# Roles Shortcut:

A - Admin

T - Teacher

S - Student

# User:

Login for user (teacher, student).

# Admin:

Register/Login for admin.

Register for student and teacher.

# Student:

* Update student info (phone number, level: upgrade only). >> A
* Get student info: by student id. >> A, T, S
* Get all students. >> A
* Get students by course id (status! = dropped). >> T
* Enroll student in a course: (consider 1. students previous schedule, 2. student level (must be the same level), 3. course capacity). >> S
* Drop student from a course. (change status in student course to == drop)>> A

# Teacher:

* Update teacher info (phone number, course load). >> A
* Get teacher info: by teacher id. >> A
* Get all teachers. >> A

# Subject:

* Add subject. >> A
* Update subject info (name, department Id). >> A
* Get subject info: by subject id. >> A
* Get all subjects. >> A

# Course:

* Add course: without teacher Id. >> A
* Update course info (teacher id, capacity), (in selecting teacher id: consider 1. teacher previous schedule, 2. teacher course load, 3. Department). >> A
* Edit: Get course info: by course id. >> A, S, T
* Get all courses. >> A.
* Get courses that have not started yet (start date is not passed yet). >> S
* New: Delete course: if start date pass, course cannot be deleted. >> A
* Edit: Get previous courses (end date is passed) with their status and AverageGrades for a student: by student id. >> S
* Get current courses (end date is not passed) for a student: by student id. (With the course days and time) >> S
* New: Get all courses for a student: by student id. >> A
* New: Get all courses for a teacher: by teacher id. >> A
* Get current courses (end date is not passed) for a teacher: by teacher id. (With the course days and time) >> T
* Calculate the average grades for the students in the course (for each student in a course update the AverageGrdaes field (in StudentCourses table) to be equal (sum of grades/number of assignments). Then by AvergaeGrades field determine the status filed (int StudentCourses table) to be pass/ fail. By course id. >> T

# Assignments:

* Add Assignment. >> T
* Get all course assignments: By course id. >> T
* Get assignment info: by assignment id. >> T, S
* Update assignment (name, description, due date, visibility). >> T
* New: Delete assignment >> T
* New: Get the student course assignments (by course id, get all assignments in the course (then for each assignment get the StudentAssignment record (from StudentAssignments table by assignment id and student id). >> S

Response: (for StudentId =1)

[

{

“AssignmentId”: 1,

“AssignmentName”:”Tunfiy Platform”,

"DueDate": "2024-09-01",

“StudentAssignments”:

{

“StudentAssignmentId” = 1,

“Grade”=10,

“Feedback”=”nice work”

}

},

{

“AssignmentId”: 2,

“AssignmentName”:”Smart Cart”,

"DueDate": "2024-08-25",

“StudentAssignments”:

{

“StudentAssignmentId” = 3,

“Grade”=10,

"Feedback"=”nice work”

}

}

]

* New, Get the students assignment record for an assignment (get all students by course id, then for each student get the StudentAssignment record (by assignment id and student id)). >> T

Response: (for AssignmentId =1)

[

{

"StudentId": 1,

"FullName": "Dima Mohamad Salem",

"StudentAssignments":

{

"StudentAssignmentId": 3,

"SubmissionDate": "2024-09-01T17:00:00",

"Grade": 0,

"Feedback": "",

}

},

{

"StudentId": 2,

"FullName": "Abdullah Mohamad Salem",

"StudentAssignments": null

}

]

# StudentAssignments:

* Add studentAssignment record. (case one: student will add the submission, case two: teacher will create the record if there is no submission and no record for the student, he will add the grade and the feedback)>> S, T
* Update studentAssignment record (grade, feedback). >> T
* Get studentAssignment record: by studentAssignment id. >> S, T
* Get studentAssignment record by assignment id and student id. >> S, T

# Department:

* Get all departments info. >> A
* Add department. >> A
* Update department info. (Name). >> A
* Get department by department id. >> A

# Lectures:

* Add Lectures method (call it in create course method) where it add all course lectures dates in the lecture table. Method in service not route.
* Delete course attendance records (Lecture attendance, Lectures (can be on delete cascade)) after course end date. (By course id check on end date if passed). >> by default, check where we can call it? Method in service not route.

# LectureAttendance:

* Add Lecture Attendance records.>> T

Request:

{

“LectureId”:1,

“StudentAttendance”: [

{

“StudentId”:”1,

“Status”: ”present”

},

{

“StudentId”:”2,

“Status”: ”absence”

}

]

}

* Get absence dates by course id and student id. >> S.
* Get the course students absence rate: by course id (get the students in the course and then calculate the rate for each student= (num of absence lectures/ num of total lectures)\*100%. >> T --- one route should load the page.

# Report: