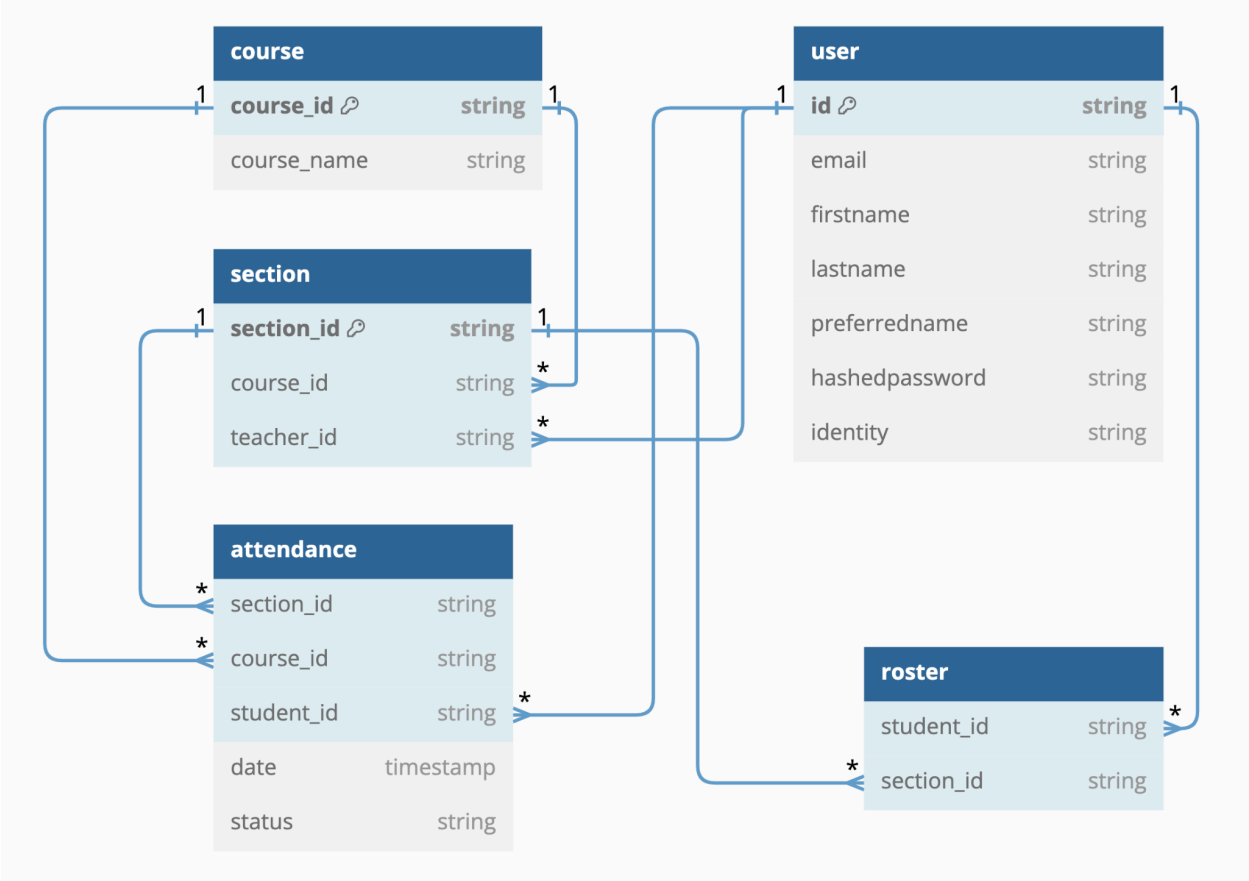
Specification of Requirements

1. System Architecture of the Project
    a. An admin app that manage user and courses
        i. should be a JavaFX app
        ii. functions for users:
            1. have admin account to login and do operations in the admin app
            2. signup function: add user into the system
            3. update user information
            4. delete an user in the system
            5. add permission to change student names
        iii. functions for courses:
            1. add, remove and update classes and sections functions
            2. when removing a course, need a double check
            3. enroll students into classes (one particular section). Enrollment can be manual (one by one) or batch (e.g., reading from a csv file).
            4. drop student from class
    b. Client/Server Architecture for the attendance take web app (for both professor and student)
        i. multithreading for client
        ii. student function
            1. login function
            2. take attendance when being present in the class location
            3. request for an attendance report including course name, all attendance records for this course and attendance points (total points and each attendance point)
            4. change course attendance notification preference on the profile page
        iii. professor function
            1. login function
            2. professors can choose which course(s) to teach.
                - once a professor decide which course to teach, he or she cannot give up teaching it
                - a professor can choose to teach multiple courses
            3. add new attendance for students to initialize a new limited-time attendance taking process for present students
            4. retrieve the attendance history for all students for a previous date or modify the attendance for a student on a particular day in a class.

5. generate attendance report showing the attendance participation points of each student in a class (Tardy counts 80%)

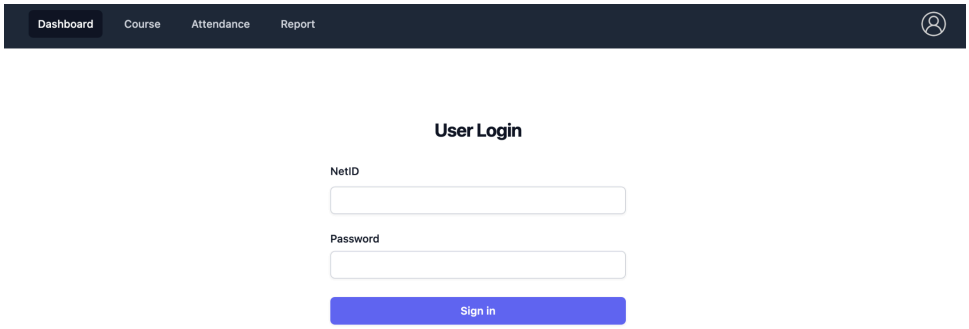
2. Database and DAO

a. JavaFX implementing the DAO layer from evolution 2 to separate the data process logic in a separate layer
b. Server/Client architecture used another DAO combined with entity and services of spring boot to support the data transmission between frontend and backend.
c. database design

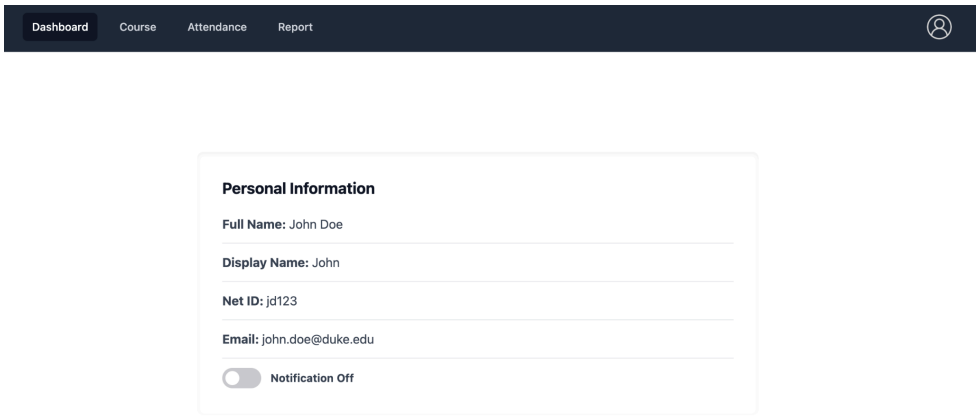


3. GUI for Taking Attendance (Examples with fake data)

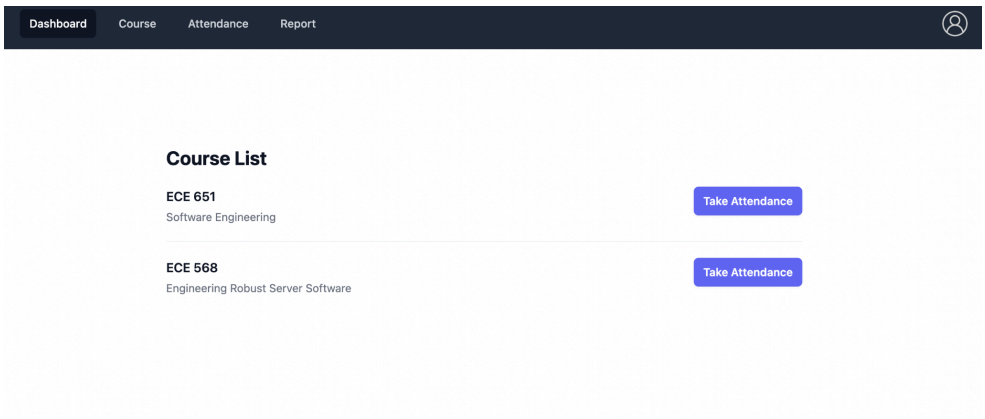
a. login pages for professors and students



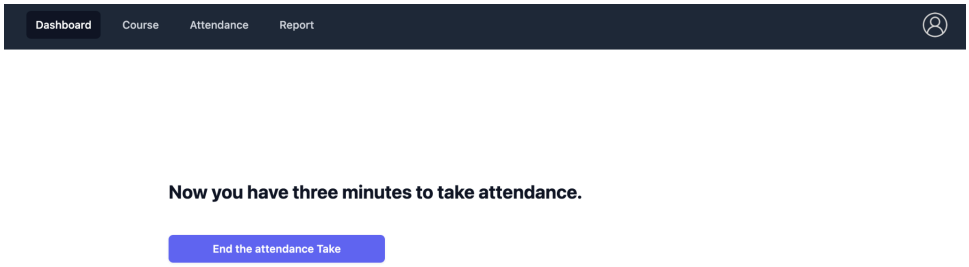
b. profile pages for professors and students



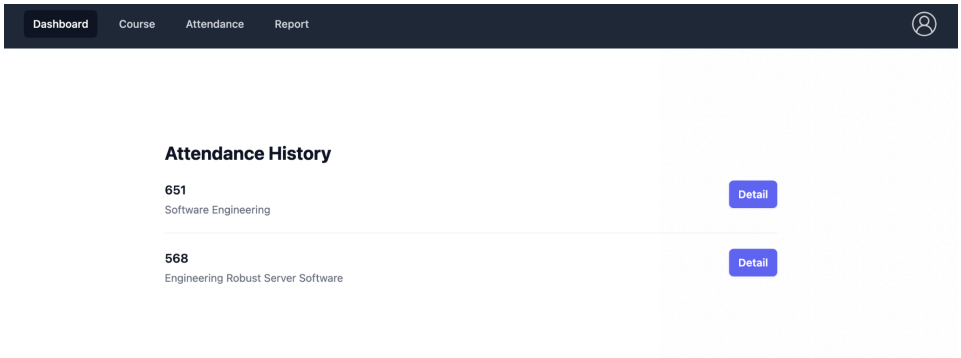
c. course list pages for professors and students

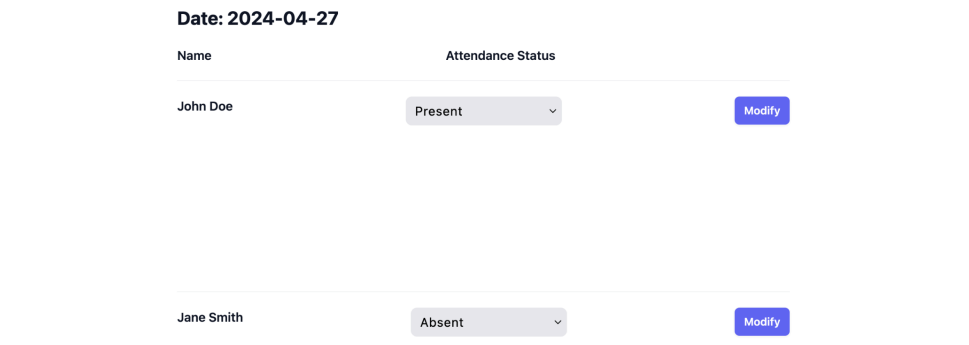


d. attendance taking pages for professors and students



e. attendance history pages for professors and students





4. Extra Credit

a. The attendance taking process takes less than 3 minutes
    i. When the professor click the take attendance button, the attendance taking process will automatically begin and students and hit the take attendance button on their web page to take attendance during a three-minute time window
    ii. After the time window ends, the professor can hit end attendance taking button to end the attendance taking process
b. During the attendance taking process, the professor do not need to do anything
    i. The attendance taking process is stated in the above bullet points so the professor only needs to hit two buttons to begin and end the attendance taking session. During the attendance taking process, the professor does not need to to anything
c. Prevent cheating when taking attendance
    i. We used the location matching to avoid cheating. When the professor hits the attendance taking button to start the attendance taking process, we will get the professor's location, which should be the classroom location. When each student hits taking attendance button, we also get the student’s location and compare it to the location of professor and set a reasonable radius between them to make sure only students who is actually present in the classroom can take attendance successfully