CS 586 Winter Term Project - Deliverable 1

Ryan Streur, streur@pdx.edu

Subject Area

The subject area for this project is application development and party planning. I'm writing a web application which allows users to plan events and create invitations to those events for other users. The main tables I've got planned out are `user`, `session`, `role`, `event`, `invitation`, and `request_log`.

Users will be able to create accounts and log into those accounts. When a user logs in a session will be created in the system, and the session ID will be issued to the client. The client will then include the session ID as a Bearer token in further requests to make authenticated requests. Authenticated users will be able to create events, and those events will be visible to them and other users based on their roles. Users with the "owner" role will be able to edit events, "owners" and "organizers" will be able to send invitations, and "guests" will be able to respond to invitations and edit their responses. All HTTP requests will be logged in a `requests_log` table.

I'm planning on having a view called SessionUser which accepts a session key (UUID) and returns both the session information and user information.

Questions

Since this is going to be a web application, I'm going to include queries which mutate the database as well as those which retrieve information:

1. Create a new user from an email address and password (after hashing the password, of course).
2. Retrieve a user from the database by email address.
3. Create a new session given a UserId.
4. Retrieve session and user information given a session ID.
5. For a given SessionId and new event data, create a new event with an ownership role for the corresponding user
6. For a given SessionId, what are all the events that the corresponding user owns?
7. For a given SessionId and event data which corresponds to an existing event, does the user have the right to update that event?
8. For a given SessionId for an authorized user and event update data, update the specified event.
9. For a given SessionId for an authorized user and an event ID, display the event information.
10. For a given sessionId for an authorized user and an event ID, delete the corresponding event and any roles and invitations associated with it.
11. For a given SessionId and an invitation ID, retrieve information about the invitation.

12. For a sessionID for an authorized user and an invitation ID, delete the invitation.
13. For a sessionID for an authorized user, an information ID, and a response, update the invitation response.
14. For a SessionId for an authorized user and an event ID, retrieve a list of invited users and their responses.
15. For a given HTTP Request, log that request to the database.
16. For a given URL, how many requests have been made to the server?
17. What are the IP addresses of the most frequent requesters?
18. What is the most-visited URL on the API?
19. How many distinct IP addresses visited the site?
20. Is the site more frequently visited by authenticated or anonymous users?

Data Source, Size, and Backup Plan

I intend to deploy a logging-only prototype of this site as soon as I get the request logging behavior working, hopefully sometime this weekend. I'll probably put a link to the site on my GitHub. Once I do, it'll likely start logging requests from bots, and I figure there should be at least a couple hundred of those by the time the project is due. I would also provide Jesse (or whichever grader needs it) with credentials to the production database. If this plan isn't acceptable, my backup plan is to use data from the Portland Open Data Portal - I'm sure I could come up with a number of interesting questions to ask about the open data about our fair city.