Iamneo | Rest API with Spring Boot and JPA

Objective

To build a complete REST API backend

Day 1 | Class Exercise Lab 2 | Introduction to REST API

In this Lab, we will create three services using proper URIs and HTTP methods:

1. *@GetMapping("/courses"):*
   Retrieve all the courses
   API Endpoint uri - /course.
2. *@GetMapping("/course/{courseId}"):*
   Retrieve specific course
   API Endpoint uri - /course/{courseId}.

What will you learn?

You will learn

- What is a REST Service?
- How to bootstrap a Rest Service application with Spring Initializr?
- How to create a Get REST Service for retrieving the courses that a student registered for?
- How to create a Post REST Service for registering a course for student?
- How to execute Rest Services from Postman?

User Story #1 | Tools and Setup

You should be able to install the following tools in the system
* Maven 3.0+ is your build tool
* Your favourite IDE. We use Eclipse.
* JDK 17
* Postman or Swagger

User Story #2 | Bootstrap a Rest Service application with Spring Initializr

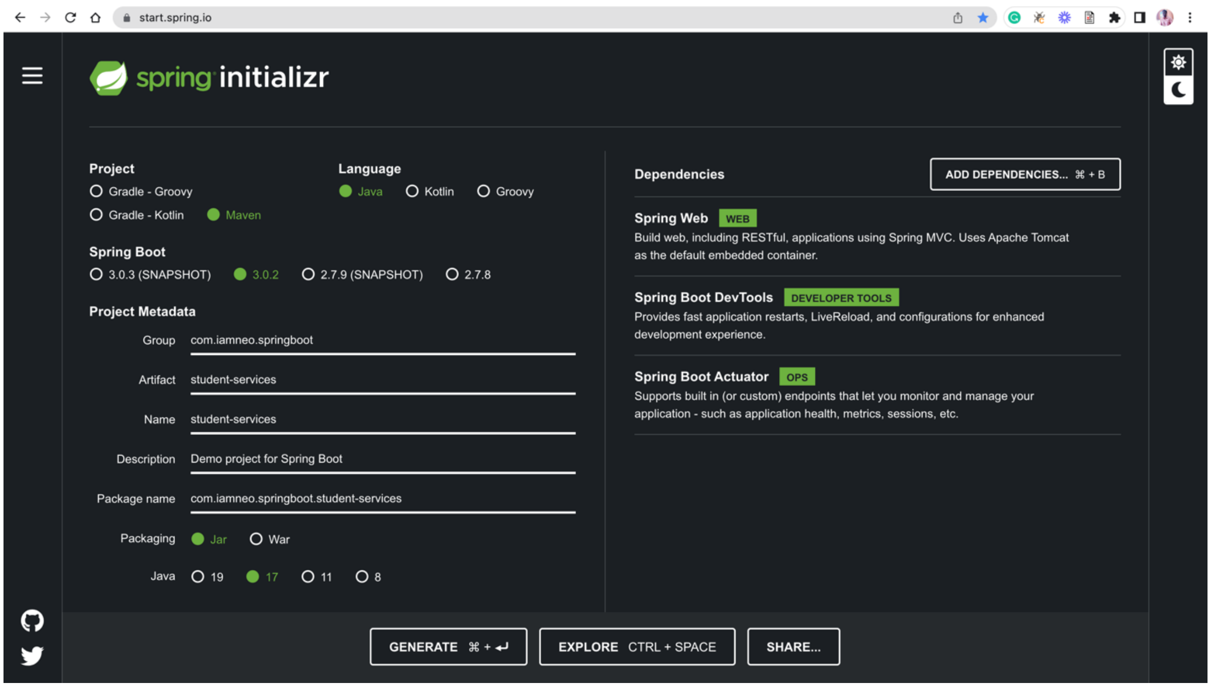
You should be able to bootstrap the REST Services with Spring Initializr. To do this, click [https://start.spring.io/]. Spring Initializr [http://start.spring.io/] is great tool to bootstrap your Spring Boot projects.

* Launch Spring Initializr and choose the following
* Choose com.iamneo.springboot as Group
* Choose student-services as Artifact
* Choose the following dependencies

- Web
- Actuator
- DevTools

* Check if maven and Jdk 17 are selected.
* Click Generate Project.
* Import the project into Eclipse. File -> Import -> Existing Maven Project.

Output Screenshot:



start.spring.io

spring initializr

Project
Gradle - Groovy
Gradle - Kotlin [✓ Maven]

Language
[✓ Java] Kotlin Groovy

Spring Boot
3.0.3 (SNAPSHOT) [✓ 3.0.2] 2.7.9 (SNAPSHOT) 2.7.8

Project Metadata

Group	com.iamneo.springboot
Artifact	student-services
Name	student-services
Description	Demo project for Spring Boot
Package name	com.iamneo.springboot.student-services
Packaging	[✓ Jar] War
Java	19 [✓ 17] 11 8

Dependencies

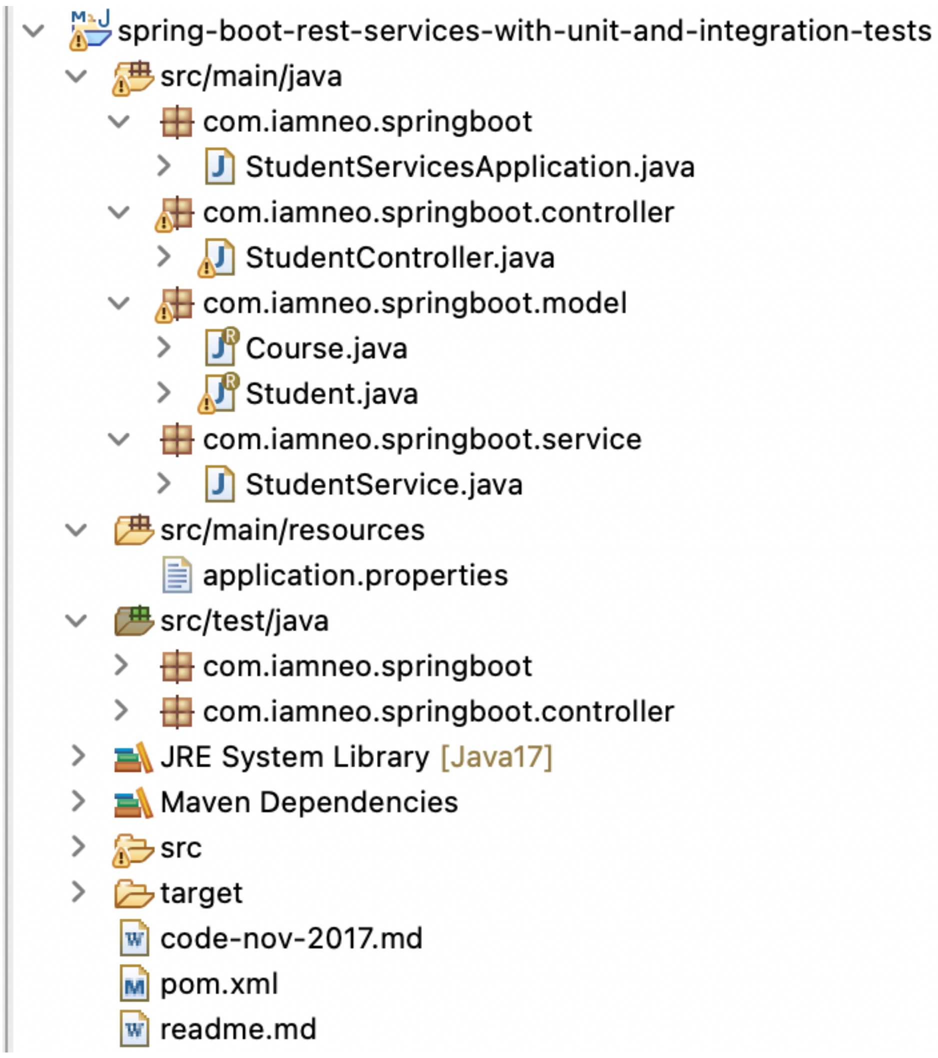
Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Spring Boot Actuator OPS
Supports built in (or custom) endpoints that let you monitor and manage your application - such as application health, metrics, sessions, etc.

User Story #3 | Create the project structure

Reference Project Structure



```
spring-boot-rest-services-with-unit-and-integration-tests
    src/main/java
        com.iamneo.springboot
            StudentServicesApplication.java
        com.iamneo.springboot.controller
            StudentController.java
        com.iamneo.springboot.model
            Course.java
            Student.java
        com.iamneo.springboot.service
            StudentService.java
    src/main/resources
        application.properties
    src/test/java
        com.iamneo.springboot
        com.iamneo.springboot.controller
    JRE System Library [Java17]
    Maven Dependencies
    src
    target
    code-nov-2017.md
    pom.xml
    readme.md
```

A few details:
StudentController.java - Rest controller exposing all three service methods discussed above.

Course.java, Student.java, StudentService.java - Business Logic for the application.
StudentService exposes a couple of methods we would consume from our Rest Controller.

StudentServicesApplication.java - Launcher for the Spring Boot Application. To run the application, just launch this file as Java Application.

pom.xml - Contains all the dependencies needed to build this project. We will use Spring Boot Starter Web.

Implement the Model Classes

A student can take multiple courses. A course has an id, name, description and a list of steps you need to complete to finish the course. A student has an id, name, description and a list of courses he/she is currently registered for.

User Story #4 | Create a Model Class Course.java

Create a record class called Course.java inside com.iamneo.springboot.model and define the attributes

Course.java

```
public record Course(String id,
                     String name,
                     String description,
                     List<String> steps) {
    public void setId(String id) {
    }
}
```

Implement the Business Layer

User Story #5 | Create a class called StudentService.java and implement the following methods

- public List<Course> retrieveAllStudents() - Retrieve details for all courses
- public Student retrieveStudent(Course CourseId) - Retrieve a specific course details

Implement the REST Controller

Adding a couple of GET Rest Services

User Story #6 | Implement the StudentController.java

The Rest Service StudentController exposes a couple of get services.

- @Autowired private StudentService studentService : We are using Spring Autowiring to wire the student service into the StudentController.
- @GetMapping("/courses "): Exposing a Get Service

- @GetMapping("/course/{courseId}"): Exposing a Get Service for retrieving specific course.

User Story #7 | Executing the Http Get Operation Using Postman

We will access a request to http://localhost:8080/course to test the service.