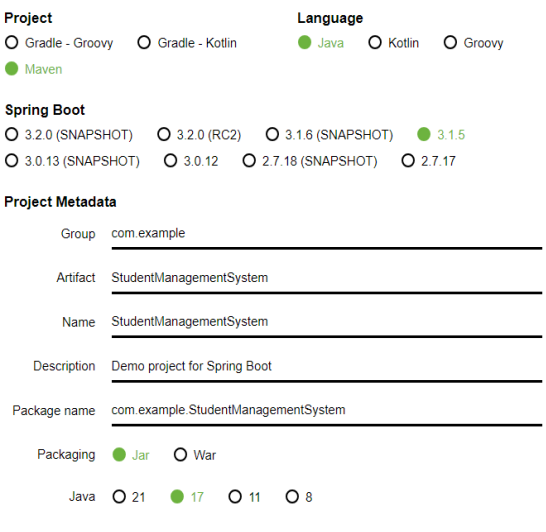
Buddy Application

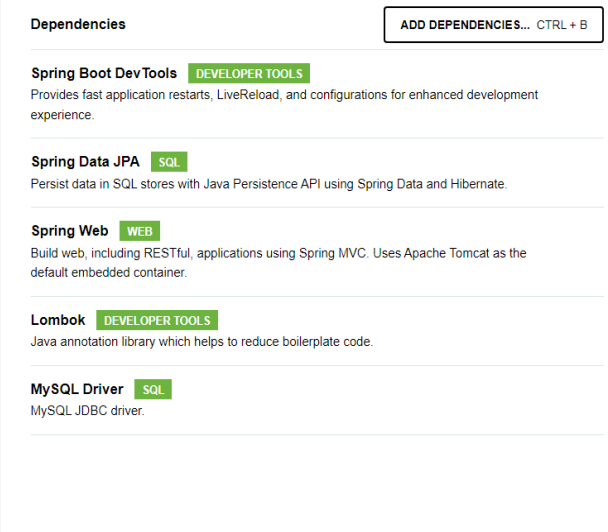
Introduction

The Student Management System is a web-based application designed to streamline and manage student-related tasks within an educational institution. It serves as a centralised platform for handling student information and operations efficiently. The application's primary aim is to facilitate the management of student records and tasks for educational institutions, enabling administrators and faculty to handle student-related activities seamlessly.

❖ Step 1 (Spring Initializr):

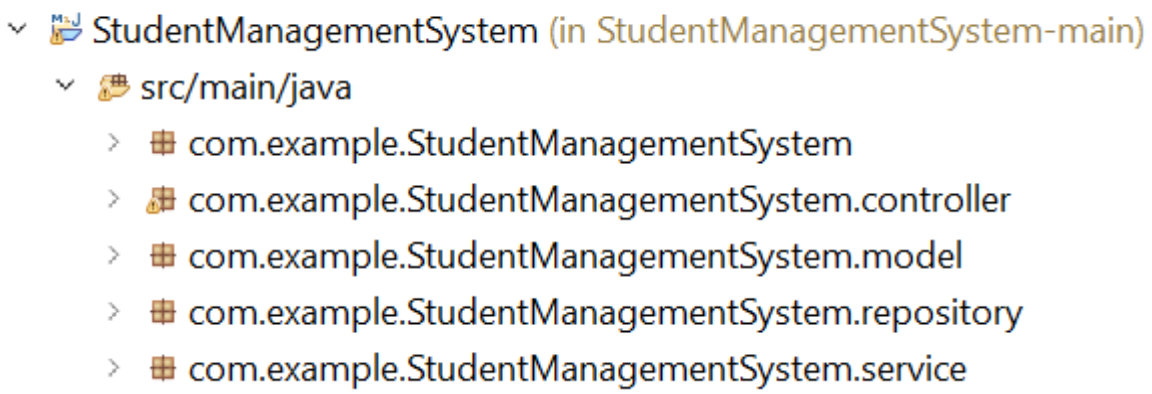
➔ Go to Spring Initializr and create a project with the following dependencies as shown below:





❖ Step 2 (Creating Packages):

➔ Create the following package inside the package as shown below:



❖ Step 3 (Student Entity Class):

➔ Create a new Student entity class in the model package with the following listed attributes.

1. sid (int)
2. name (String)
3. age (int)
4. course (String)
5. address (String)
6. emailId (String)
7. contact (String)

➔ Add required annotation for this entity class and Lombok annotations to auto-generate the default and all arg constructors, getters, and setters for the attributes.

❖ Step 4 (StudentsDal Interface):

➔ Create an interface named **StudentsDal** in the repository package**.**

➔ Add proper annotations and extend the interface with *JpaRepository*<> with the required parameters.

❖ Step 5 (StudentsService Class):

➔ Create a **StudentsService** class and annotate it with *@Service* to mark it as a service layer.

➔ Declare a class variable of *StudentsDal* and make it final.

➔ Define a constructor that injects an instance of the *StudentsDal*.

➔ Create the following methods:

- *increementService(int num)* - The method adds 1 to the input number and returns the incremented value.
- *addStudent(Student student)* - The method saves the provided student entity using the studentDal repository's "save" method and returns the saved student object.
- *deleteStudent(int sid*) - The method attempts to delete a student by ID using the studentDal repository and returns true if successful; otherwise, it returns false.
- getStudents() - The method retrieves and returns a list of all student records from the data access layer using the "findAll()" method of the studentDal repository.

❖ Step 6 (studentsController):

➔ Create a **studentsController** class by adding the required *@RestController* and *@RequestMapping("/students")* annotation.

➔ Add *@AllArgsConstructor* and *autowire* StudentsService class.

➔ Create a *getStudents()* method to handle a GET request for retrieving a list of all students. Annotate it with *@GetMapping* and specify the path ***"/getStudents"***.

➔ Create an *addStudent()* method to handle a POST request for adding a new student. Annotate it with *@PostMapping* and specify the mapping path ***"/addStudent"***. This method will take a **Student** object as the request body and call the *addStudent* method of the **StudentsService** to create the student.

➔ Create a *deleteStudent()* method to handle a DELETE request for deleting a student by its sid. Annotate it with *@DeleteMapping* and specify the mapping path ***"/deleteStudent/{sid}"***. This method will take a **Student sid** as the PathVariable and call the *deleteStudent* method of the **StudentsService** to delete a student. This method returns a String (***Student does not exist!***) if student by sid is not found else it returns ***"Deleted Student SID: "+ sid***.

❖ Step 7 (application.yml):

➔ Create an **application.yml** file with ***MySQL*** and ***hibernate*** database configurations.

Please find the Final code of the **StudentManagement Application** provided below for reference: Link