**Note : StudentResgisterationApplication is main class which need to be executed to start spring boot application.**

**Model Student Course Registration – Use whatever technology you prefer, not necessarily JPA/Hibernate (preferred since that is what we use).**

Student: ID, Name

Course: ID, Name

A student can take many courses and a course can have many students.

1).Write skeleton code of entity bean (or whatever technologies you like) classes to model Student and Course and student course registration.

**Tech Used :**

**1. Spring boot**

**2. H2 in memory Database**

**3. Spring jpa repository**

**4. Java**

**5. Maven**

**Java Model Classes :**

**1.**Course

**2.**Student

**3.**StudentCourse

**Entity Models: - H2 database Tables.**

**1.** STUDENTS (ID column is pk)

**ID NAME**

**2.** COURSES (ID column is pk)

**ID NAME**

**3.** STUDENT\_COURSES (ID column is pk)

**ID COURSE\_ID STUDENT\_ID**

2). Write a skeleton Student DAO class that support

**Refere** : **StudentRepository**, **CourseRepository**, **StudentCourseRepository,**  for more details refer dao layer in code.

2.1). add a new student along with their course registrations.

Ans : API - **http://localhost:8080/registerStudent**

2.2). Delete a student.

Ans : **API** **- http://localhost:8080/deleteStudent?stundentId=10**

2.3).Get all students, sorted by their name, for a given course with course name as input.

Ans : **API** **- http://localhost:8080/getStudentsByCourseId?courseId=1**

**Bonus Points**:

2.4). What if we want to record course scores? What possible changes need to be made?   
 Explain briefly.

Ans : **we can add score column in** STUDENT\_COURSES **table**

2.5). How to find all students who don’t register for a given course?

Ans : **API- http://localhost:8080/getNonRegisterStudentsForCourseId?courseId=1**

**\*\*Query: "select sco from StudentCourse sco where sco.studentId not in (SELECT sc.studentId FROM StudentCourse sc WHERE sc.courseId = :courseId)"**

**This custom query is present StudentCourseRepository class.**

**Notes**:

1. Code skeleton is enough. No need to write every single line.
2. Need both entity bean classes as well as table DDLs.
3. For the DAO/Repository classes need query details for relevant questions.
4. Show proper transaction management.
5. **Show best practice(s)** when you can solve the problems with multiple approaches. If possible comment why one approach is better than the others.
6. We love Hibernate and also hate it!! Would love to hear your opinion on best practices.

**Time**: 45 – 60 minutes.