Database Schema

Tables

1. `users`

Stores user details and authentication credentials.

Column	Type	Constraints
id	SERIAL	PRIMARY KEY
username	VARCHAR(80)	UNIQUE, NOT NULL
password	TEXT	NOT NULL

2. `events`

Stores details of events.

Column	Type	Constraints
id	SERIAL	PRIMARY KEY
name	VARCHAR(100)	NOT NULL
date	TIMESTAMP	NOT NULL
location	VARCHAR(200)	NOT NULL
description	TEXT	NULLABLE

3. `agenda`

Stores schedules for events.

Column	Type	Constraints
id	SERIAL	PRIMARY KEY
event_id	INTEGER	FOREIGN KEY REFERENCES `events(id)`, NOT NULL
title	VARCHAR(200)	NOT NULL
time	TIMESTAMP	NOT NULL
speaker	VARCHAR(100)	NULLABLE

4. `payments`

Stores transaction details for event registrations.

Column	Type	Constraints
id	SERIAL	PRIMARY KEY
user_id	INTEGER	FOREIGN KEY REFERENCES `users(id)`, NOT NULL
event_id	INTEGER	FOREIGN KEY REFERENCES `events(id)`, NOT NULL
amount	FLOAT	NOT NULL
status	VARCHAR(50)	NOT NULL (e.g., 'pending', 'completed')

5. `messages`

Stores real-time chat messages.

Column	Type	Constraints
id	SERIAL	PRIMARY KEY
sender_id	INTEGER	FOREIGN KEY REFERENCES `users(id)`, NOT NULL
receiver_id	INTEGER	FOREIGN KEY REFERENCES `users(id)`, NULLABLE
event_id	INTEGER	FOREIGN KEY REFERENCES `events(id)`, NULLABLE
message	TEXT	NOT NULL
timestamp	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

Relationships

- **Users → Payments**: One-to-Many (a user can have multiple payments)
- **Events → Agenda**: One-to-Many (an event can have multiple agenda items)
- **Users → Messages**: One-to-Many (a user can send multiple messages)
- **Events → Messages**: One-to-Many (messages can be linked to events for networking)

This schema is optimized for scalability and real-time interactions. Let me know if you need modifications! 🚀