Springboot University System Backend

- It is a full backend project build on the university database at the previous week.
- It is build on the MVC model(the view is not implemented yet)
- You can open this project using Intellij IDE and you can find the source files at src/main/java/com.example.demo

APIs

- you can import the APIs to postman using that file Open Postman JSON File

Student Management APIs

Description: A RESTful API for managing student records, supporting CRUD operations and filtering by faculty or academic level.
Base URL: `http://localhost:8080/students`

---

1. Get All Students

- **Endpoint**: `GET /students/list`
- **Description**: Retrieves a list of all students in the system.
- **Parameters**: None
- **Response**:
  - `200 OK`: A JSON array of student objects.

---

2. Get Students by Faculty

- **Endpoint**: `GET /students/list/faculty/{faculty_name}`
- **Description**: Retrieves students belonging to a specific faculty.
- **Path Parameter**:
  - `faculty_name` (string): The name of the faculty.
- **Response**:
  - `200 OK`: A JSON array of student objects matching the faculty.

---

3. Get Students by Level

- **Endpoint**: `GET /students/list/level/{level}`
- **Description**: Retrieves students by their academic level.
- **Path Parameter**:
  - `level` (integer): The academic level.
- **Response**:
  - `200 OK`: A JSON array of student objects matching the level.

---

4. Count Students

- **Endpoint**: `GET /students/count`
- **Description**: Returns the total number of students in the system.
- **Parameters**: None
- **Response**:
  - `200 OK`: An integer representing the count of students.

---

5. Add New Student

- **Endpoint**: `POST /students/add`
- **Description**: Creates a new student record.
- **Request Body** (JSON):

```
{
  "studentName": "string",
  "studentLevel": integer,
  "facultyName": "string"
}
```

- **Response**:
  - `200 OK`: The created student object.
  - `400 Bad Request`: If the request body is invalid.

---

6. Find Student by Name

- **Endpoint**: `GET /students/find/name/{name}`
- **Description**: Retrieves a list of students with a matching name.
- **Path Parameter**:
  - `name` (string): The name of the student.
- **Response**:
  - `200 OK`: A JSON array of student objects with matching names.
  - `404 Not Found`: If no students match the name.

---

7. Find Student by ID

- **Endpoint**: `GET /students/find/id/{id}`
- **Description**: Retrieves a student by their unique ID.
- **Path Parameter**:
  - `id` (integer): The ID of the student.
- **Response**:
  - `200 OK`: The student object with the matching ID.
  - `404 Not Found`: If no student matches the ID.

---

8. Delete Student by ID

- **Endpoint**: `DELETE /students/delete/id/{id}`
- **Description**: Deletes a student by their unique ID.
- **Path Parameter**:
  - `id` (integer): The ID of the student to delete.
- **Response**:
  - `200 OK`: The deleted student object.
  - `404 Not Found`: If no student matches the ID.

---

9. Delete Student by Name

- **Endpoint**: `DELETE /students/delete/name/{name}`
- **Description**: Deletes a student by their name.
- **Path Parameter**:
  - `name` (string): The name of the student to delete.
- **Response**:
  - `200 OK`: The deleted student object.
  - `404 Not Found`: If no student matches the name.

---

10. Update Student Information

- **Endpoint**: `PUT /students/update/{id}`
- **Description**: Updates a student's information by their ID.
- **Path Parameter**:
  - `id` (integer): The ID of the student to update.
- **Request Body** (JSON):

  ```json
  {
    "studentName": "string",
    "studentLevel": integer,
    "facultyName": "string"
  }
  ```

- **Response**:
  - `200 OK`: The updated student object.
  - `404 Not Found`: If no student matches the ID.
  - `400 Bad Request`: If the request body is invalid.

Faculty Management API

Description: A RESTful API for managing faculty records, supporting CRUD operations, searching by name, and retrieving associated courses and students.

Base URL: `http://localhost:8080/faculties`

---

1. Get All Faculties

- **Endpoint**: `GET /faculties/list`
- **Description**: Retrieves a list of all faculty members in the system.
- **Parameters**: None
- **Response**:
  - `200 OK`: A JSON array of faculty objects.

---

2. Get Faculty by ID

- **Endpoint**: `GET /faculties/id/{id}`
- **Description**: Retrieves a specific faculty member by their ID.
- **Path Parameter**:
  - `id` (integer): The ID of the faculty.
- **Response**:
  - `200 OK`: The faculty object with associated students and courses.
  - `404 Not Found`: If no faculty matches the ID.

---

3. Get Faculty by Name

- **Endpoint**: `GET /faculties/name/{name}`
- **Description**: Searches for faculties by name (partial match).
- **Path Parameter**:
  - `name` (string): The faculty name or part of the name.
- **Response**:
  - `200 OK`: A JSON array of matching faculty objects.
  - `404 Not Found`: If no faculties match the name.

---

4. Count Faculties

- **Endpoint**: `GET /faculties/count`
- **Description**: Returns the total number of faculties in the system.
- **Parameters**: None
- **Response**:
  - `200 OK`: An integer representing the count of faculties.

---

5. Add New Faculty

- **Endpoint**: `POST /faculties/add`
- **Description**: Creates a new faculty record.
- **Request Body** (JSON):

```
{
  "facultyName": "string",
  "specialization": "string"
}
```

- **Response**:
  - `200 OK`: The created faculty object.
  - `400 Bad Request`: If the request body is invalid.

---

6. Update Faculty Information

- **Endpoint**: `PUT /faculties/update/{id}`
- **Description**: Updates a faculty's details by their ID.
- **Path Parameter**:
  - `id` (integer): The ID of the faculty to update.
- **Request Body** (JSON):

```
{
  "facultyName": "string",
  "specialization": "string"
}
```

- **Response**:
  - `200 OK`: The updated faculty object.
  - `404 Not Found`: If no faculty matches the ID.
  - `400 Bad Request`: If the request body is invalid.

---

7. Delete Faculty by ID

- **Endpoint**: `DELETE /faculties/delete/id/{id}`
- **Description**: Deletes a faculty by their unique ID.
- **Path Parameter**:
  - `id` (integer): The ID of the faculty to delete.
- **Response**:
  - `200 OK`: The deleted faculty object.
  - `404 Not Found`: If no faculty matches the ID.

---

8. Delete Faculty by Name

- **Endpoint**: `DELETE /faculties/delete/name/{name}`
- **Description**: Deletes a faculty by their exact name.
- **Path Parameter**:

    - `name` (string): The exact name of the faculty to delete.
- **Response**:
    - `200 OK`: The deleted faculty object.
    - `404 Not Found`: If no faculty matches the name.

---

9. Get Faculty Courses

- **Endpoint**: `GET /faculties/{id}/courses`
- **Description**: Retrieves all courses offered by a specific faculty.
- **Path Parameter**:
    - `id` (integer): The ID of the faculty.
- **Response**:
    - `200 OK`: A JSON array of course objects.
    - `404 Not Found`: If no faculty matches the ID.

---

10. Get Faculty Students

- **Endpoint**: `GET /faculties/{id}/students`
- **Description**: Retrieves all students belonging to a specific faculty.
- **Path Parameter**:
    - `id` (integer): The ID of the faculty.
- **Response**:
    - `200 OK`: A JSON array of student objects.
    - `404 Not Found`: If no faculty matches the ID.

Course Management API

Description: A RESTful API for managing course records, supporting CRUD operations, searching by name, and filtering by faculty or minimum level.
Base URL: `http://localhost:8080/courses`

---

1. Get All Courses

- **Endpoint**: `GET /courses/list`
- **Description**: Retrieves a list of all courses in the system.
- **Parameters**: None
- **Response**:
    - `200 OK`: A JSON array of course objects.

---

2. Find Course by ID

- **Endpoint**: `GET /courses/id/{id}`
- **Description**: Retrieves a specific course by its ID.

- **Path Parameter**:
  - `id` (integer): The ID of the course.
- **Response**:
  - `200 OK`: The course object.
  - `404 Not Found`: If no course matches the ID.

---

3. Find Courses by Name

- **Endpoint**: `GET /courses/name/{name}`
- **Description**: Searches for courses by name (partial match).
- **Path Parameter**:
  - `name` (string): The course name or part of the name.
- **Response**:
  - `200 OK`: A JSON array of matching course objects.
  - `404 Not Found`: If no courses match the name.

---

4. Count Courses

- **Endpoint**: `GET /courses/count`
- **Description**: Returns the total number of courses in the system.
- **Parameters**: None
- **Response**:
  - `200 OK`: An integer representing the count of courses.

---

5. Add New Course

- **Endpoint**: `POST /courses/add`
- **Description**: Creates a new course record.
- **Request Body** (JSON):

```
{
  "courseName": "string",
  "minLevel": integer,
  "facultyId": integer
}
```

- **Response**:
  - `200 OK`: The created course object.
  - `400 Bad Request`: If the request body is invalid.

---

6. Update Course Information

- **Endpoint**: `PUT /courses/update/{id}`

- **Description**: Updates a course's details by its ID.
- **Path Parameter**:
  - `id` (integer): The ID of the course to update.
- **Request Body** (JSON):

```
{
  "courseName": "string",
  "minLevel": integer
}
```

- **Response**:
  - `200 OK`: The updated course object.
  - `404 Not Found`: If no course matches the ID.
  - `400 Bad Request`: If the request body is invalid.

---

7. Delete Course by ID

- **Endpoint**: `DELETE /courses/delete/id/{id}`
- **Description**: Deletes a course by its unique ID.
- **Path Parameter**:
  - `id` (integer): The ID of the course to delete.
- **Response**:
  - `200 OK`: The deleted course object.
  - `404 Not Found`: If no course matches the ID.

---

8. Get Courses in Faculty

- **Endpoint**: `GET /courses/faculty/{facultyId}`
- **Description**: Retrieves all courses associated with a specific faculty.
- **Path Parameter**:
  - `facultyId` (integer): The ID of the faculty.
- **Response**:
  - `200 OK`: A JSON array of course objects.
  - `404 Not Found`: If no faculty matches the ID.

---

9. Get Courses with Minimum Level

- **Endpoint**: `GET /courses/level/{minLevel}`
- **Description**: Retrieves all courses with a specific minimum level requirement.
- **Path Parameter**:
  - `minLevel` (integer): The minimum level requirement.
- **Response**:
  - `200 OK`: A JSON array of course objects.

    - `404 Not Found`: If no courses match the minimum level.

---

10. Delete Course by Name

- **Endpoint**: `DELETE /courses/delete/name/{name}`
- **Description**: Deletes a course by its name.
- **Path Parameter**:
    - `name` (string): The name of the course to delete.
- **Response**:
    - `200 OK`: The deleted course object.
    - `404 Not Found`: If no course matches the name.

Enrollment Management API

Description: A RESTful API for managing student enrollments in courses, supporting registration, grade updates, unenrollment, and querying enrollment details.
Base URL: `http://localhost:8080/enrollments`

---

1. Enroll Student in Course

- **Endpoint**: `POST /enrollments/enroll`
- **Description**: Registers a student for a course.
- **Request Body** (JSON):

```
{
  "courseId": integer,
  "studentId": integer
}
```

- **Response**:
    - `200 OK`: The enrollment record with course and student details.
    - `400 Bad Request`: If the student's level is too low for the course.
    - `409 Conflict`: If the student is already enrolled in the course.

---

2. Update Student Grade

- **Endpoint**: `PUT /enrollments/grade`
- **Description**: Records or updates a student's grade for a course.
- **Request Body** (JSON):

```
{
  "courseId": integer,
  "studentId": integer,
  "grade": double
}
```

- **Response**:
  - `200 OK`: The updated enrollment record.
  - `404 Not Found`: If the enrollment does not exist.
  - `400 Bad Request`: If the request body is invalid.

---

3. Unenroll Student

- **Endpoint**: `DELETE /enrollments/unenroll/course/{courseId}/student/{studentId}`
- **Description**: Removes a student from a course.
- **Path Parameters**:
  - `courseId` (integer): The ID of the course.
  - `studentId` (integer): The ID of the student.
- **Response**:
  - `204 No Content`: If the unenrollment is successful.
  - `404 Not Found`: If the enrollment does not exist.

---

4. Get Student Enrollments

- **Endpoint**: `GET /enrollments/student/{studentId}`
- **Description**: Lists all courses a student is enrolled in.
- **Path Parameter**:
  - `studentId` (integer): The ID of the student.
- **Response**:
  - `200 OK`: A JSON array of course objects with grades.
  - `404 Not Found`: If no enrollments are found for the student.

---

5. Get Course Roster

- **Endpoint**: `GET /enrollments/course/{courseId}`
- **Description**: Lists all students enrolled in a specific course.
- **Path Parameter**:
  - `courseId` (integer): The ID of the course.
- **Response**:
  - `200 OK`: A JSON array of student objects with grades.
  - `404 Not Found`: If no students are enrolled in the course.

---

6. Get Specific Grade

- **Endpoint**: `GET /enrollments/grade/course/{courseId}/student/{studentId}`

- **Description**: Retrieves a student's grade for a specific course.
- **Path Parameters**:
    - `courseId` (integer): The ID of the course.
    - `studentId` (integer): The ID of the student.
- **Response**:
    - `200 OK`: The grade for the enrollment.
    - `404 Not Found`: If the enrollment does not exist.

---

7. Get All Enrollments

- **Endpoint**: `GET /enrollments/list`
- **Description**: Provides an administrative view of all enrollment records.
- **Parameters**: None
- **Response**:
    - `200 OK`: A JSON array of complete enrollment records.
