Specification – Social Network “Cubegram”

This web application is a social network for speedcubers. Speedcubers are people who solve Rubik's cubes and other similar puzzles as fast as possible. My application will help users better track their results, share their achievements with others, create rooms, and enjoy their favorite hobby together with friends.

Functional requirements

Roles

The application distinguishes two roles:

- **User** - a person who can manage their account, use the timer, view other users results, create rooms, maintain their profile, track and analyze their attempts and results based on attempt charts and additional information.
- **Moderator/Admin** - can delete (ban) users who violate the rules, remove user posts that violate the rules, and remove results from the lists if they are considered suspicious. They can perform these actions both at the database server level and through the GUI, as well as manage other admins at the database level.

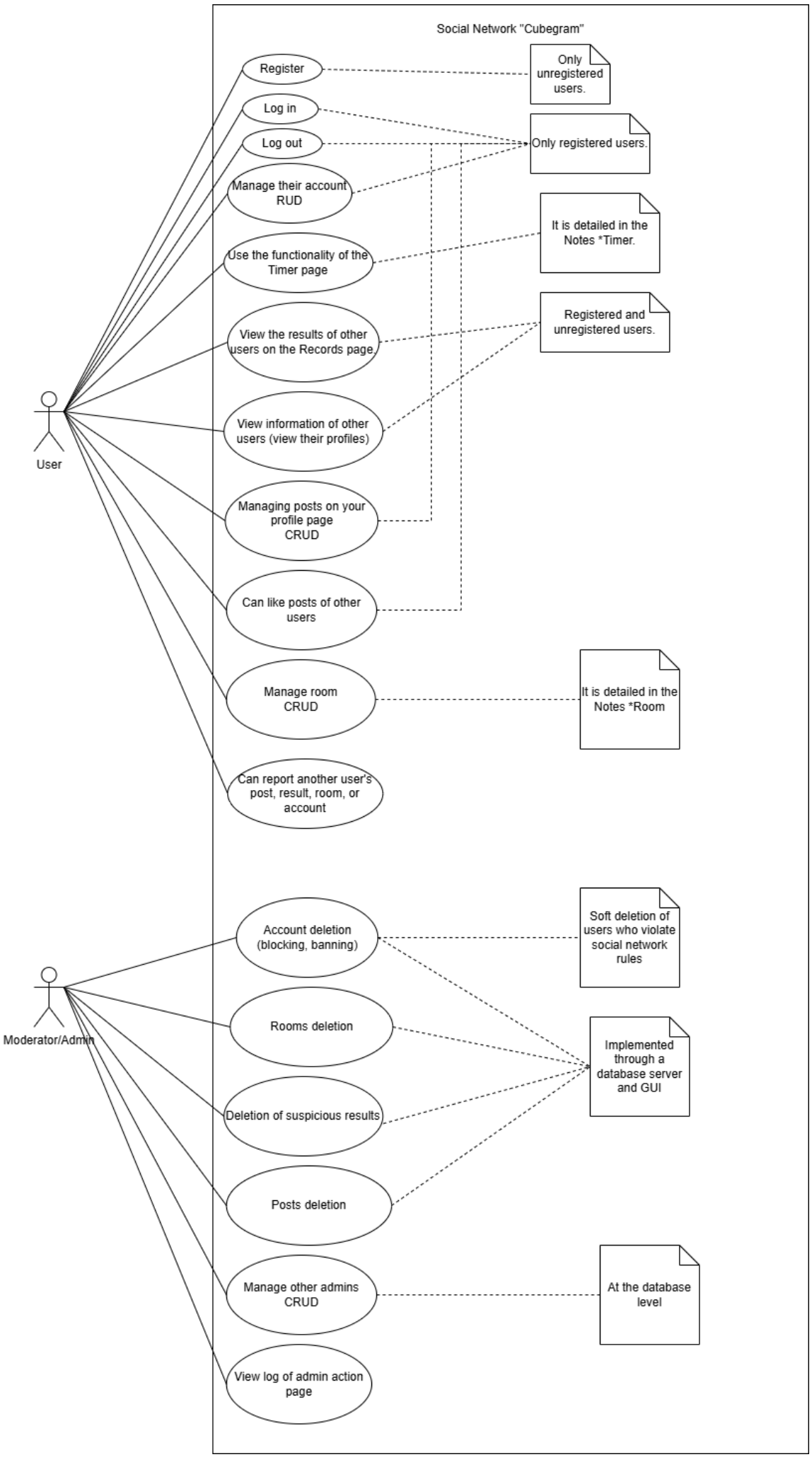
Use Case Diagram

Notes:

- **CRUD** = Create, Read, Update, Delete
- **Delete**: Some delete functionalities for the admin role may be implemented as a “soft”
- **Timer** - Timer page: This page provides the user with a timer, a **scramble** (algorithm for solving the Rubik's Cube), and the option to choose which Rubik's Cube they will solve (2x2, 3x3, 4x4, 5x5). The timer will also include additional features for speedcubers, such as +2 (penalty for an unsolved edge) and DNF (timer stopped but the puzzle was not solved). The user will also have access to their statistics, including: viewing all their attempts, their best attempt (record), their worst attempt, the average of all attempts, the average of the last 5, 10, 25, and 50 attempts, the total number of attempts, and a graph

of their attempts. In the graph, the Y-axis will show the time spent on solving the puzzle, and the X-axis will display the relevant time or date. The graph will be available in different modes: all attempts for the day, month, year, or all time. The X-axis of the graph will adjust according to the selected mode. Unregistered users will have limitations; all data will be stored in the session, and when the session is cleared, the data will be lost. For registered users, the data will be saved in the database.

- **Room** - Rooms page: This page displays rooms that users can create. When creating a room, the user must specify which puzzle will be solved and provide a name. After creation, both the room name and the selected puzzle can be edited. Only registered users can access rooms. To enter a room, a user must be manually added to the allowed list by the room owner. This ensures that only selected participants can join. The purpose of these rooms is to allow users to solve puzzles together with their friends and track their results collectively. They will have synchronized scrambles, creating a sense of competition. Additionally, information such as the average solving time of all attempts, a list of all attempts, best/worst attempts, and the average of the last 5/12 attempts for all users present will be available.
- **Reports -** Every registered user can report a post, a user, a room if it violates the social network's rules, or a record/attempt of another user if it seems suspicious. Subsequently, anything that has a complaint filed against it will be displayed in the Admin Panel, where admins can decide whether to delete or keep the reported content. Additionally, a Log page will be available, showing which admin deleted what and when.
- **Posts -** Every registered user can write, delete, and edit posts on their profile. Other registered users can like posts made by other users.
- **Records page -** A page where both registered and unregistered users can view the top 100 best results of other users with filters such as: male/female, disciplines, and country.
- The following functionality is optional:
    - All Admin functionality
    - Reporting posts by users
    - Reporting suspicious results


Social Network "Cubegram"
Register
Log in
Log out
Manage their account
RUD
Use the functionality of the
Timer page
View the results of other
users on the Records page
View information of other
users (view their profiles)
Managing posts on your
profile page
CRUD
Can like posts of other
users
Manage room
CRUD
Can report another user's
post, result, room, or
account
User
Only
unregistered
users.
Only registered users.
It is detailed in the
Notes *Timer.
Registered and
unregistered users.
It is detailed in the
Notes *Room
Account deletion
(blocking, banning)
Rooms deletion
Deletion of suspicious results
Posts deletion
Manage other admins
CRUD
View log of admin action
page
Moderator/Admin
Soft deletion of
users who violate
social network
rules
Implemented
through a
database server
and GUI
At the database
level

Data model

The following conceptual data model contains the entities, attributes and relations.

Entities and attributes

User

- A user identified uniquely by id. The user has their puzzle-solving attempts, and they can create 1 room. They also have posts associated with them.
- Attributes:
  - *password*: Stored as a hash value
- Constraints & rules:
  - The user must have a unique login and email.

Attempt

- Represents a user's attempt, the time it took for the user to solve the puzzle.
- Attributes:
  - *flags*: enum {none, +2, DNF}
- Constraints & rules:
  - If an attempt belongs to a user who is deleted or banned, it will be ignored when displayed on the pages.

Room

- Represents a room where users can be present.
- Attributes:
  - *users*: A set of users id who are in a room.
  - *password*: Stored as a hash value.
- Constraints & rules:
  - A room can have a maximum of 4 users.
  - A room has exactly one owner.
  - If the owner of a room becomes deleted or banned, the room ceases to exist.

Room_attempt

- Represents a user's attempt that is linked to a specific room.

Post

- Represents a post that is linked to a specific user.

Puzzle

- A trivial entity, represents a specific puzzle.
- Attributes:
    - *suspicious_result*: The time that serves as the upper limit for a attempt to be considered suspicious.

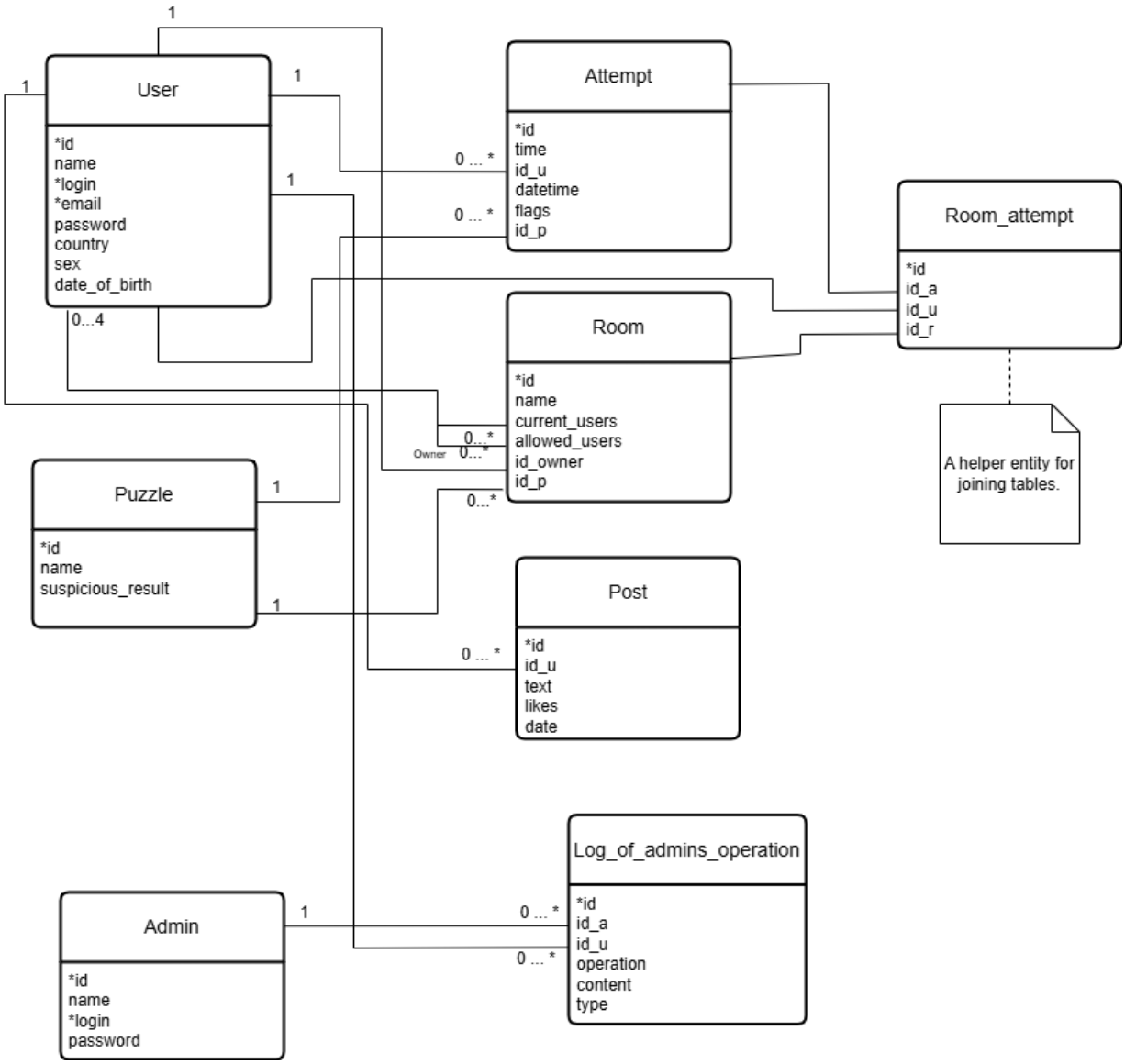
Admin

- Represents an admin, with a unique login for the admin.

Log_of_admins_operation

- Represents an admin's operation.
- Atributes:
    - *operation*: delete/banned.
    - *content*: Content that has been deleted or banned.
    - *type*: The type of deleted content. Can be user, post, attempt, room.

ER model



Architecture

- The application will be based on the client-server architecture and it will use the SPA (Single Page Application) approach.

Technological requirements

- Client-side: React 18, JavaScript, HTML5, CSS3, cubing.js
- Server-side: node.js 23, express.js 4.21.2, JavaScript
- Database: PostgreSQL 16
- Interface client - server: Rest API
- Hosting: render.com
- Supported browsers: Chrome, Firefox

Time schedule

Week 4

- Set up React, Start developing the frontend for the Timer and Profile pages.

Week 5

- Set up Express js, PostgreSQL, Create all tables in the database, implement authentication and registration, develop the frontend for the Records page, and complete the frontend for Profile page (if not finished earlier).

Week 6

- Write the backend for the Profile page (Manage posts CRUD), edit user data, and allow account deletion. Complete the timer page so that the data is handled through an API with the backend, except for the attempts graph.

Week 7

- Develop the frontend for the Rooms page and the Admin Panel.

Week 8

- Develop user search, complete the routing entirely, and implement the backend for the Admin Panel.

Week 9(Beta version)

- Backend development for the Rooms page. Deployment to hosting.

Week 10

- Rewrite the attempts graph to work with the backend.

Week 11(Final version)

- Testing, filter for the Records page.