Specification – Homework manager

The goal of this web application is to provide a lightweight tool for homework planning. The primary target users are students who need to plan and track their homework, and teachers who need to publish homeworks for their students. Students can see other teachers' homeworks and add them to their lists of homeworks, if the teacher has published that homework.

Functional requirements

Roles

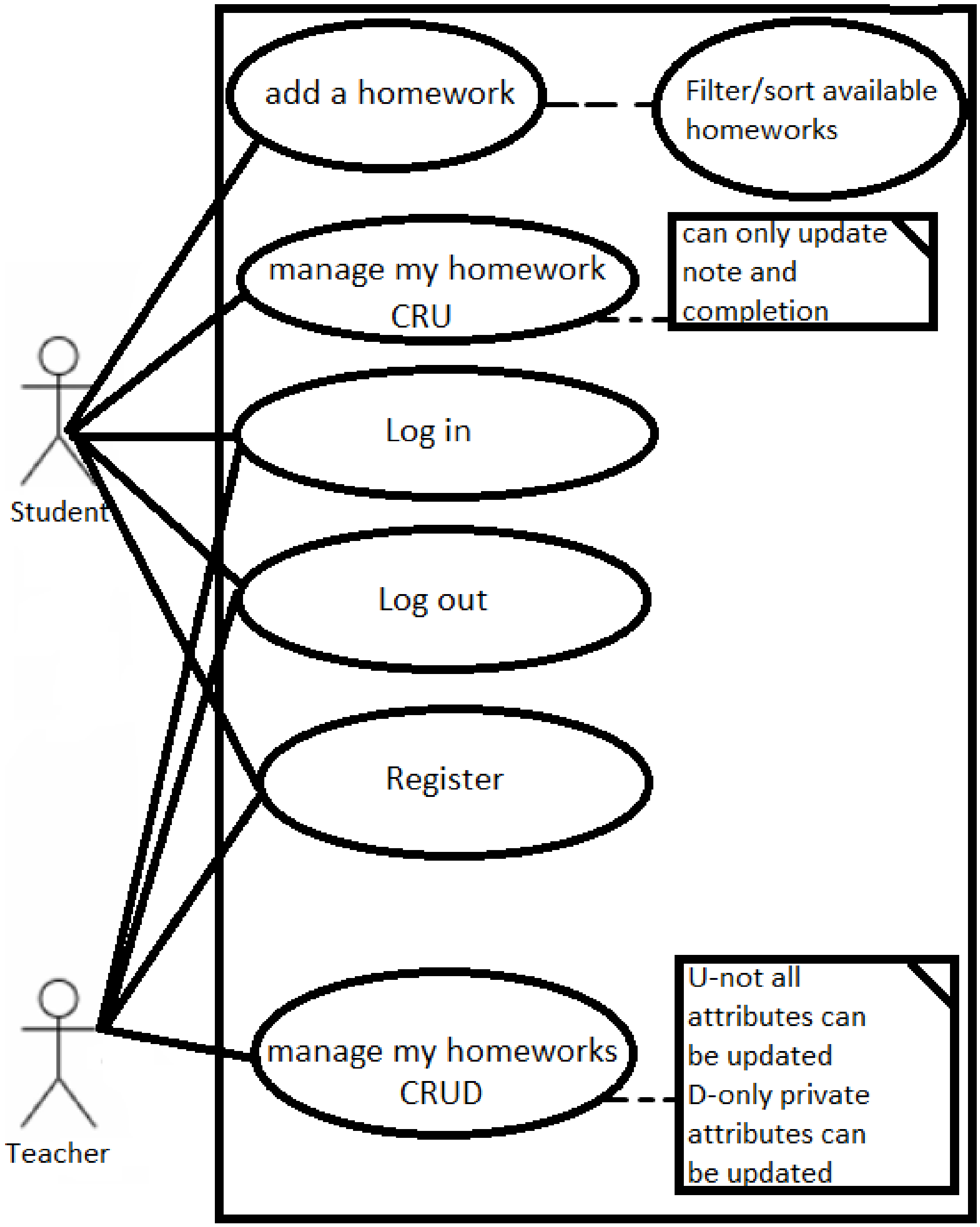
The application distinguishes two roles:

- **Student - t**hey want to see homeworks added by teachers and add them to their private list of homeworks.
- **Teacher –** creates homeworks/drafts of homeworks, that he can publish.

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”).


add a homework
Filter/sort available homeworks
manage my homework CRU
can only update note and completion
Log in
Log out
Register
manage my homeworks CRUD
U-not all attributes can be updated
D-only private attributes can be updated
Student
Teacher

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.

Note: User and Account might get merged into one table on the database level.

User

- A user is identified uniquely by user_id.
- A user can be either a teacher or a student.

Account

- An account used for authentication to the application. The username is unique.
- Attributes:
    - *password:* Stored as a hash value
    - role: determines if the account belongs to a teacher or a student.

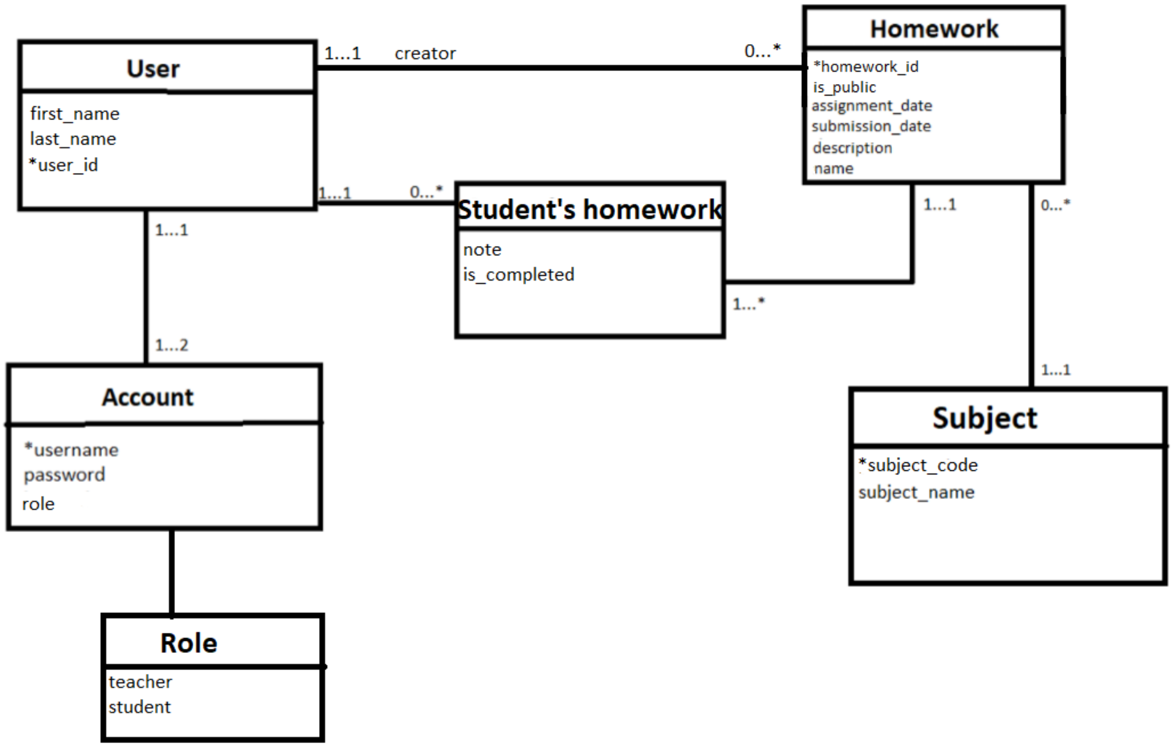
Homework

- Homework identified by its unique name.
- Attributes:
    - is_public: determines if the students can add this homework to their lists or it is just a draft created by a teacher.
    - *assignment_date:* The date when a given homework is added to the list of current homeworks. Assignment date can't be changed if is_public is true.
    - *due_date:* The date when a given homework is due. Due_date can only be postponed if is_public is true.
    - description: Contains a description of what the student is supposed to do in this homework. Can't be changed if is_public is true.
    - name: can't be changed if is_public is true.

Student's homework

- An assignment of a user to a homework as a user who needs to do this homework.
- Attributes:
    - *Note: Contains a note from the student about the homework, e.g. the progress of the homework.*

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

Time schedule

Week 4
● Setup of the developmental environment
Week 5
● Draw gui on paper, set up localhoast, React
Week 6
● Implement React components for adding new homeworks as a teacher, React routing
Week 7
● Set up Node.js + Express.js + database, create tables
Week 8
● Implement api calls that return data of new homeworks being added, authentication
Week 9 (Beta version)
● Deployment to hosting, testing
Week 10
● Backend functionality + connection for the remaining features
Week 11 (Final version)
● Deployment to hosting of those features, testing