accounts. Each invitation is stored in the attendee’s personal message box.

4. On checking the message box and receiving an invitation, attendees respond to it: They can choose to accept, decline, or tentatively accept the invitation. Their responses are recorded in the event data.

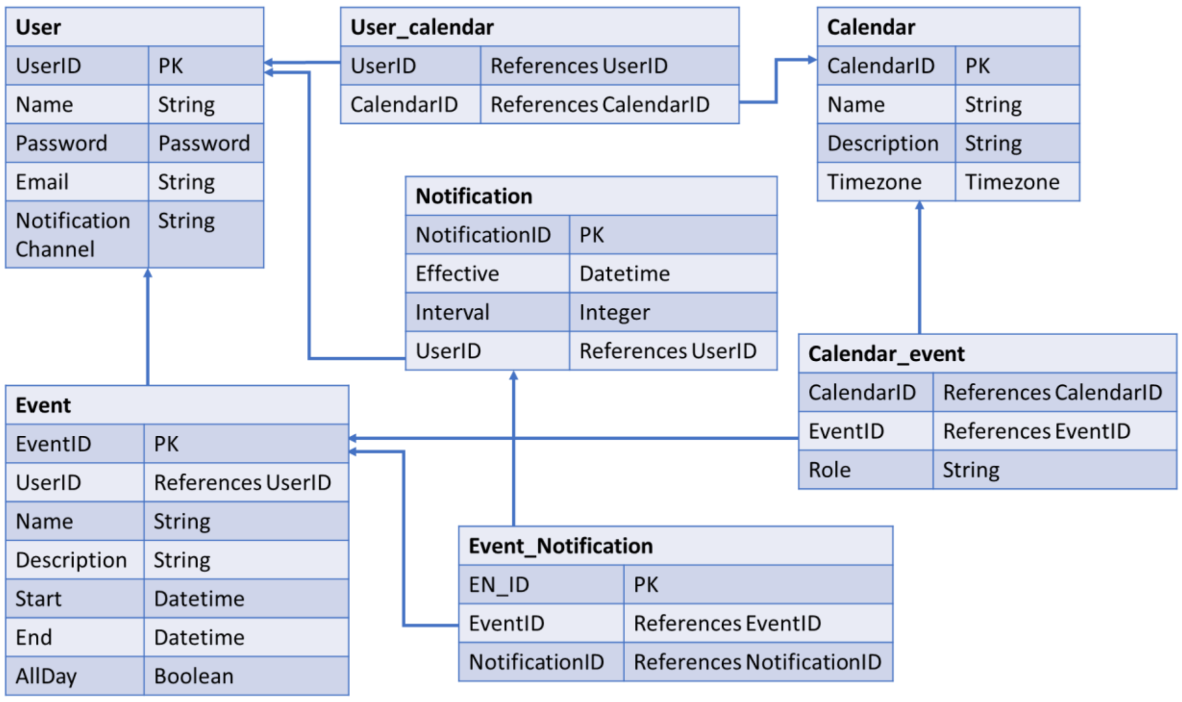
5. Event reminders: If the user has set reminders for the event, these reminders are sent to each attendee’s personal message box at the appropriate times, based on the user’s preference, for example, setting a reminder every 15 minutes, every 30 minutes or none.

6. Event updates: If the user makes changes to the event, such as changing the date or time, attendees are notified of the changes.

7. Event deletion: If the user deletes the event, it is removed from his or her calendar account and attendees are notified that the event has been canceled.

Database schema and design

You would have the following tables to work along:



User	
UserID	PK
Name	String
Password	Password
Email	String
Notification Channel	String

User_calendar	
UserID	References UserID
CalendarID	References CalendarID

Calendar	
CalendarID	PK
Name	String
Description	String
Timezone	Timezone

Notification	
NotificationID	PK
Effective	Datetime
Interval	Integer
UserID	References UserID

Calendar_event	
CalendarID	References CalendarID
EventID	References EventID
Role	String

Event	
EventID	PK
UserID	References UserID
Name	String
Description	String
Start	Datetime
End	Datetime
AllDay	Boolean

Event_Notification	
EN_ID	PK
EventID	References EventID
NotificationID	References NotificationID

Database Schema Design

It is basically about understanding the requirement from a functional perspective and deciding how you are going to utilize the database and how are you going to design it.

API design of the system

Let's do a basic API design for your services:

1. Display info for each user.
GetUserInfo(userID)

For Scheduled Events

2. Get events for the user for the specific period.
GetEvents(userID, date, days)

3. Creating an event
CreateEvent(userID, startTime, endTime, users, subject, description)

e.g., CreateEvent(userID, startTime, endTime, ["abc", "xyz"], "Important Event", "Let's go out for dinner")

4. Updating the event
UpdateEvent(eventID, startTime, endTime, users, subject, description)

e.g., UpdateEvent(eventID, startTime, endTime, ["abc", "xyz"], "Event Cancelled", "Let's go out for dinner")

For Reminders

A reminder consists of:

- When to show the reminder, expressed in intervals before the start of the event.

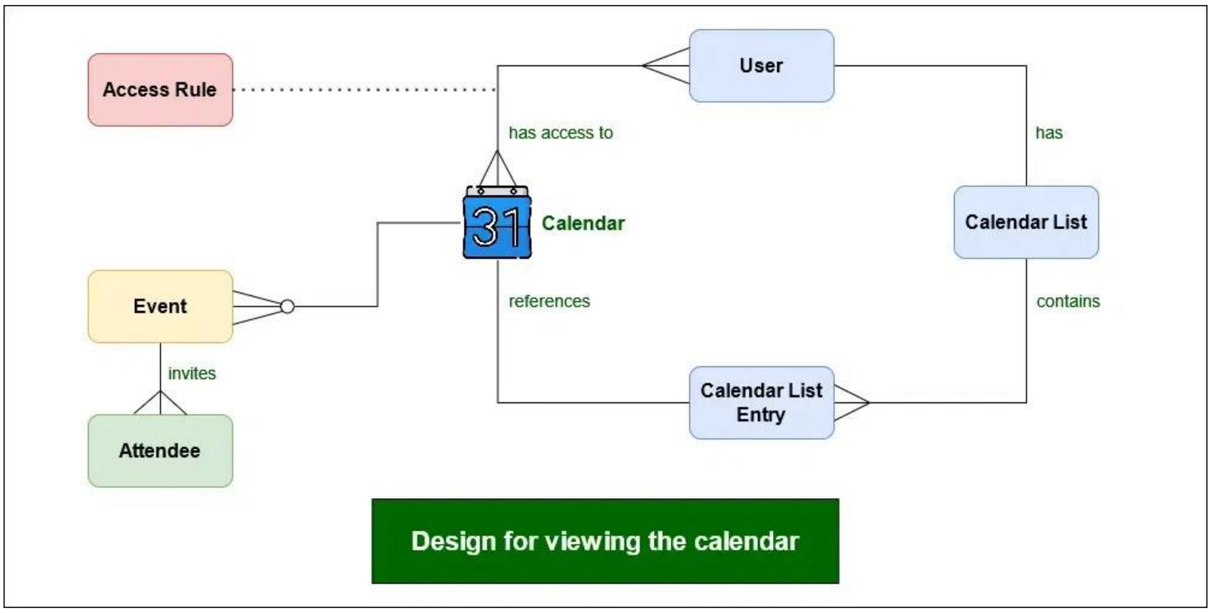
Reminders can be specified for whole calendars and for individual events. Users can set *default reminders* for each of their calendars; these defaults apply to all events within that calendar. However, users can also override these defaults for individual events, replacing them with a different set of reminders. Overrides can be set if and only if using *default reminders* is set to false.

5. Setting up the default event reminder
SetReminder(userID, useDefault, method, effective, interval)

The delivery methods offered by the system are pop-op on the calendar application or email.

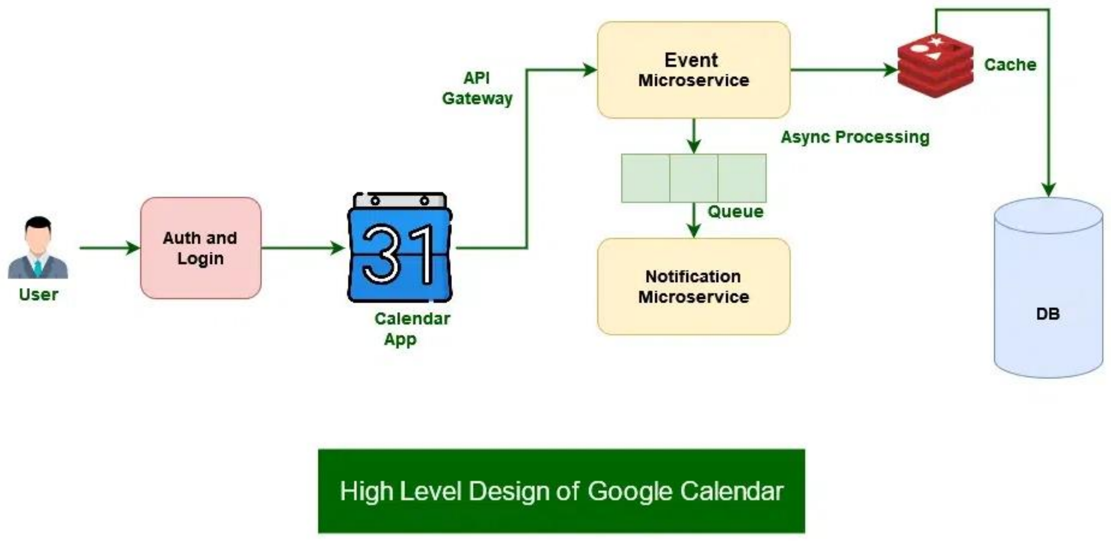
High-Level Design of the System

Design for a particular user viewing the calendar and the related events to that user



Design for a particular user viewing the calendar and the related events to that user

Overall High-Level Design of the System



The high-level design of the system

Microservices Used (Optional)

Event Microservice

Let’s have a look at how a user can create an event for family members:

- A request for a new event comes to the backend. After validation of each field, the backend makes a call to the event microservice to create an event.
- This microservice makes an entry in the events table with all the details from the request body.
- For each user in the invitees’ list, add the entry for each user in the invites table.
- Send the list of invites to the notification generation microservice to notify all users of the event info.

Notification Microservice

- This is very simple. It reads from the queue all the invites and sends a notification to the users.

Conclusion

In conclusion, designing a system like a family calendar needs careful consideration of various components and features which would be crucial to its functionality. The system must be scalable and flexible, allowing for easy integration with other applications and platforms. It should also be secure and reliable.

Overall, a well-designed system for the family calendar should provide users with a reliable and efficient tool for managing their daily schedules and events, while also should be allowing for easy collaboration and integration with other applications and platforms.