Technology Stack

- Spring Boot
- H2 Database (in-memory database)
- JPA
- JUnit
- Postman

How to run

If using Eclipse IDE, go to File->Import->Maven->Existing Maven Projects. Select root directory to the folder. This project is developed using Java 11, so Java SE 11 is required to run the application. To build the application, go to Package Explorer -> Right click on project name -> select “Run As” -> Maven Build. This would also run all the JUnit tests. To run the application, right click on file TaskApplication.java (located inside src/main/java) -> select “Run As” -> Java Application. Then you can use Postman to perform CRUD operations. The raw data will be populated using data.sql file under the resources package

EER Diagram

I created an EER diagram to show a basic visualization of my database, to help with the API CRUD operations. You can also go to h2-console to look at the schema.

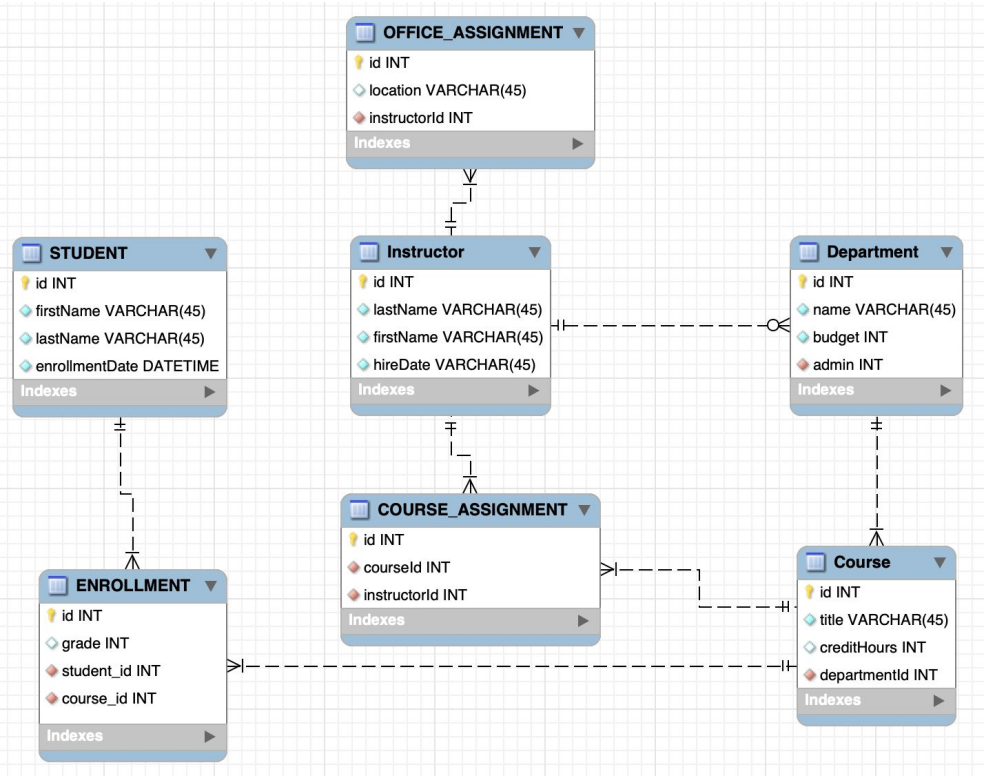

Figure 1

CRUD Operations

The following are the operations that you are able to perform. Request body is of type JSON.

Student

- GET /api/student
    - Get all students
- GET /api/student/{id}
    - Get student by id
- GET /api/student/{id}/enrollments
    - Get all enrollments for a student
- POST /api/student
    - Create a new student
    - Request body consists of firstName and lastName
    - Checks for required fields firstName and lastName
    - Request body does not need enrollmentDate, it is set to the time of creation
- PUT /api/student/{id}
    - Request body consists of firstName (optional) and lastName (optional)
    - Update firstName and lastName of a given student
- DELETE /api/student/{id}
    - Delete a student

Enrollment

- GET /api/enrollments
    - Get all enrollments
- GET /api/enrollments/{id}
    - Get enrollment by id
- DELETE /api/enrollments/{id}
    - Delete an enrollment
- POST /api/management/enroll
    - Enroll a student
    - Request body consists of student_id, course_id, and grade (optional)
    - Checks for valid student_id and course_id
    - Does not allow student to be registered in the same course more than once
- PUT /api/management/update-enroll/{id}
    - Updates an enrollment
    - Request body consists of grade
    - Able to update the grade field

Course

- GET /api/course

    - Get all courses
- GET /api/course/{id}
    - Get course by course id
- GET /api/course/{id}/enrollments
    - Get all enrollments for a particular course
- GET /api/course/{id}/course-assignments
    - Get all course assignments for a given course
- POST /api/course
    - Create a new course
    - Request body consists of title, creditHours (optional), and departmentId (optional)
    - Checks for required field title
    - Checks for valid departmentId if provided
- PUT /api/course/{id}
    - Update a course
    - Request body consists of title (optional), creditHours (optional), and departmentId (optional)
    - Can update title, creditHours, and departmentId
    - Checks for valid departmentId if provided
- DELETE /api/course{id}
    - Delete a course

Department

- GET /api/department
    - Get all departments
- GET /api/department/{id}
    - Get department by id
- GET /api/department/{id}/courses
    - Get all courses for a department
- POST /api/department
    - Create a new department
    - Request body consists of name, budget, and admin (optional)
    - Checks for required fields name and budget
    - Can also specify admin/head of department. This field is optional. Department can only have one assigned/head instructor, or none.
- PUT /api/department/{id}
    - Update a department
    - Request body consists of name (optional), admin (optional), and budget (optional)
    - Can change name, admin, and budget
- DELETE /api/department/{id}
    - Delete a department

Instructor

- GET /api/instructor
    - Get all instructors
- GET /api/instructor/{id}
    - Get instructor by id
- GET /api/instructor/{id}/course-assignments
    - Get all course assignments for an instructor
- GET /api/instructor/{id}/office-assignments
    - Get all office assignments for an instructor
- GET /api/instructor/{id}/departments
    - If an instructor is a department admin/head, then this would return the department details
- POST /api/instructor/
    - Create a new instructor
    - Request body consists of firstName and lastName
    - Checks for required fields firstName and lastName
    - Request body does not need hireDate, it is set to the time of creation
- PUT /api/instructor/{id}
    - Update an instructor
    - Request body consists of firstName (optional), and lastName (optional)
    - Able to change firstName and lastName
- DELETE /api/instructor/{id}
    - Delete an instructor

Course Assignment

- GET /api/course-assignments
    - Get all course assignments
- GET /api/course-assignments/{id}
    - Get course assignment by id
- DELETE /api/course-assignments/{id}
    - Delete a course assignment
- POST /api/management/assign-course/{instructorId}/{courseId}
    - Creates a new course assignment
    - This POST request does not require a body, the request url specifies the instructorId and courseId
    - Checks for valid instructorId and courseId
    - Instructors are able to teach multiple courses in the same department
    - The Head/Admin of a department cannot teach a course outside of their department. If the instructor is not the head of a department then they may teach a course in multiple departments
- PUT /api/management/updated-assigned-course/{id}/{instructorId}/{courseId}

    - Updates a course assignment
    - This PUT request does not require a body, the request url specifies the instructorId and courseId
    - Can update instructorId and courseId
    - Checks for valid instructorId and courseId
    - Instructors are able to teach multiple courses in the same department
    - The Head/Admin of a department cannot teach a course outside of their department. If the instructor is not the head of a department then they may teach a course in multiple departments

Office Assignment

- GET /api/office-assignments
    - Get all office assignments
- GET /api/office-assignments/{id}
    - Get office assignment by id
- DELETE /api/office-assignments/{id}
    - Delete an office assignment
- POST /api/management/assign/{instructorId}/{location}
    - Creates a new Office Assignment
    - This POST request does not need a request body - url specifies instructorId and location
- PUT /api/management/update/office-assignment/{id}
    - Update an office assignment
    - Request body consists of instructorId (optional) and location (optional)
    - Can update instructorId and location

Testing

- Wrote unit tests using JUnit
- Some of the test cases I've included:
    - Insert a new course
    - Find course by id
    - Remove a course
    - Create a new department
    - Check if you can find a department by its admin/head instructor
    - Check if a course is assigned with a given department
    - Check if a student has already registered in a given course