CSC 440 (001) Database Management Systems - Group Project Proposal

Team Chupacabra

Caleb Rollins, Matthew Welker, Finn Bacheldor, Alex Hill, Jakari Robinson

---

Project Summary

Our team is proposing a database system that supports the operation of an IoT device for opening the door to an apartment. A functional codebase that deals with the action of opening the door already exists from a team member's personal project. As of now the code does not have any interaction with a database, and only responds to one command. We will be extending this to be a more robust discord bot that stores critical information in a MySQL database. The most important entities we expect to have in our database will be for User, UserInstance, Door, EntryPhoto, OpenLog, PenaltyLog, ScheduledEvent. A User entity will store a user's unique screen name and a hashed password. A UserInstance will have a reference to this User entity and store information specific to a particular Door for the referenced user, such as their permissions (admin, resident, or guest). The user interacts with the system through the Discord Bot chat, writing commands to authenticate, add users if an admin, and open the door. This resembles a command line interface but done through Discord. The system will support multiple IoT door devices and will maintain records of all doors each user has been given access to along with their current status of offline or online. Each time someone enters the room and opens the door, a picture of them along with entry information (username, time, door, etc.) will be logged in the database for accounting purposes and administrative review. Each time the user tries to open the door unsuccessfully (not authenticated, no permissions, spamming, wrong time of day) their attempt will also be logged. We will have the ability for an administrator to add entries to a table of scheduled events (like parties) that will let all authenticated users through the door regardless of their typical permissions.

Preliminary Entities

- user
  - discordUUID(varchar)
  - hashedPassword (varchar)
  - defaultDoor (foreign key)
  - developer (boolean)
- userInstance
  - userType (foreign key)
  - user (foreign key)
  - score (int)
  - door (foreign key)
- userType (enum: admin, resident, guest, etc.)
  - category (varchar)

    - startTime (time)
    - endTime (time)
- door
    - displayName (varchar)
    - location (varchar)
    - owner (foreign key)
    - doorStatus (foreign key)
- doorStatus
    - status (online offline?) (varchar? enum?)
    - ip address (varchar)
    - dns (varchar)
    - port (varchar)
- entryPhoto
    - fileName (varchar)
    - timestamp (time)
- openLog
    - openType (enum: manual, bot, scheduled)
    - timestamp (time)
    - userInstance (foreign key)
    - door (foreign key)
    - photo (foreign key)
- penaltyLog
    - penaltyType (enum: noPermission, spam)
    - userInstance (foreign key)
    - scoreDeducted (int)
    - door (foreign key)
- scheduledEvent
    - timeStart (time)
    - timeEnd (time)
    - name (varchar)
    - door (foreign key)
    - Invitation (foreign key)
- userToEvent
    - user (foreign key)
    - event (foreign key)

Brainstormed Ancillary Components

Discord bot commands:

1. `authenticate <password>`
2. `password <password>` (if you do not already have a password)
3. `help` (List commands with brief instructions on how to use them)

If authenticated:

1. `open <door>`
2. `add <user> <role> <door>` (requires admin for door)

3. `setpermission <user> <role> <door>` (requires admin for door)
4. `schedule event <event name> <time interval> <door>` (requires resident for door)
5. `update password <password>`
6. `defaultdoor <door>` (sets the default door for your user, so it can be ignored as an argument in other commands)
7. `list` (shows a list of doors you can access)
8. `events <door>` (shows a list of events for a given door, with more detail for residents and admins)
9. `myevents` (shows a list of events you are invited to)