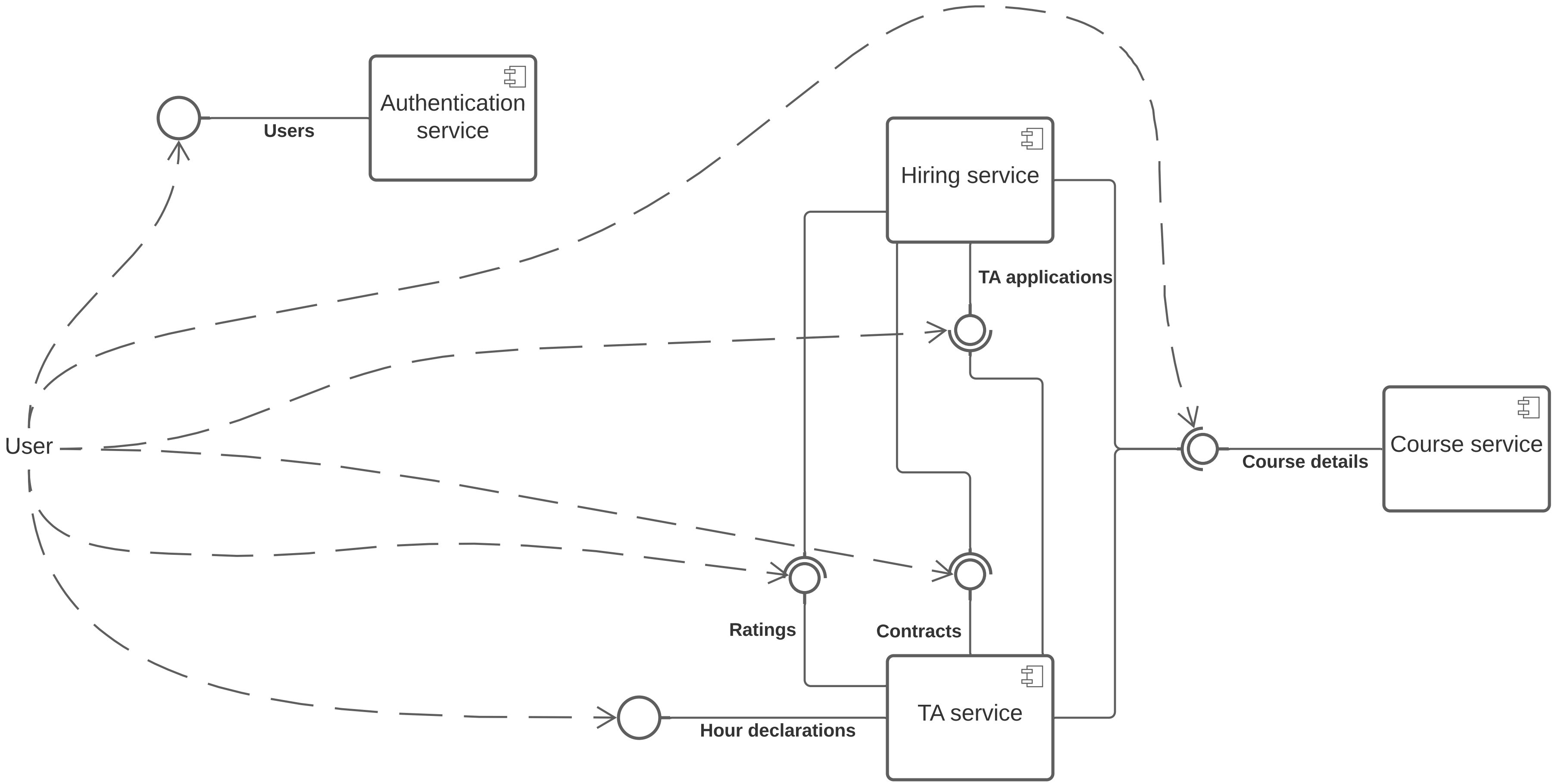


Course service:

Stores all information of a course (dates, lecturer, etc.).

Users can create courses, and other services request course information (e.g. the netid of the responsible lecturer in order to check permissions, etc.).

Hiring service:

Handles applications (incl. grades).

Users can apply to be TAs, and lecturers can manage applications.

Requests course details to check if the user can apply at that time. Requests the netid of the lecturer for authorization before allowing users to modify applications (e.g. approve, reject, etc.).

Requests ratings and past contracts when showing applications to lecturers.

TA service:

Handles active and past contracts, hour declarations, and ratings.

TAs can submit hour declarations, and lecturers can approve them.

Requests the netid of the lecturer for authorization before allowing users to approve hours or create contracts.

Automatically approves a TA application when a contract is created.

Authentication service:

Handles authentication.

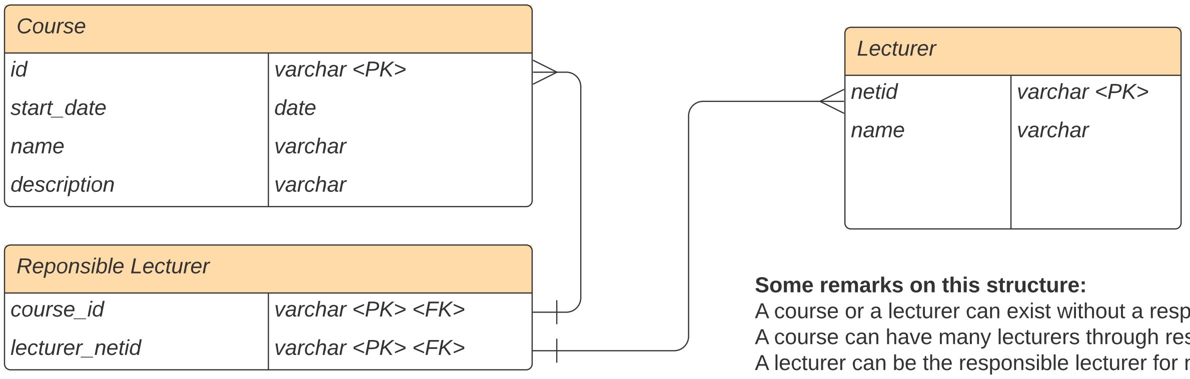
Allows users to register.

Allows registered users to login, and provides them with a JWT token that other services can then verify independently.

ER-diagram database (Courses, Hiring and TA)

SEM Group 13B | November 26, 2021

Courses Microservice



Course	
id	*varchar <PK>*
start_date	*date*
name	*varchar*
description	*varchar*

Reponsible Lecturer	
course_id	*varchar <PK> <FK>*
lecturer_netid	*varchar <PK> <FK>*

Lecturer	
netid	*varchar <PK>*
name	*varchar*

Some remarks on this structure:
A course or a lecturer can exist without a responsible lecturer assigned.
A course can have many lecturers through responsible lecturers
A lecturer can be the responsible lecturer for many courses.

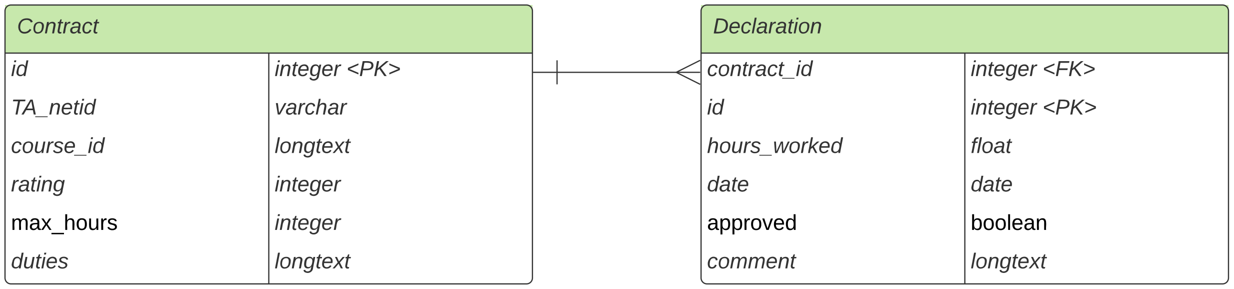
Note that we can now easily add more columns to the lecturer in the future if ever needed and have less duplication if a lecturer is responsible for multiple courses (occurs often in reality).

Hiring Microservice

Application	
course_id	*varchar <PK>*
netid	*varchar <PK>*
motivation	*longtext*
grade	*float*
status	*varchar*

Remarks
The hiring microservice will handle everything that has to do with the hiring proccess. It will only need to store applications of students with their motivation and grade. This all can be stored in one simple table.

TA Microservice



Contract	
id	*integer <PK>*
TA_netid	*varchar*
course_id	*longtext*
rating	*integer*
max_hours	*integer*
duties	*longtext*

Declaration	
contract_id	*integer <FK>*
id	*integer <PK>*
hours_worked	*float*
date	*date*
approved	boolean
comment	*longtext*