Specification - Organization app

The goal of the project is to create an organizational application that allows users to efficiently plan tasks. The core features will include a calendar displaying events with the ability to add, edit, and delete tasks, as well as sending notifications about upcoming events. The application is designed primarily for students who need an effective tool to manage their academic workload, group projects, and deadlines.

Functional requirements

Roles

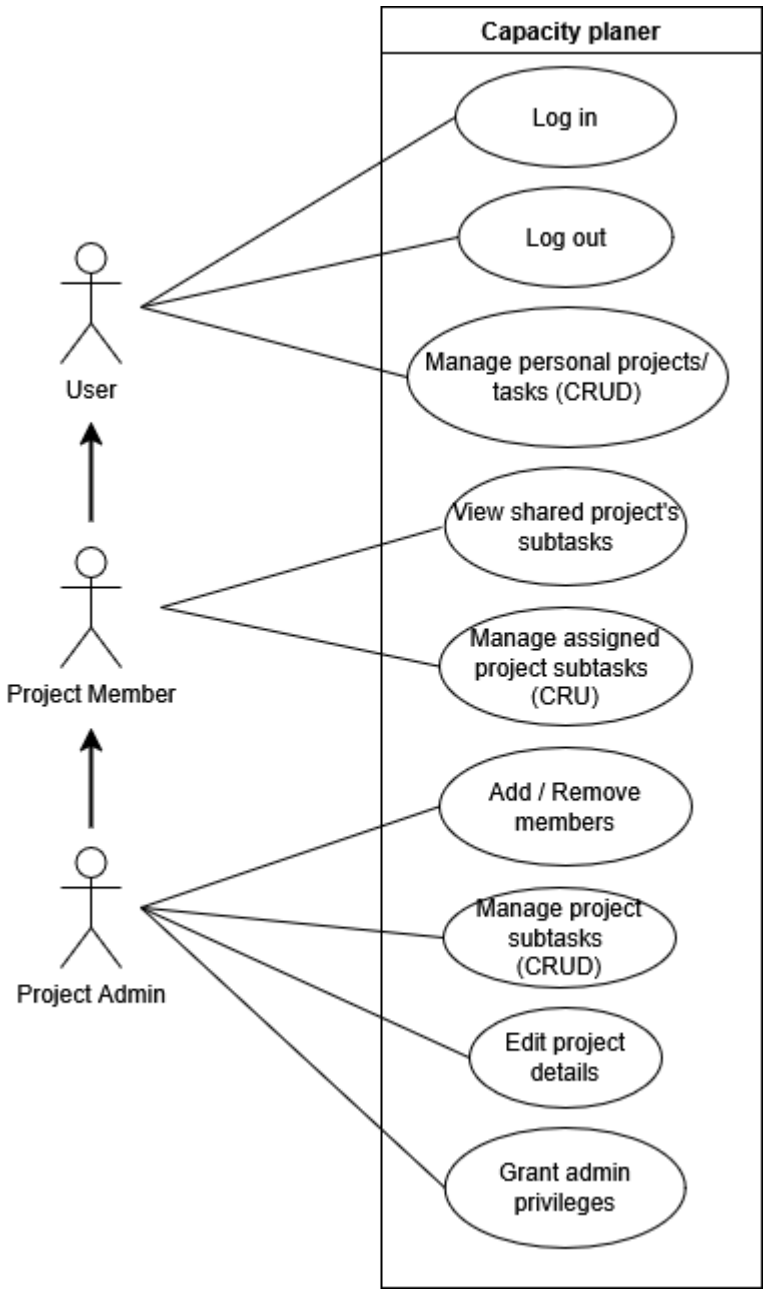
The application distinguishes following roles:

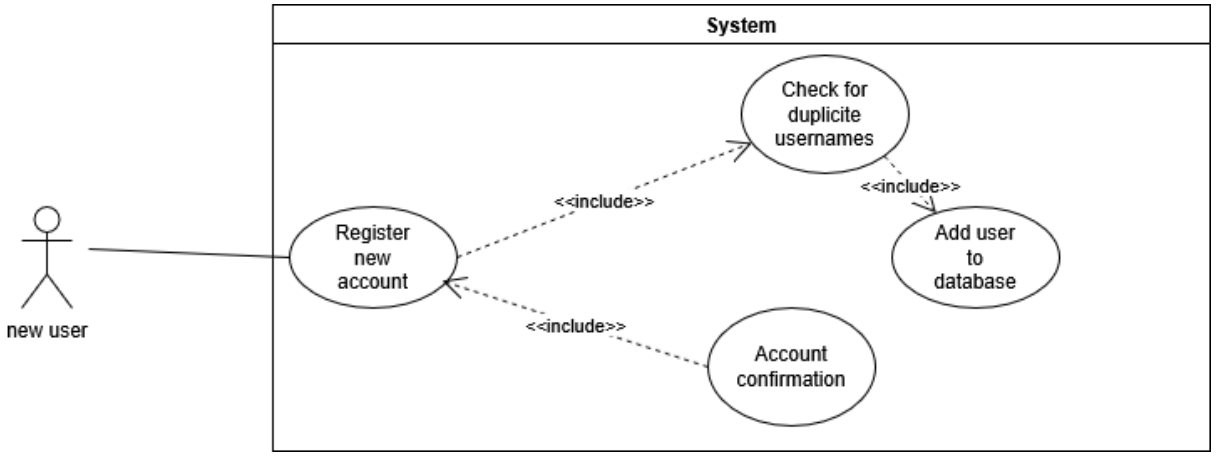
- **User** - any registered user. They want to see their (personal) tasks/projects, edit or delete them and get notifications when deadline is approaching.
- **Project Admin** - user who creates shared project or user who is assigned admin permissions by initiator of project. They can invite other users to the project, assign tasks and also make changes to the project.
- **Project Member** - user invited to shared project. They can see details of the project (tasks, deadlines, current state) but cannot make changes to project details or tasks they're haven't been assigned.

Use Case Diagram

Notes:

- CRUD = Create, Read, Update, Delete
- "Log in" and "Log out" use cases are not user goals, but they are included in the diagram for completeness. All other use cases require the corresponding role (primary actor) login to complete (they have the precondition "[role] is logged in").
- Delete: Some delete functionalities for the admin role may be implemented as a "soft" delete due to incoming references. This will be further refined during implementation.


Capacity planer
Log in
Log out
Manage personal projects/ tasks (CRUD)
View shared project's subtasks
Manage assigned project subtasks (CRU)
Add / Remove members
Manage project subtasks (CRUD)
Edit project details
Grant admin privileges
User
Project Member
Project Admin


System
Check for duplicite usernames
<<include>>
<<include>>
Register new account
Add user to database
<<include>>
Account confirmation
new user

Data model

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

Entities and attributes

Attributes that are self-describing are not mentioned explicitly in this section.

User

- user identified by unique id. User can become Project member or admin.

Account

- An account used for authentication to the application. The username is unique.
- Constraints & rules:
    - Each account belongs to exactly one employee.

Project Member

- An assignment of a user to a project as a member.
- Attributes:
    - admin: True/False, user who created the project is admin by default
    - task: project subtask assigned to the user
    - deadline: for the subtask

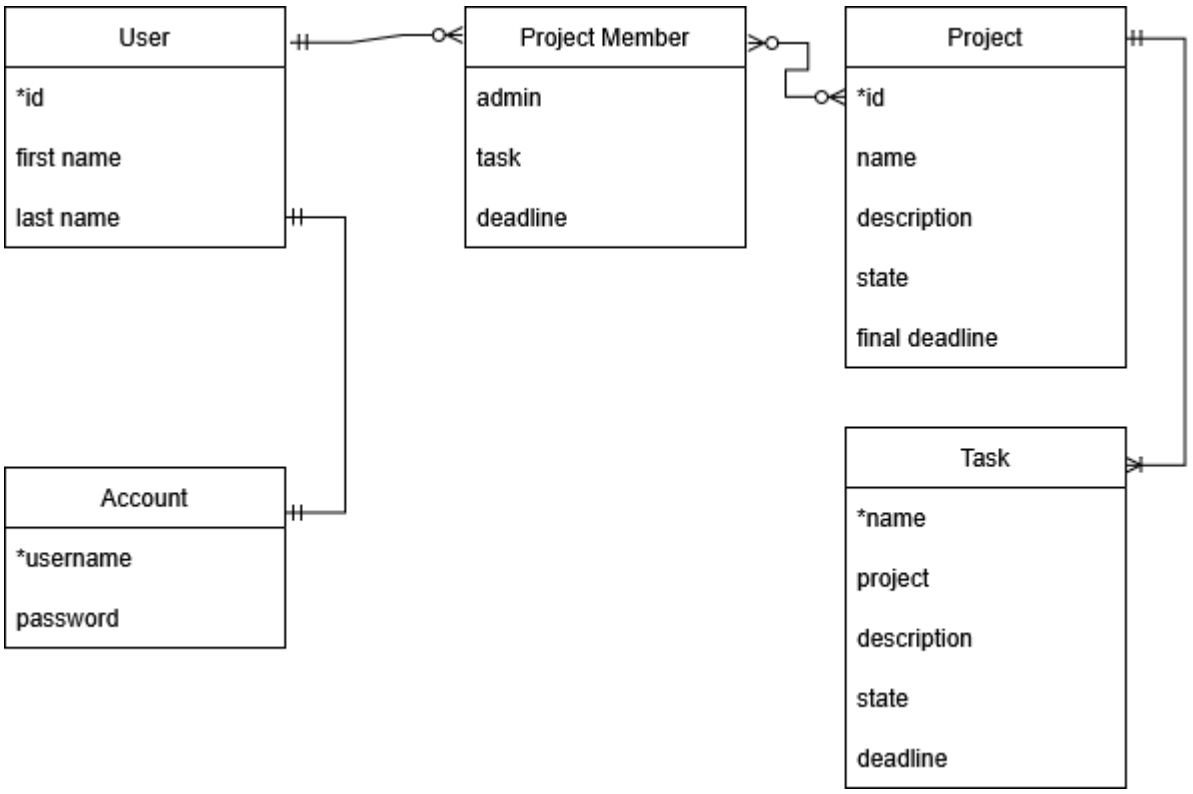
Project

- (shared) project to which other users may be invited.
- Attributes:
    - final_deadline: of the project as a whole
    - state: in progress/ overdue / finished
- Constraints & rules:
    - project must have at least one admin (the user who created it)

Task

- smaller tasks that can be assigned to members
- task past deadline are still visible but cannot be edited
- Attributes:
    - project: id/name of a project task belongs under (for tasks that don't belong to any project value is 0)
    - deadline: of the subtask
    - state: in progress/ overdue / finished
- Constraints & rules:
    - deadline can't be later than projects final deadline
    - tasks past deadline cannot be assigned to members

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
- 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

Future Work

Time schedule

Week 4

- update specification
- set up dev environment

Week 5

- set up SQL database
- basic html
- implement functionality to create projects and tasks
- set up CRUD operations for projects and tasks

Week 6

- user registration, login and logout functionality, password hashing and storage
- set up permissions for tasks and projects
- ensure end-to-end functionality

Week 7

- functionality for users to share projects and tasks with other users + permissions for shared access
- basic notification system (e.g. for task assignments, project changes, upcoming deadlines)

Week 8

- fix bugs, refine features.
- implement any remaining features as mock

Week 9 (Beta version)

Week 10

- implement improvements based on feedback
- implement previously mock features

Week 11 (Final version)

- finalize project