Lungu Ștefania-Paraschiva

Simplified system that models a university student enrollment system

Requirements:

- Students with the following information:
  - ID - an integer that uniquely identifies a student
  - Name - Students full name (first name followed by last name)
  - Mobile phone number
- Courses with the following information:
  - CourseCode - a string that uniquely identifies a course
  - Name - name of the course
- Enrollment that indicates which courses a student is enrolled in
  - StudentID
  - CourseCode
  - Date Enrolled
  - Final Grade
- Due to memory and disk constraints, this data is stored in two separate databases partitioned such that: *All data pertaining to students whose first name with A through M is recorded in database_1 and data for those students whose first name starts with N through Z is in database_2*. For example, a student with the name "Justin Smith" who is taking course IDs "DB101", "ALGO201" would have two rows in the enrollment table of database_1, one for each course he's taking. Another student with name "Sanjit Patel" who is taking course IDs "DB101", "ALGO201" would have two rows in the enrollment table of database_2.

The **proposed solutio**n is a web server developed with the SpringBoot framework using the Java programming language. To meet the requirement of the problem, I used two MariaDB databases, which were created using SpringBoot JPA (*Figure 1*).

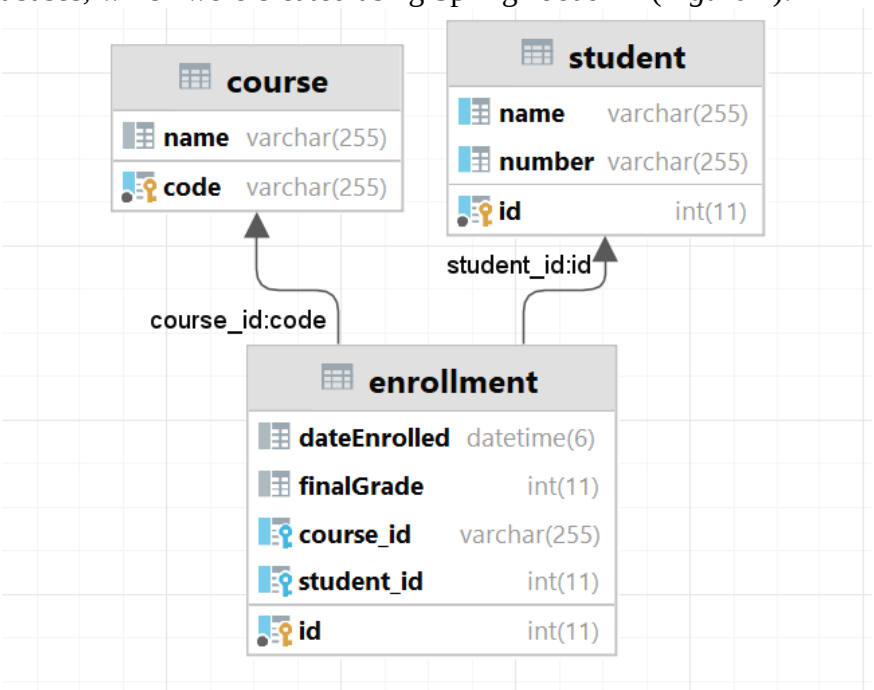

Figure 1. Structure of the database tables

To optimize memory usage, I have the same entities for both database 1 and database 2, so the tables are the same in both databases (*Figure 1*). I used two different repositories for each entity, one connected to the first database and the second connected to the second.

The following tables shows the implemented APIs and their purpose.

GET	/api/backend/students/	Returns all students from both databases
GET	/api/backend/students/db1	Returns all students with names between A and M
GET	/api/backend/students/db2	Returns all students with names between N and Z
POST	/api/backend/students/	Add a student to a database
DELETE	/api/backend/students/	Delete a student from both databases

Tabel 1. Endpoint-uri for Student

GET	/api/backend/courses/db1	Returns all courses from database 1
GET	/api/backend/courses/db2	Returns all courses from database 2
POST	/api/backend/courses/	Add a new course to both databases
DELETE	/api/backend/courses/{code}	Delete a course from both databases

Tabel 2. Endpoint-uri for Course

GET	/api/backend/enrollment/	Returns all enrollments from both databases (all the students)
GET	/api/backend/enrollment/db1	Returns all enrollments for students with names between A and M
GET	/api/backend/enrollment/db2	Returns all enrollments for students with names between N and Z
GET	/api/backend/enrollment/student?name=	Returns all the courses that a student is taking
GET	/api/backend/enrollment/course?code=	Returns all the students taking a certain course
GET	/api/backend/enrollment/students	Returns all the courses that are enrolled in by a given set of students
POST	/api/backend/enrollment/?name=&code=	Add an enrollment to database 1 or 2 based on student name
POST	/api/backend/enrollment/student?grade=&name=&code=	Add a final grade to a student based on his name

Tabel 3. Endpoint-uri for Enrollment

For testing the application I used Postman.

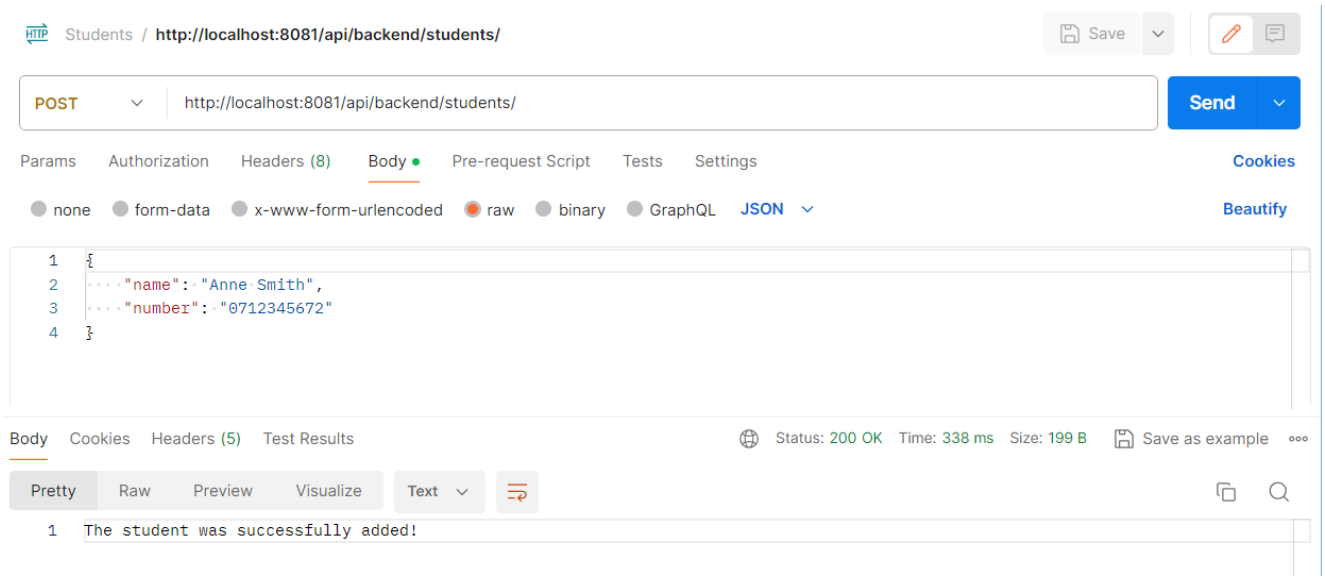

Figure 2. Adding a new student

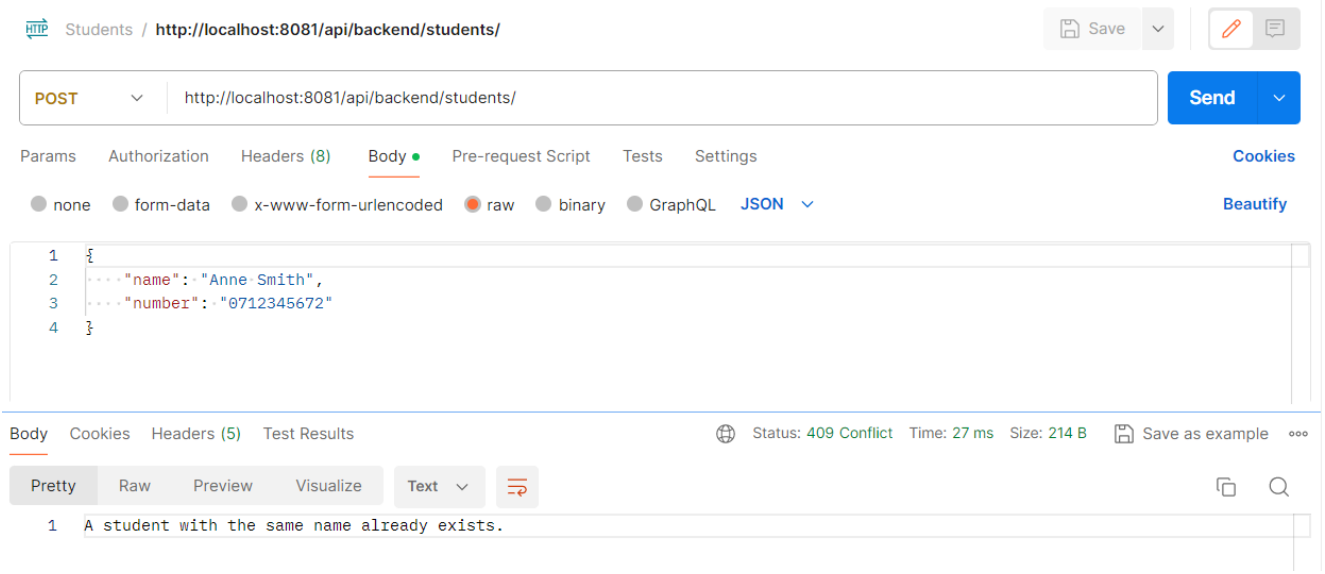

Figure 3. Trying to add an existing student

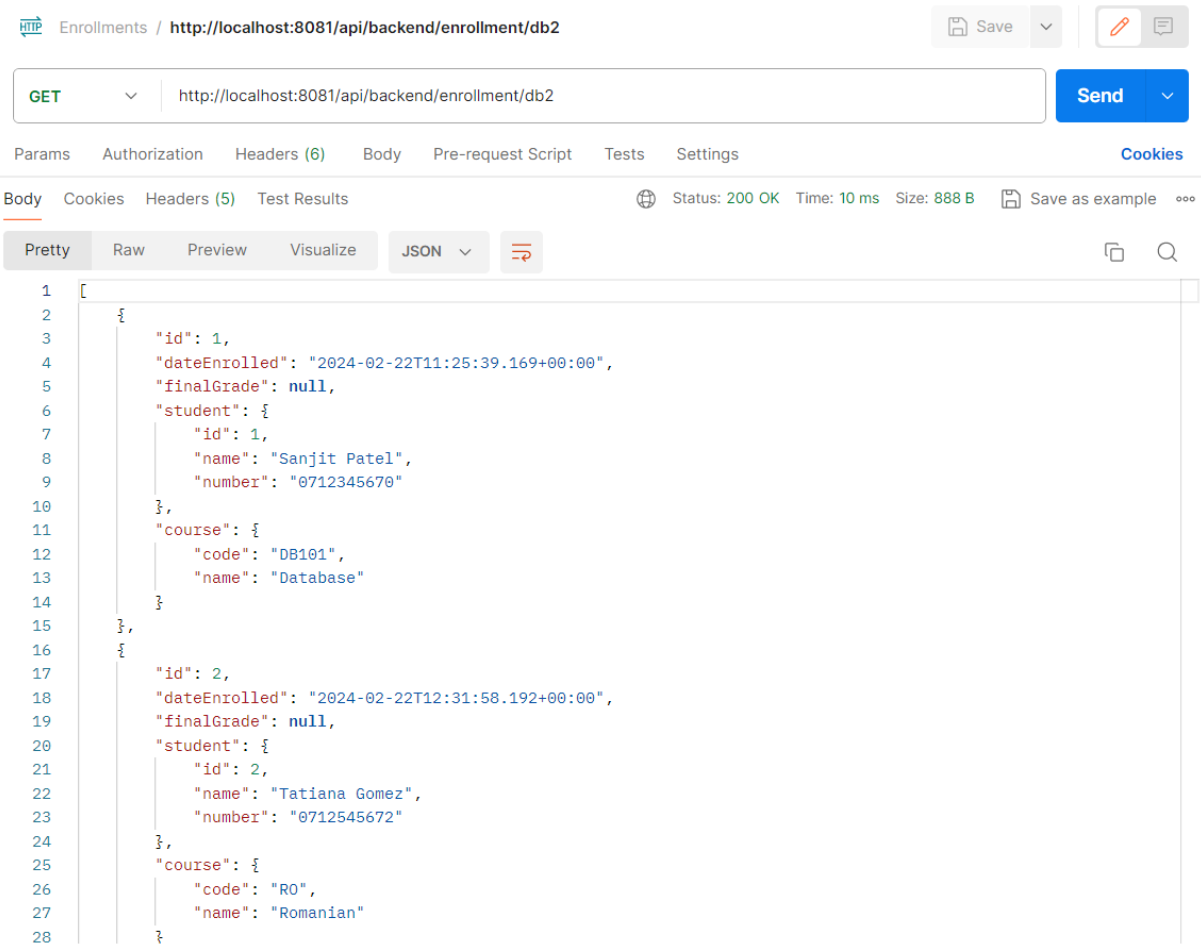

Figure 4. Find all the enrollments in database 2

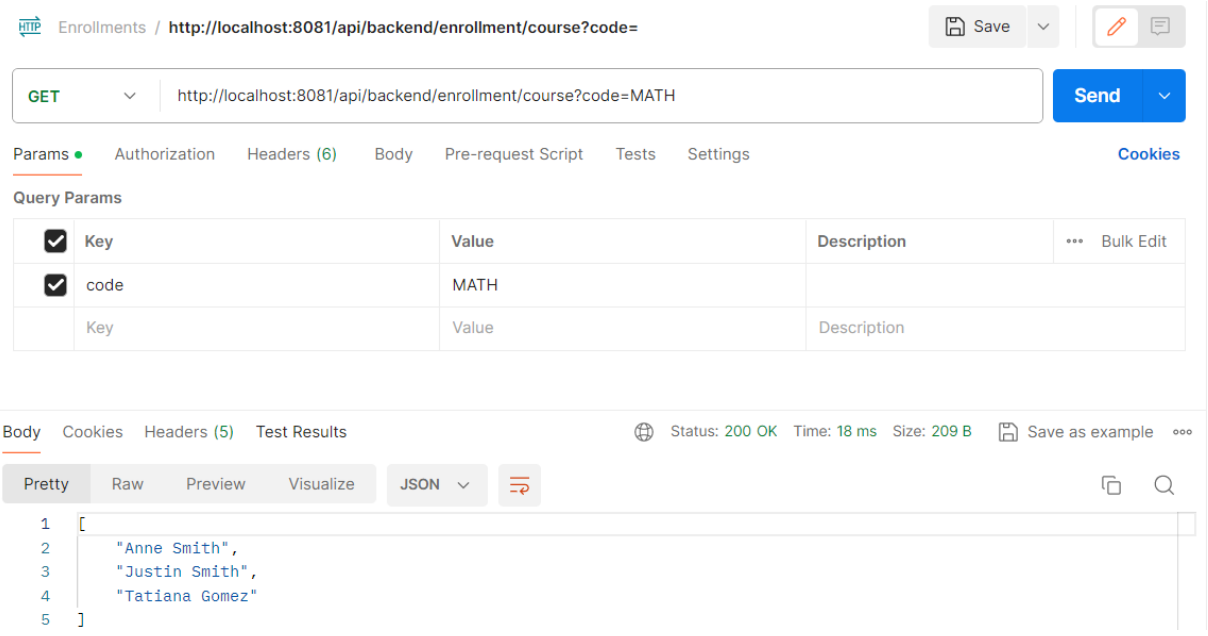

Figure 5. Find all the student from a course

The implementation can be found on my GitHub account, and the testing part in the Postman Workspace.

https://github.com/lungu-stefania-paraschiva/University-enrollment-system

https://www.postman.com/stefania-lungu/workspace/university-enrollment-backend