**Database schema:**

**User schema:**

-UserID: String,

-FullName: String,

-BirthDay: Date,

-Email: String,

-City: String,

-Country: String,

-Description: String,

-Role: String,

-Password: String,

-Courses: [String – ID of course objects in db]

**Course schema:**

-CourseCode: String,

-CourseName: String,

-Year: Int,

-Semester: Int,

-Participants: [String – ID of user objects in db]

**Grade schema:**

-UserID: String – ID of user objects in db,

-StudentName: String,

-StudentID: String,

-CourseID: String – ID of course objects in db,

-Assignment: Float; default: 0,

-Midterm: Float; default: 0,

-Final: Float; default: 0

**Available API:**

**-students:**

+query all users as student in db

**-user(id):**

+query one specific user.

+Passing one required arguments: user’s id (id of user objects in db)

**-teachers:**

+query all user as teacher in db

**-viewParticipants(courseid):**

+query all users that enroll in a course.

+Pass one required argument as courseid (id of course objects in db)

**-courses:**

+query all courses in db

**-login(email, password):**

+authenticate by jsonwebtoken.

+Pass 2 required arguments: email & password

+return a token (String)

**-createUser(email, password, role):**

+create new user account

+passing 3 required arguments: email, password, role

+return a new user (User)

**-updateUser(id, userid, fullname, birthday, email, city, country, description):**

+update user’s profile

+passing optional arguments listed above

+return an updated user (User)

**-removeUser(id)**

+remove one user

+passing one argument as user’s id (id of user objects in db)

**-enrollCourse(courseid)**

+Allow user to enroll in one course

+passing one argument as course’s id (id of course objects in db)

**-assignUserToCourse(courseid, userid)**

+Assign user to be enrolled in one course

+passing 2 required arguments: userid & courseid (id in db)

**-modifyGrade(userid, courseid, assignment, midterm, final)**

+modify grades of one student

+passing 2 required arguments: userid & courseid (id in db)

+passing optional arguments (3 kinds of grade: assignment, midterm, final)

+return new grade of user (Grade)

**-createCourse(code, name, year, semester)**

+create new course

+passing 4 required arguments: code as course’s code, name as course’s name, year as open year of course, semester as semester to which course belong.

+return new course (Course)

**Authorization**

**-students:** Admin

**-user:** Admin

**-teachers:** Admin

**-viewParticipants:** Admin | Teacher | Student

**-course:** Admin | Teacher | Student

**-createUser:** Admin

**-updateUser:** Admin | Teacher | Student

**-enrollCourse:** Student

**-assignUserToCourse:** Admin

**-createCourse:** Admin

**-modifyGrade:** Admin | Teacher