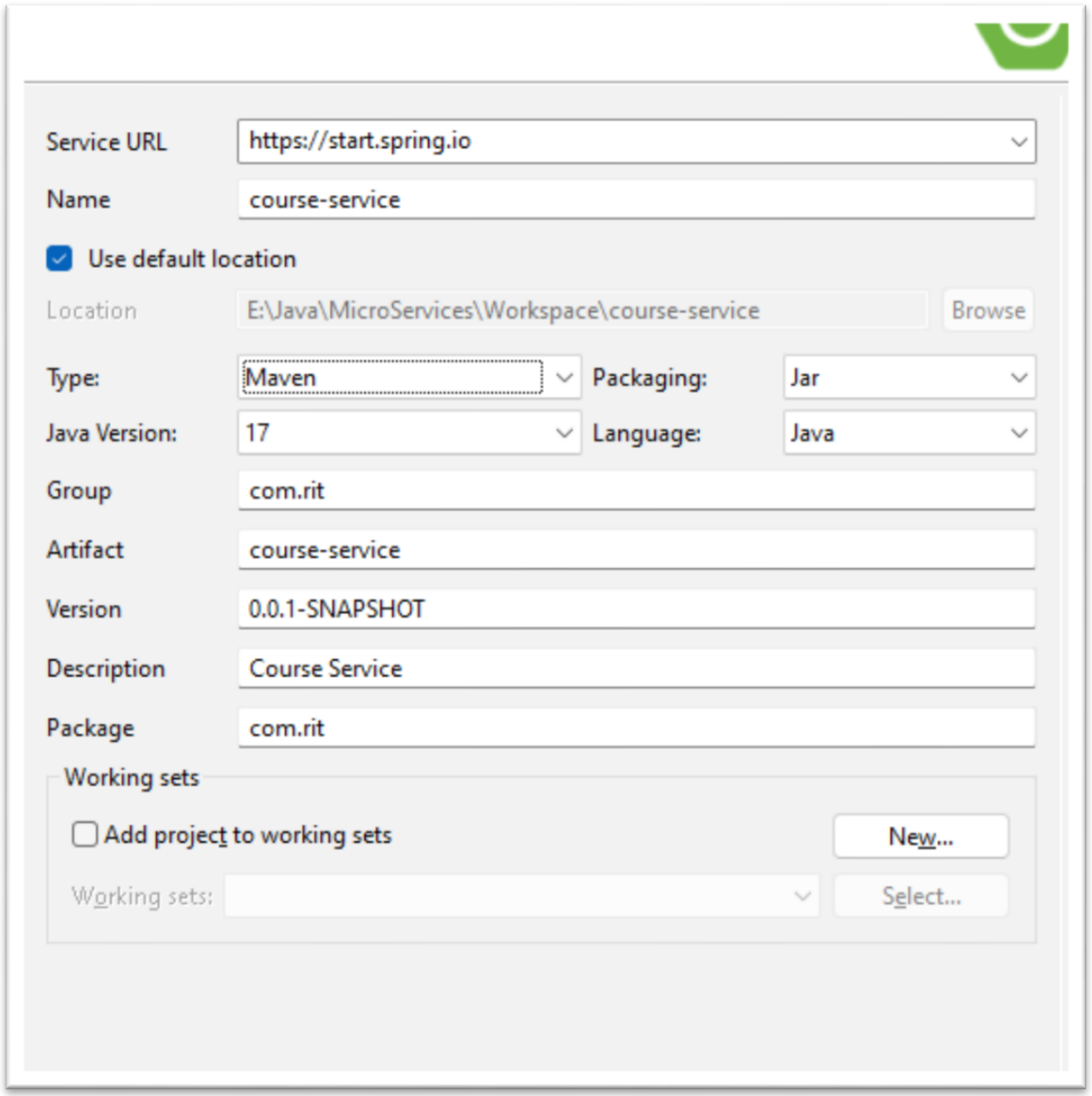
# Microservices

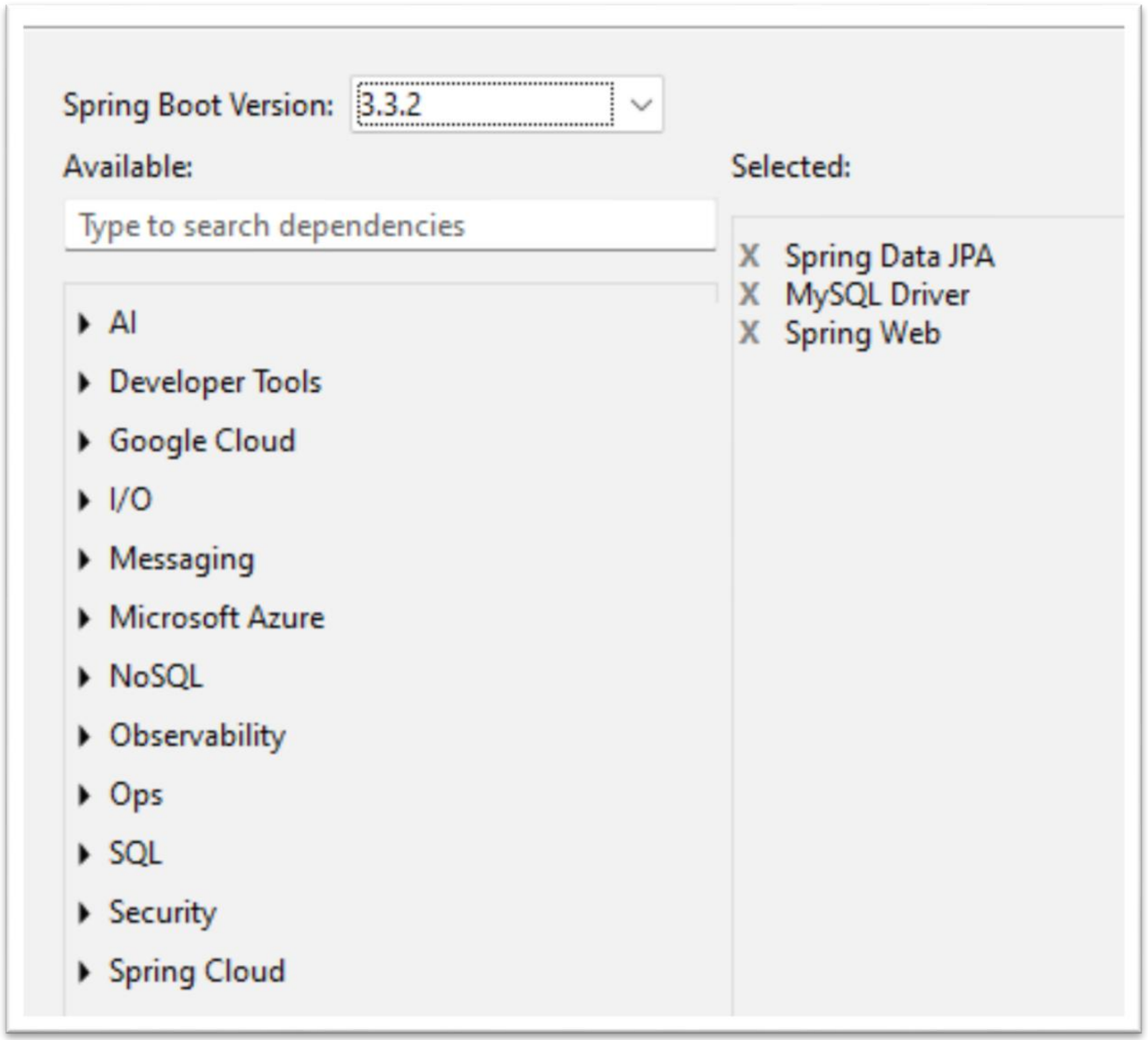
Spring Boot

## Course Microservice


| | |
|---|---|
| Service URL | https://start.spring.io |
| Name | course-service |
| ☑ Use default location | |
| Location | E:\Java\MicroServices\Workspace\course-service |
| Type: | Maven |
| Packaging: | Jar |
| Java Version: | 17 |
| Language: | Java |
| Group | com.rit |
| Artifact | course-service |
| Version | 0.0.1-SNAPSHOT |
| Description | Course Service |
| Package | com.rit |

Working sets

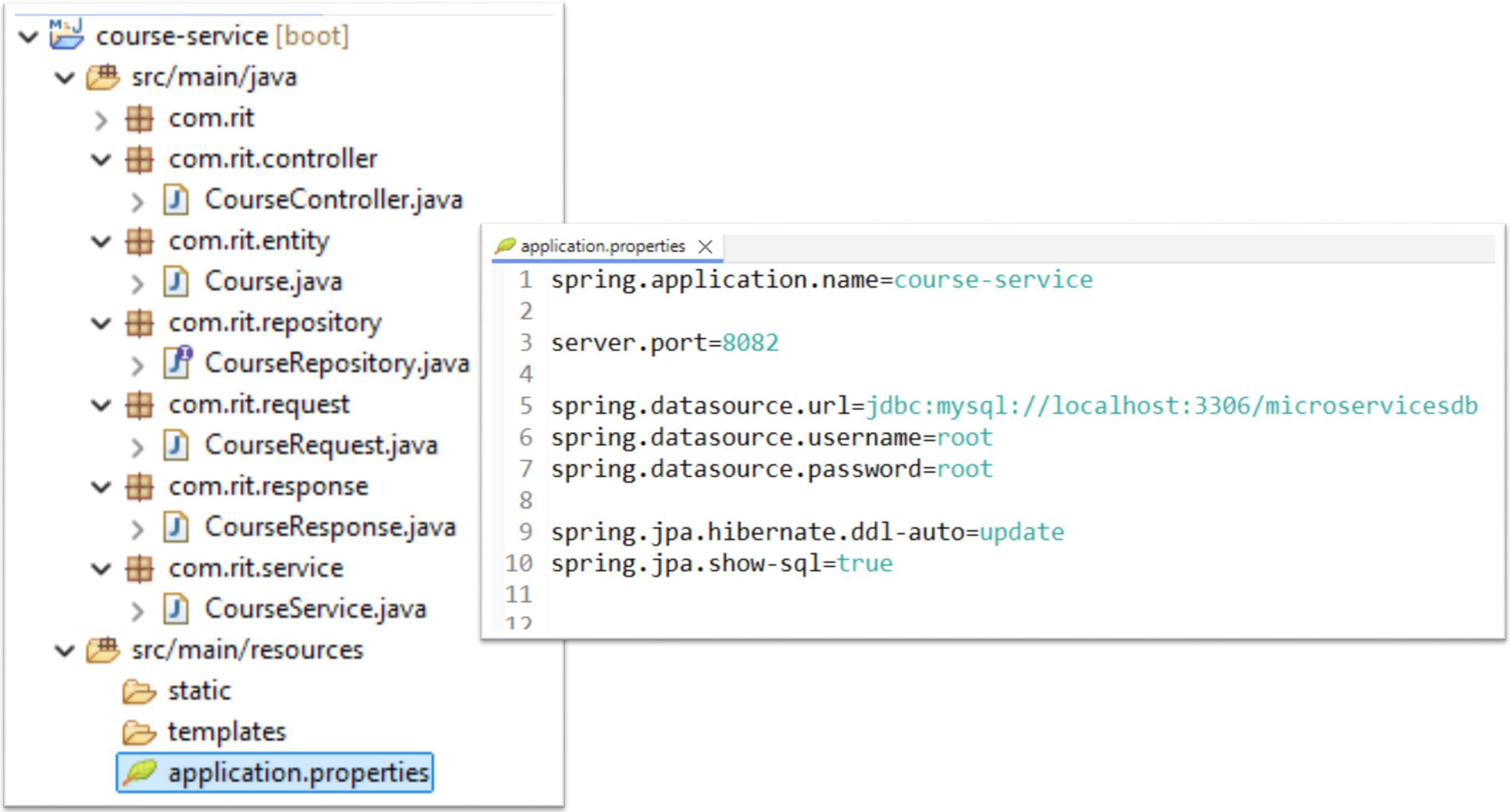
☐ Add project to working sets


Spring Boot Version: 3.3.2

Available:

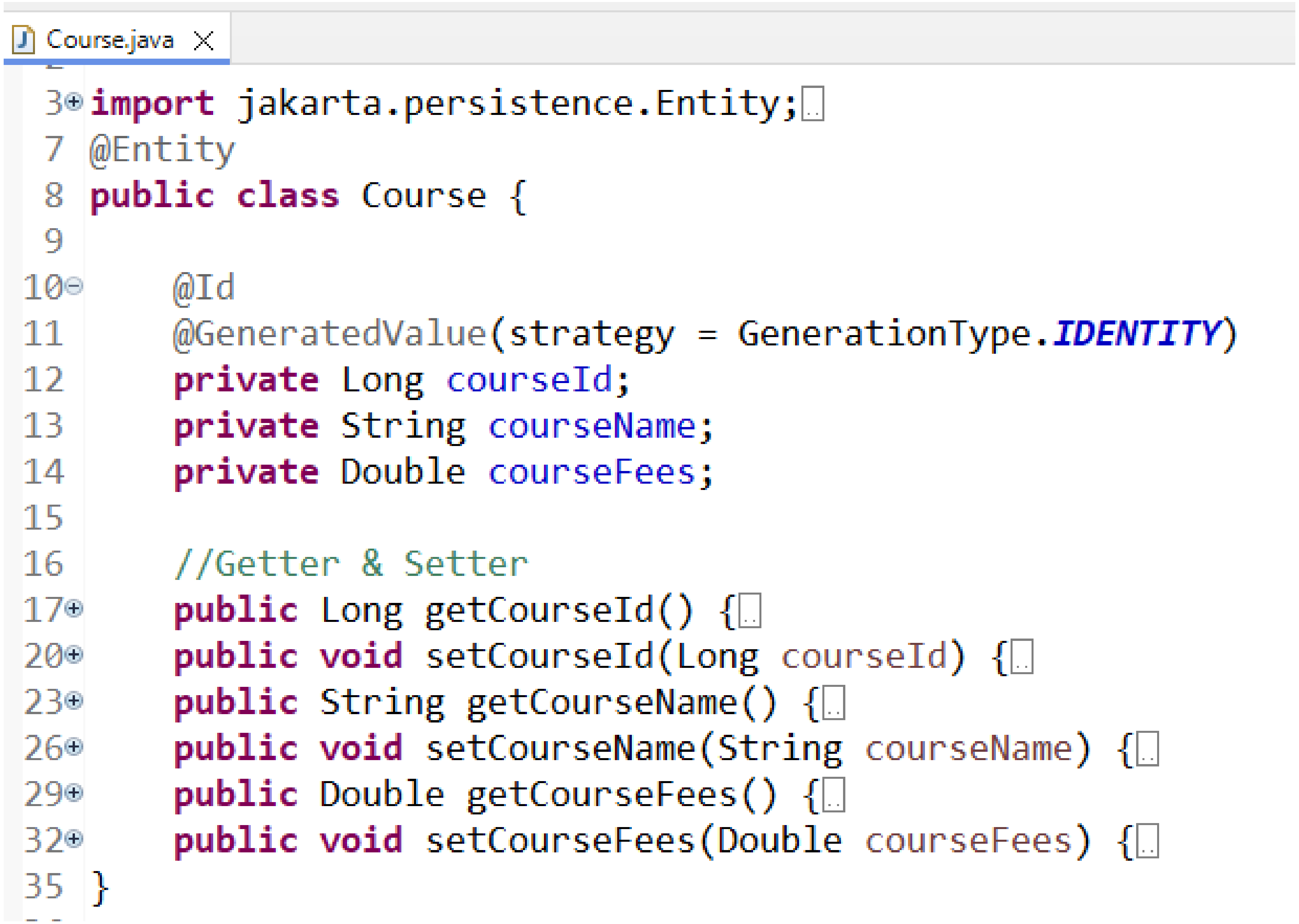
Selected:

- Spring Data JPA
- MySQL Driver
- Spring Web

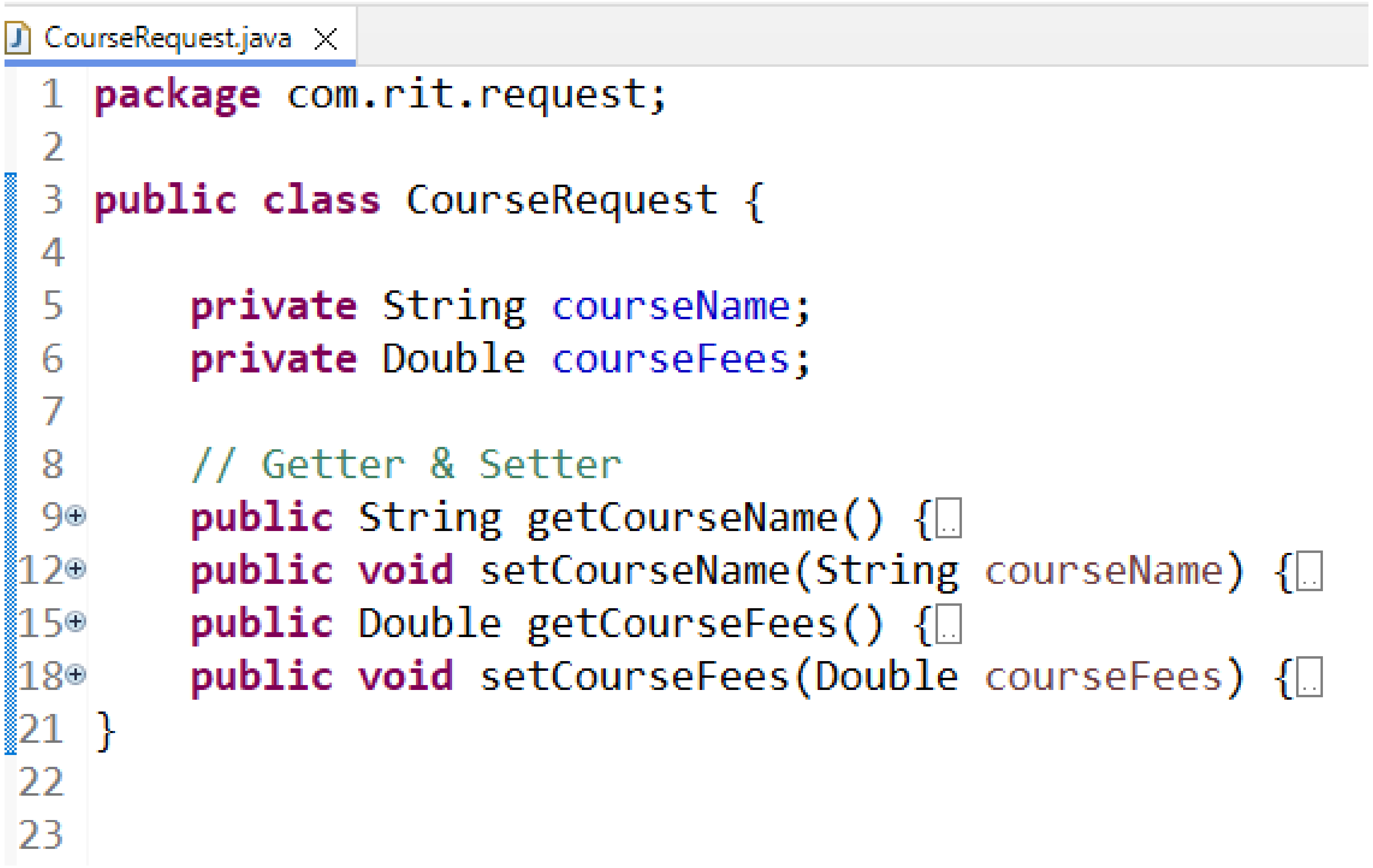

```
course-service [boot]
  src/main/java
    com.rit
    com.rit.controller
      CourseController.java
    com.rit.entity
      Course.java
    com.rit.repository
      CourseRepository.java
    com.rit.request
      CourseRequest.java
    com.rit.response
      CourseResponse.java
    com.rit.service
      CourseService.java
  src/main/resources
    static
    templates
    application.properties
```

application.properties

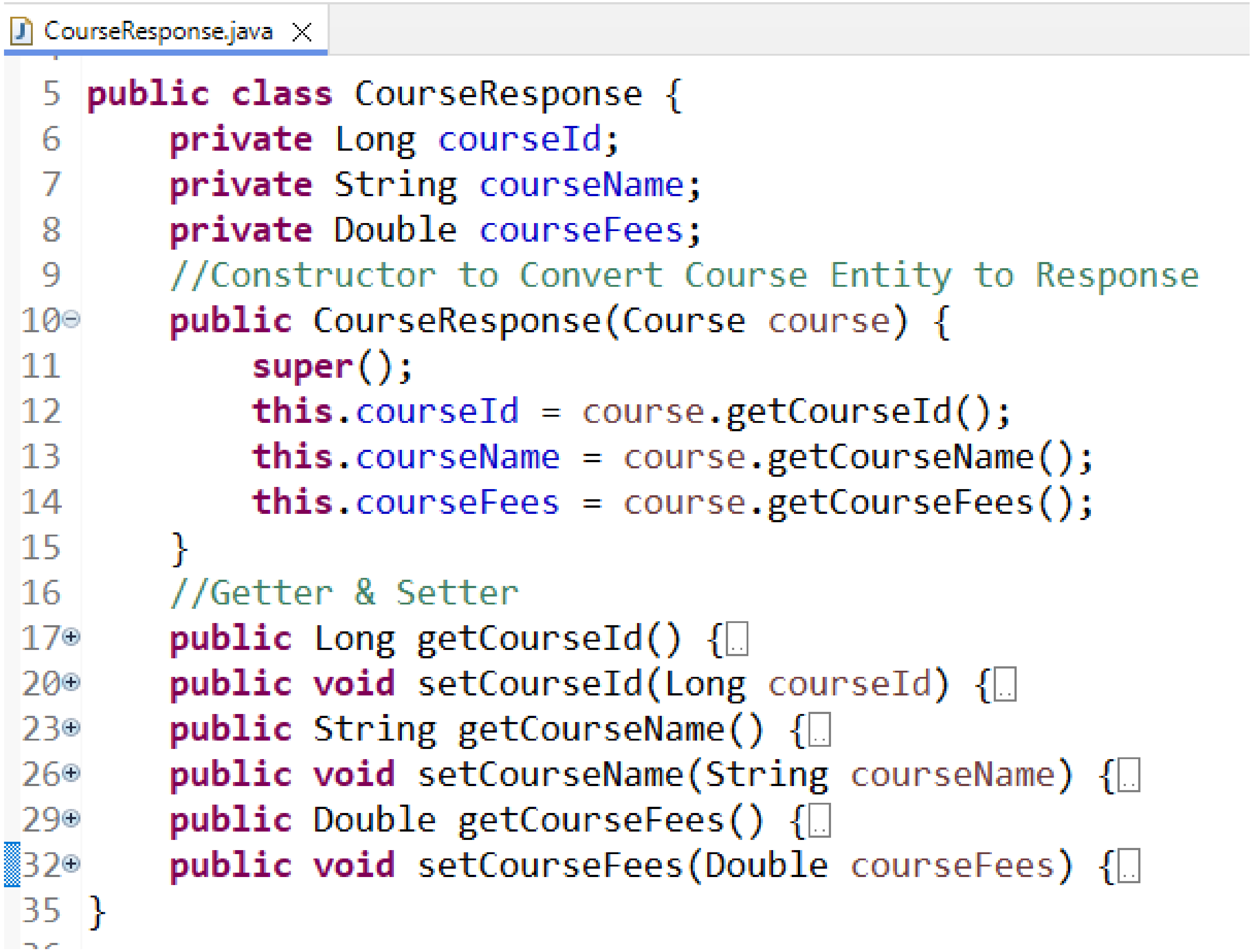
```
spring.application.name=course-service

server.port=8082

spring.datasource.url=jdbc:mysql://localhost:3306/microservicesdb
spring.datasource.username=root
spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```



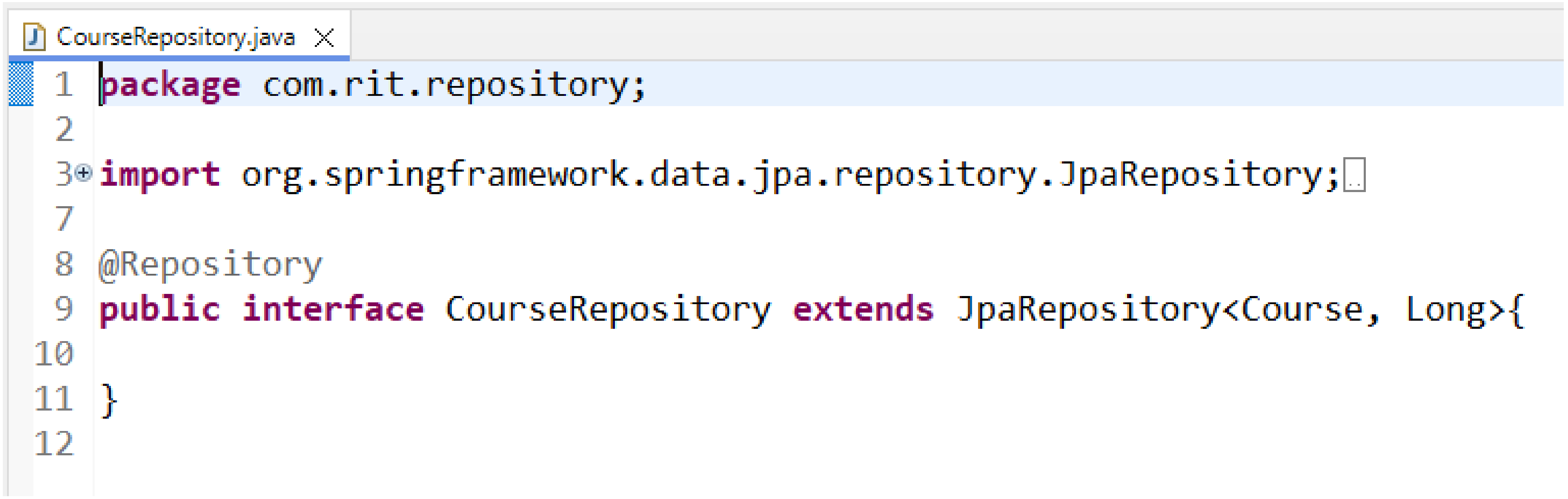
```java
import jakarta.persistence.Entity;
@Entity
public class Course {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long courseId;
    private String courseName;
    private Double courseFees;

    //Getter & Setter
    public Long getCourseId() {
    public void setCourseId(Long courseId) {
    public String getCourseName() {
    public void setCourseName(String courseName) {
    public Double getCourseFees() {
    public void setCourseFees(Double courseFees) {
}
```


CourseRequest.java

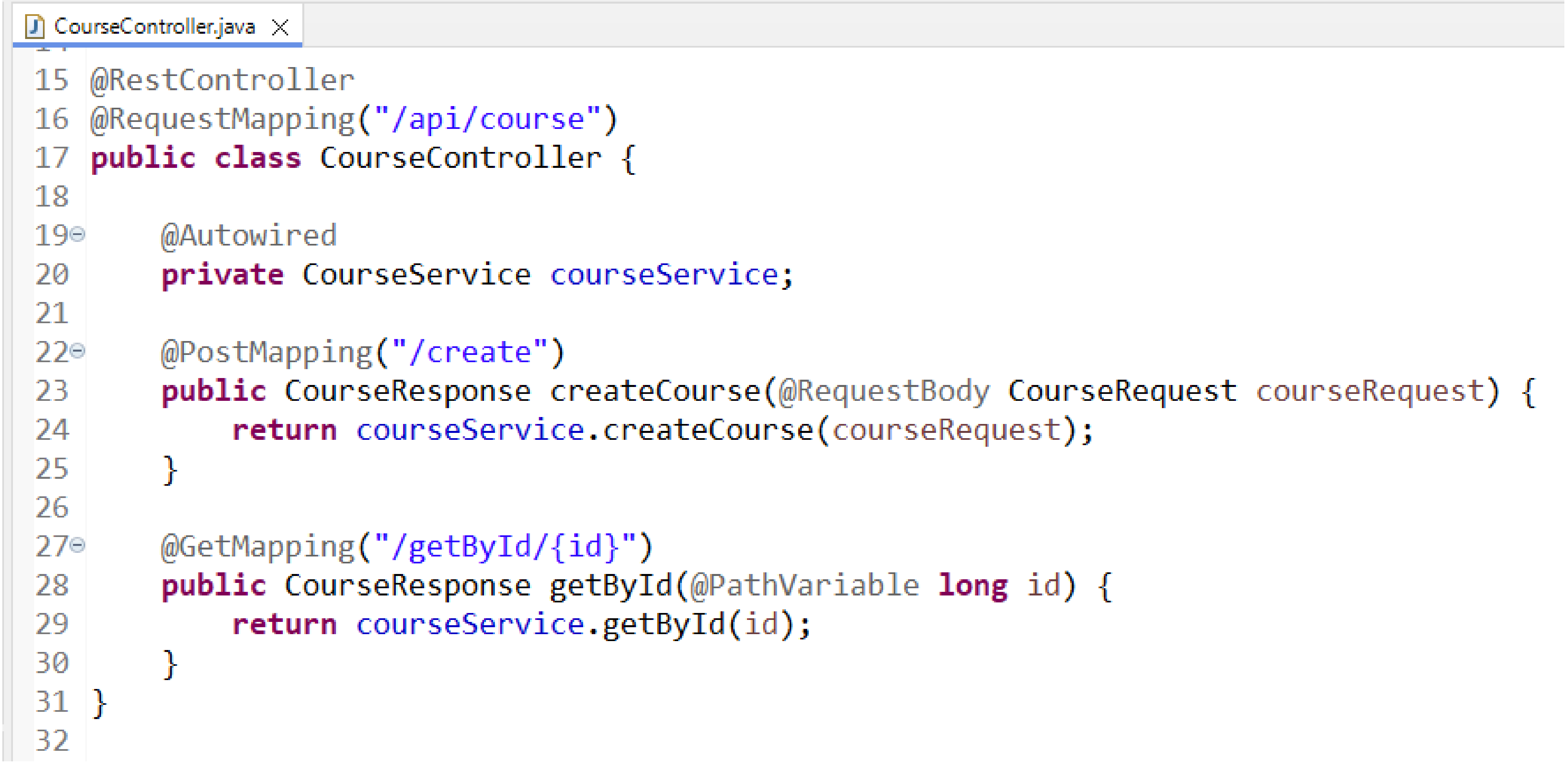
```java
package com.rit.request;

public class CourseRequest {

    private String courseName;
    private Double courseFees;

    // Getter & Setter
    public String getCourseName() {
    public void setCourseName(String courseName) {
    public Double getCourseFees() {
    public void setCourseFees(Double courseFees) {
}


```


CourseResponse.java

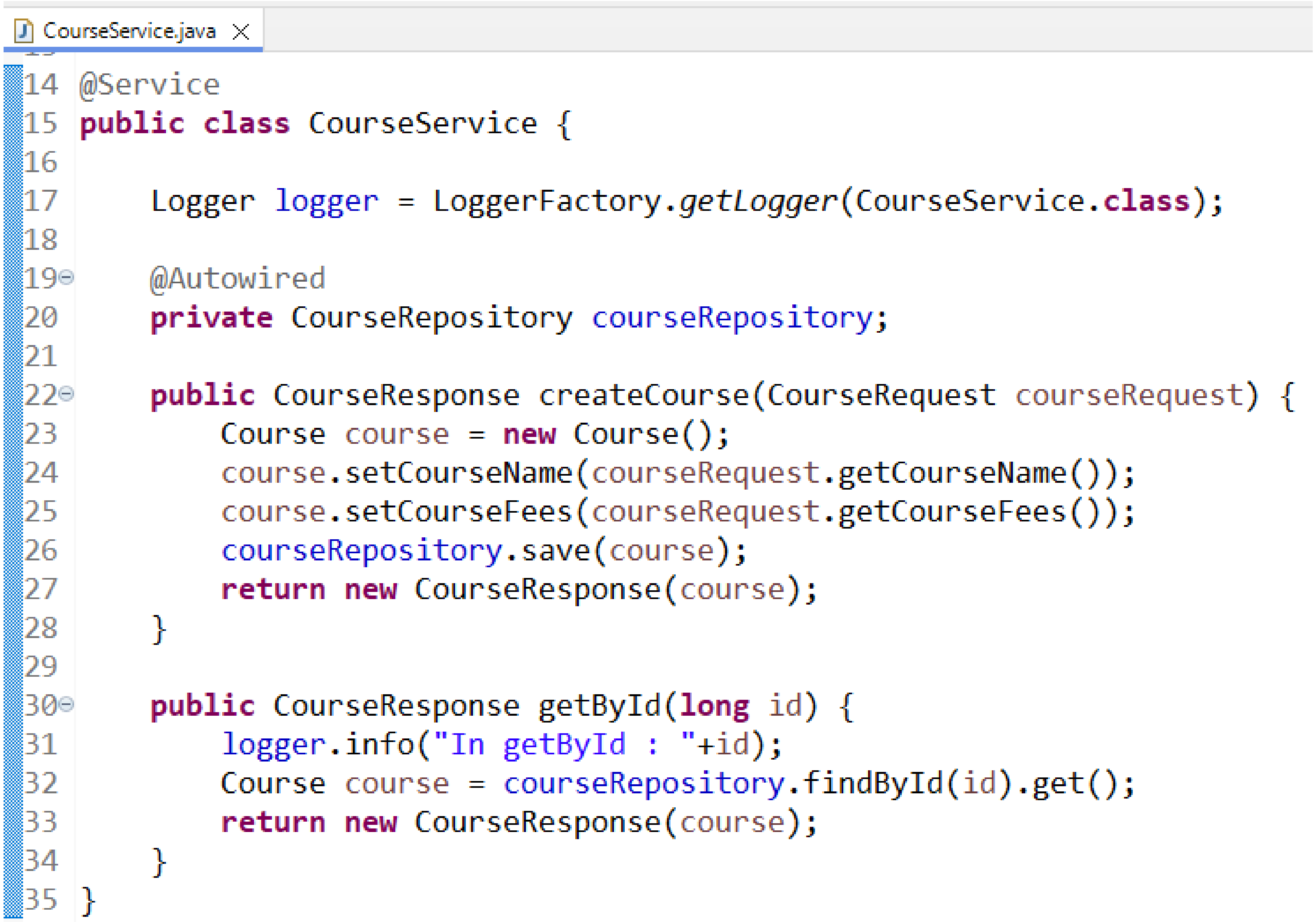
```java
public class CourseResponse {
    private Long courseId;
    private String courseName;
    private Double courseFees;
    //Constructor to Convert Course Entity to Response
    public CourseResponse(Course course) {
        super();
        this.courseId = course.getCourseId();
        this.courseName = course.getCourseName();
        this.courseFees = course.getCourseFees();
    }
    //Getter & Setter
    public Long getCourseId() {
    public void setCourseId(Long courseId) {
    public String getCourseName() {
    public void setCourseName(String courseName) {
    public Double getCourseFees() {
    public void setCourseFees(Double courseFees) {
}
```


CourseRepository.java

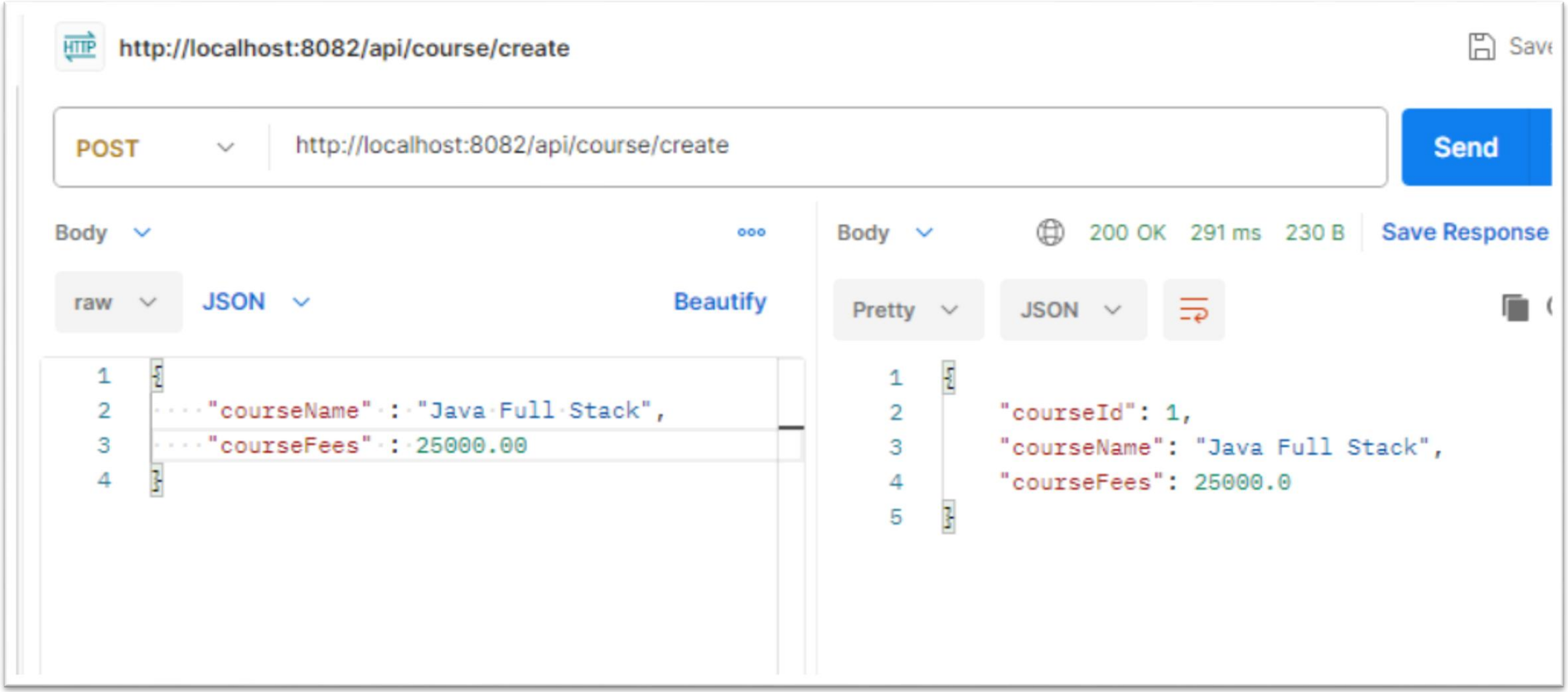
```java
package com.rit.repository;

import org.springframework.data.jpa.repository.JpaRepository;

@Repository
public interface CourseRepository extends JpaRepository<Course, Long>{

}

```



```java
@RestController
@RequestMapping("/api/course")
public class CourseController {

    @Autowired
    private CourseService courseService;

    @PostMapping("/create")
    public CourseResponse createCourse(@RequestBody CourseRequest courseRequest) {
        return courseService.createCourse(courseRequest);
    }

    @GetMapping("/getById/{id}")
    public CourseResponse getById(@PathVariable long id) {
        return courseService.getById(id);
    }
}

```



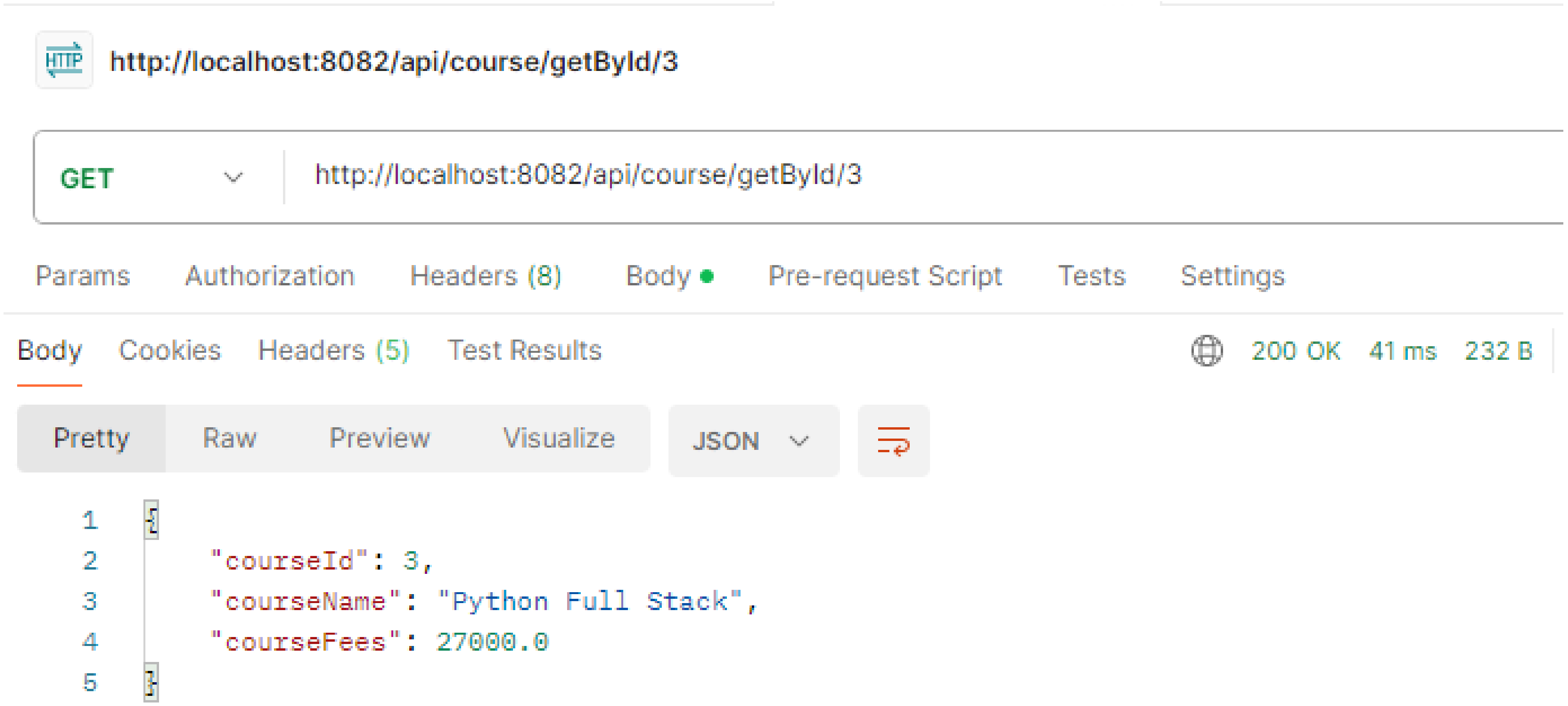
```java
@Service
public class CourseService {

    Logger logger = LoggerFactory.getLogger(CourseService.class);

    @Autowired
    private CourseRepository courseRepository;

    public CourseResponse createCourse(CourseRequest courseRequest) {
        Course course = new Course();
        course.setCourseName(courseRequest.getCourseName());
        course.setCourseFees(courseRequest.getCourseFees());
        courseRepository.save(course);
        return new CourseResponse(course);
    }

    public CourseResponse getById(long id) {
        logger.info("In getById : "+id);
        Course course = courseRepository.findById(id).get();
        return new CourseResponse(course);
    }
}
```


http://localhost:8082/api/course/create

```
http://localhost:8082/api/course/create
```

```
{
    "courseName" : "Java Full Stack",
    "courseFees" : 25000.00
}
```

```
{
    "courseId": 1,
    "courseName": "Java Full Stack",
    "courseFees": 25000.0
}
```



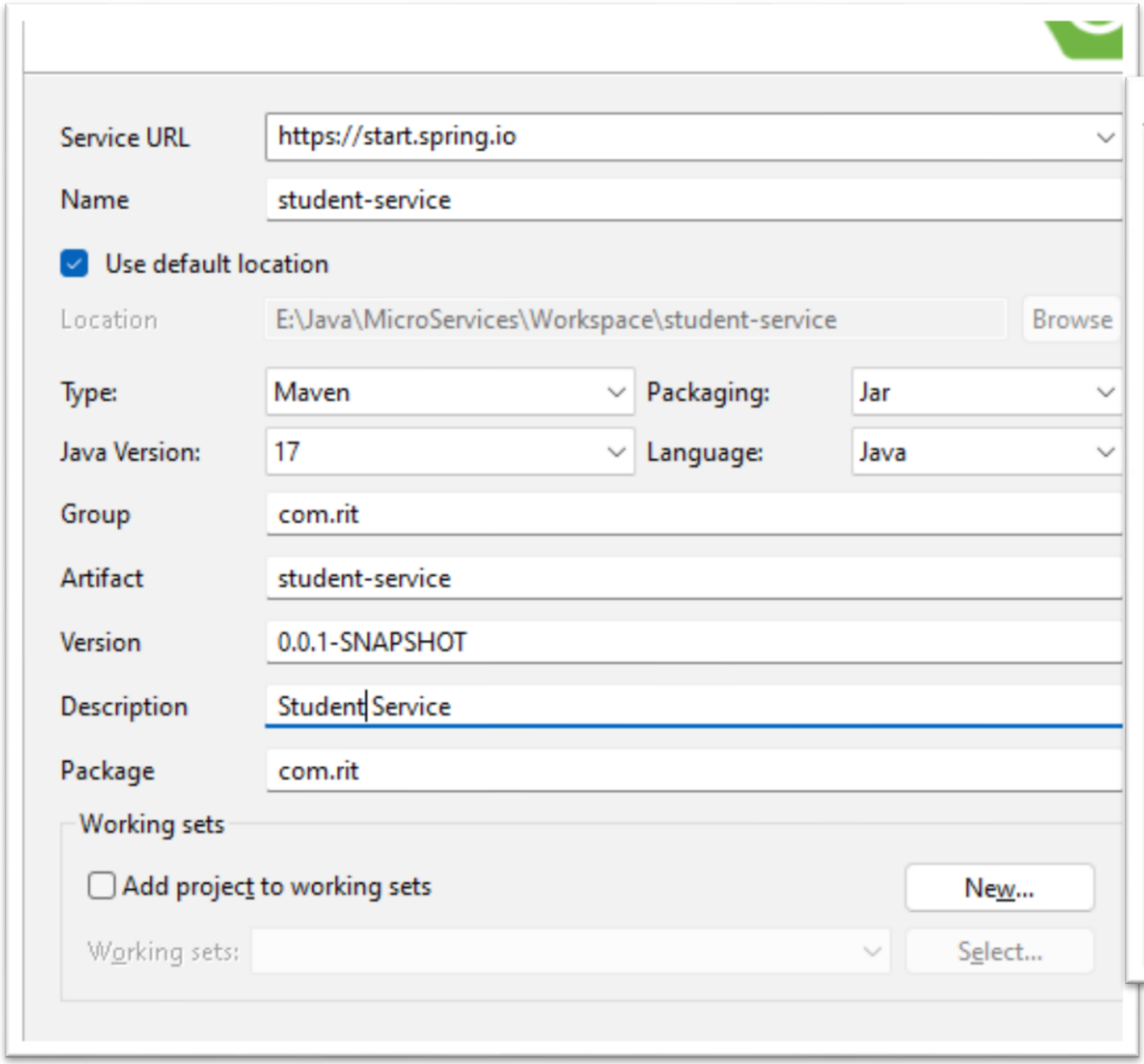
http://localhost:8082/api/course/getById/3

GET http://localhost:8082/api/course/getById/3

200 OK 41 ms 232 B

```
{
    "courseId": 3,
    "courseName": "Python Full Stack",
    "courseFees": 27000.0
}
```

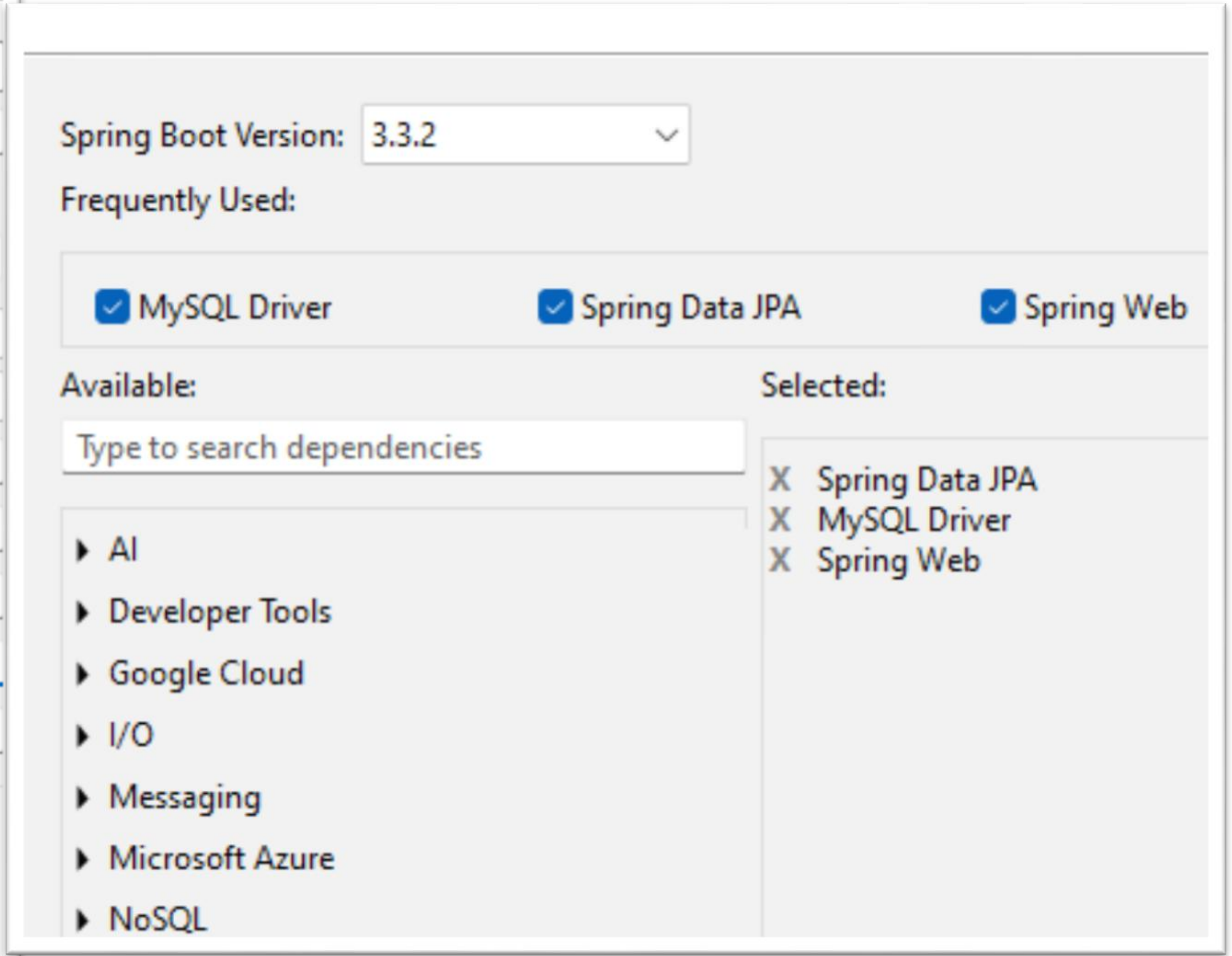
## Student Microservice



| | |
|---|---|
| Service URL | https://start.spring.io |
| Name | student-service |
| ☑ Use default location | |
| Location | E:\Java\MicroServices\Workspace\student-service |
| Type: | Maven |
| Packaging: | Jar |
| Java Version: | 17 |
| Language: | Java |
| Group | com.rit |
| Artifact | student-service |
| Version | 0.0.1-SNAPSHOT |
| Description | Student Service |
| Package | com.rit |

Working sets

☐ Add project to working sets



Spring Boot Version: 3.3.2

Frequently Used:

☑ MySQL Driver ☑ Spring Data JPA ☑ Spring Web

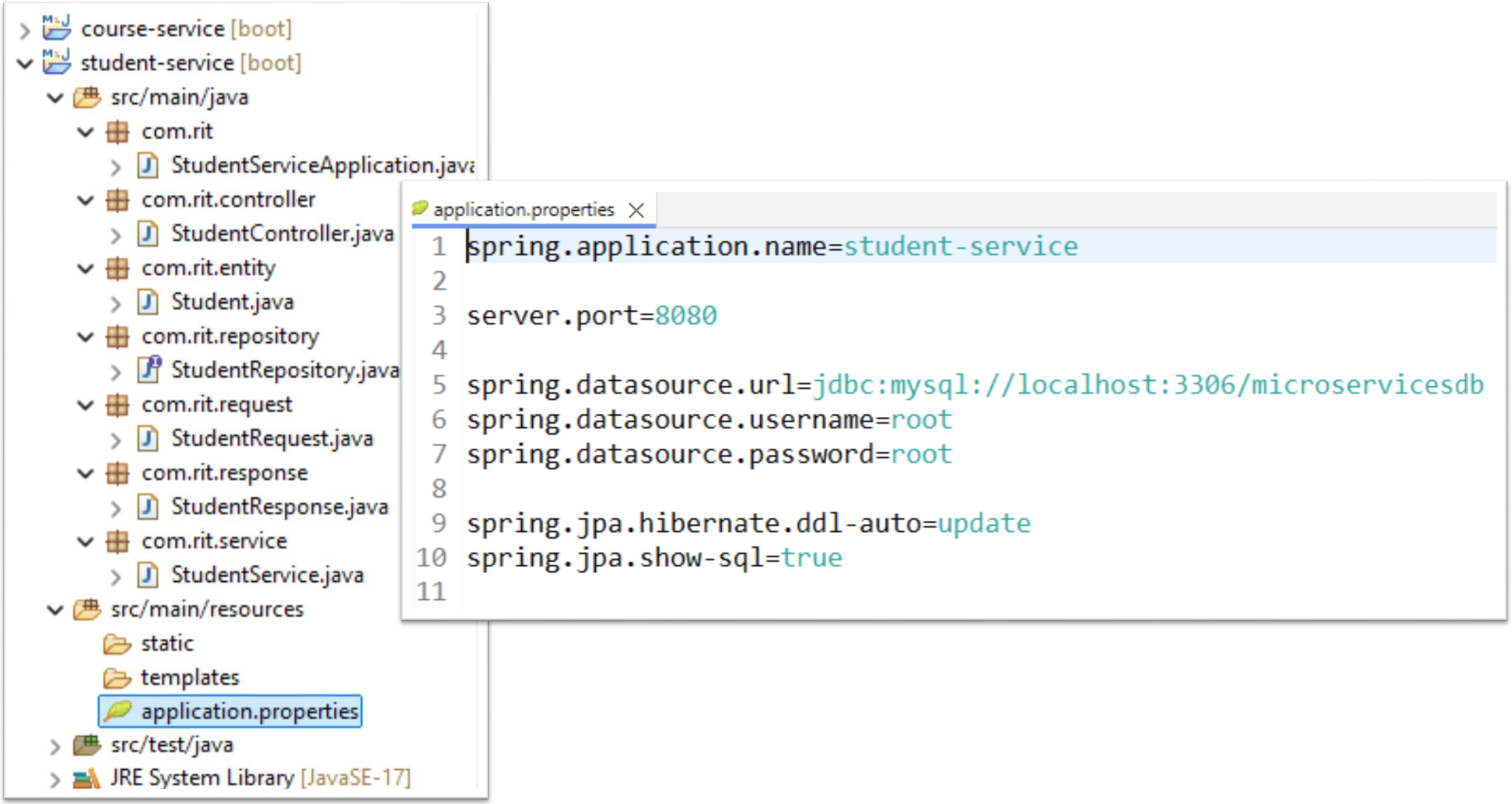
Available:

Type to search dependencies

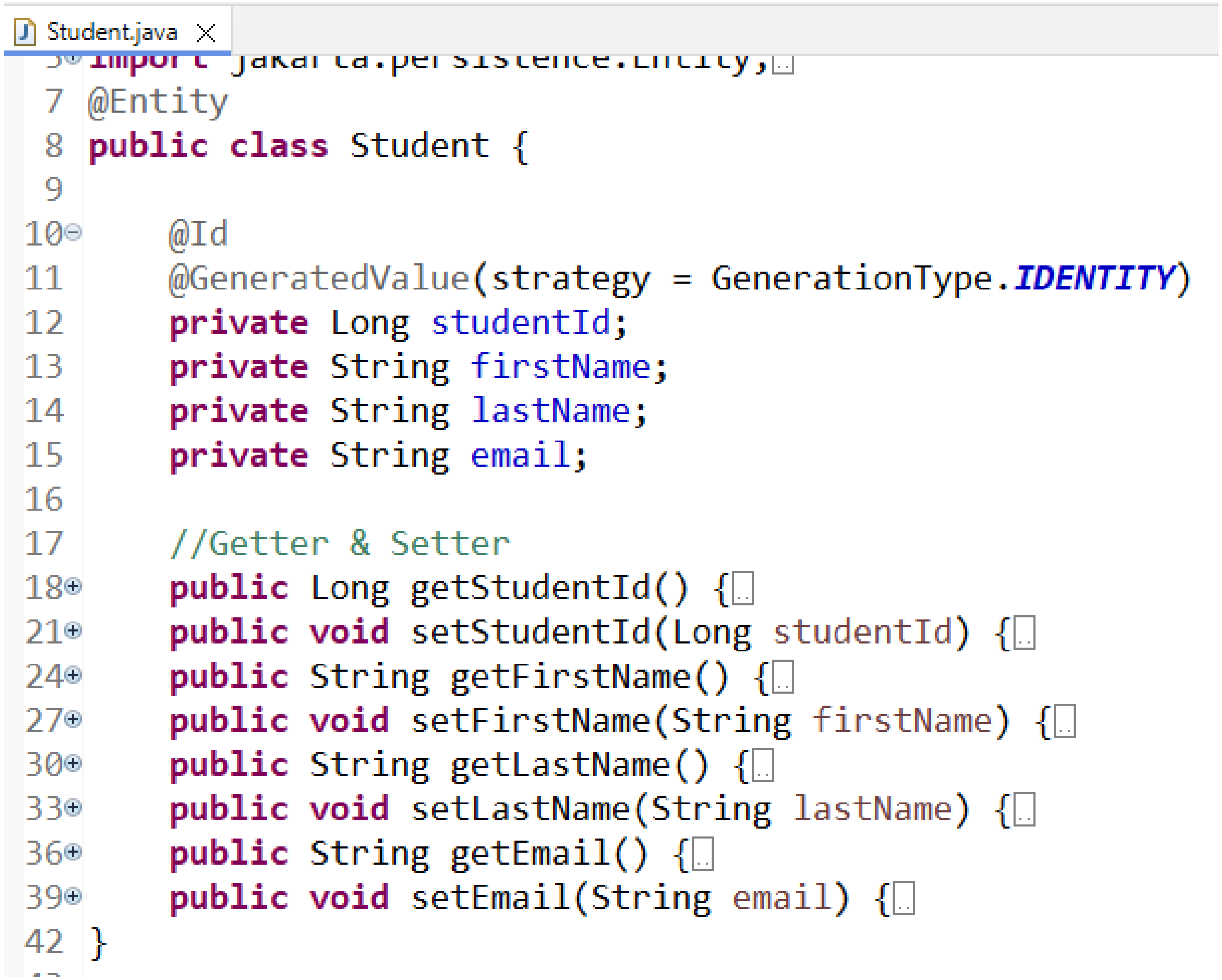
- AI
- Developer Tools
- Google Cloud
- I/O
- Messaging
- Microsoft Azure
- NoSQL

Selected:

X Spring Data JPA
X MySQL Driver
X Spring Web

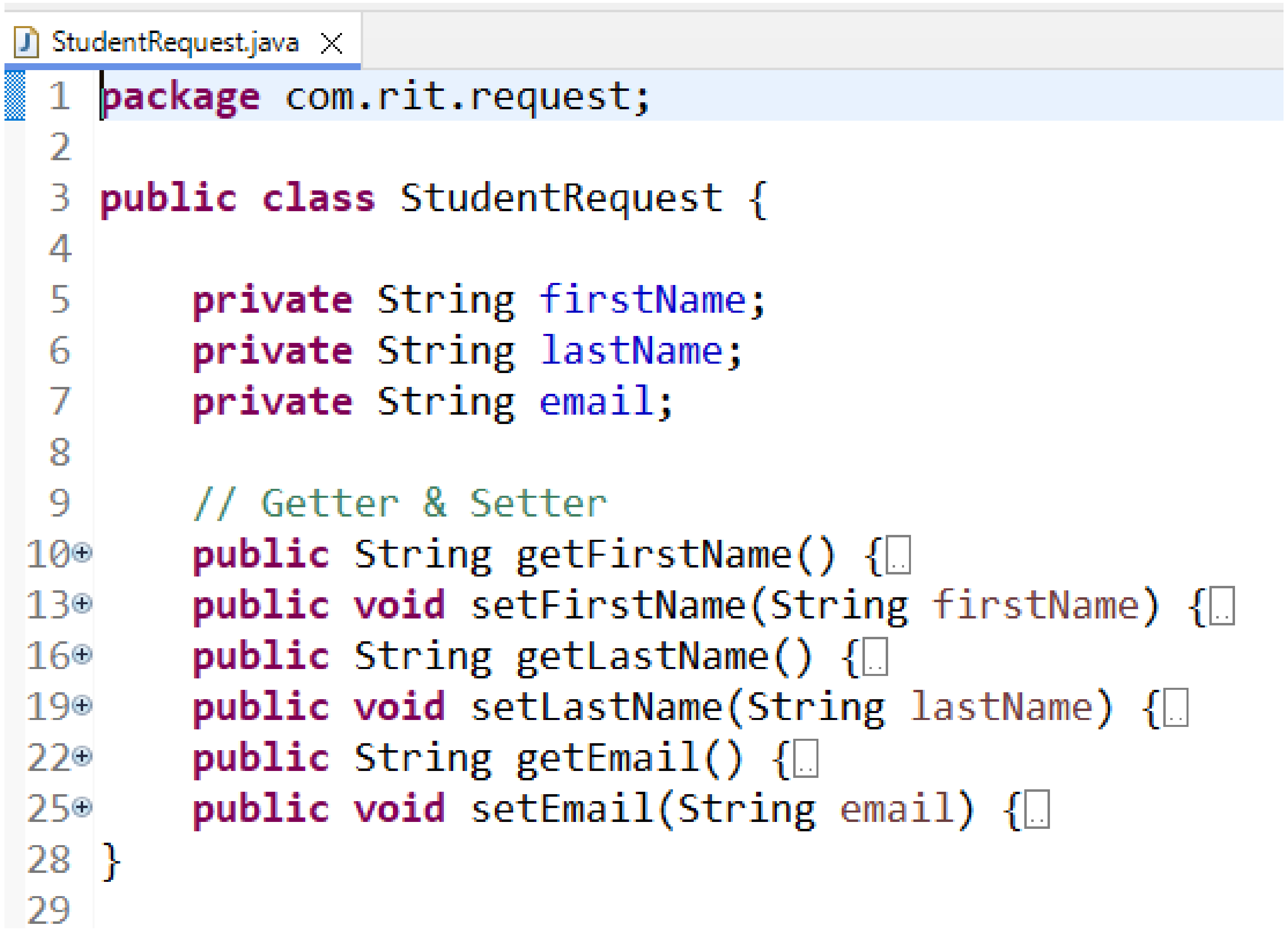


```
spring.application.name=student-service

server.port=8080

spring.datasource.url=jdbc:mysql://localhost:3306/microservicesdb
spring.datasource.username=root
spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

```

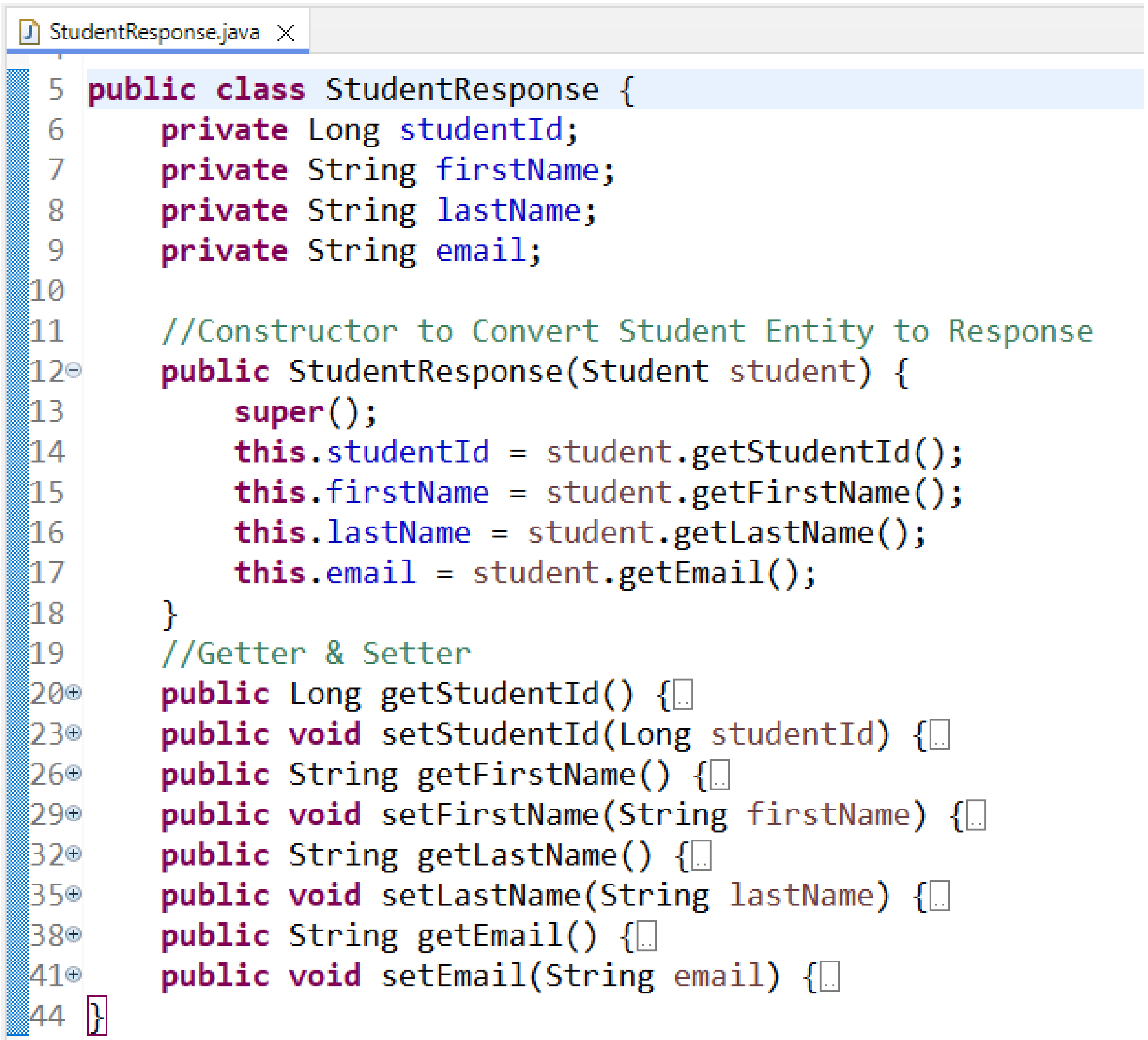


Student.java

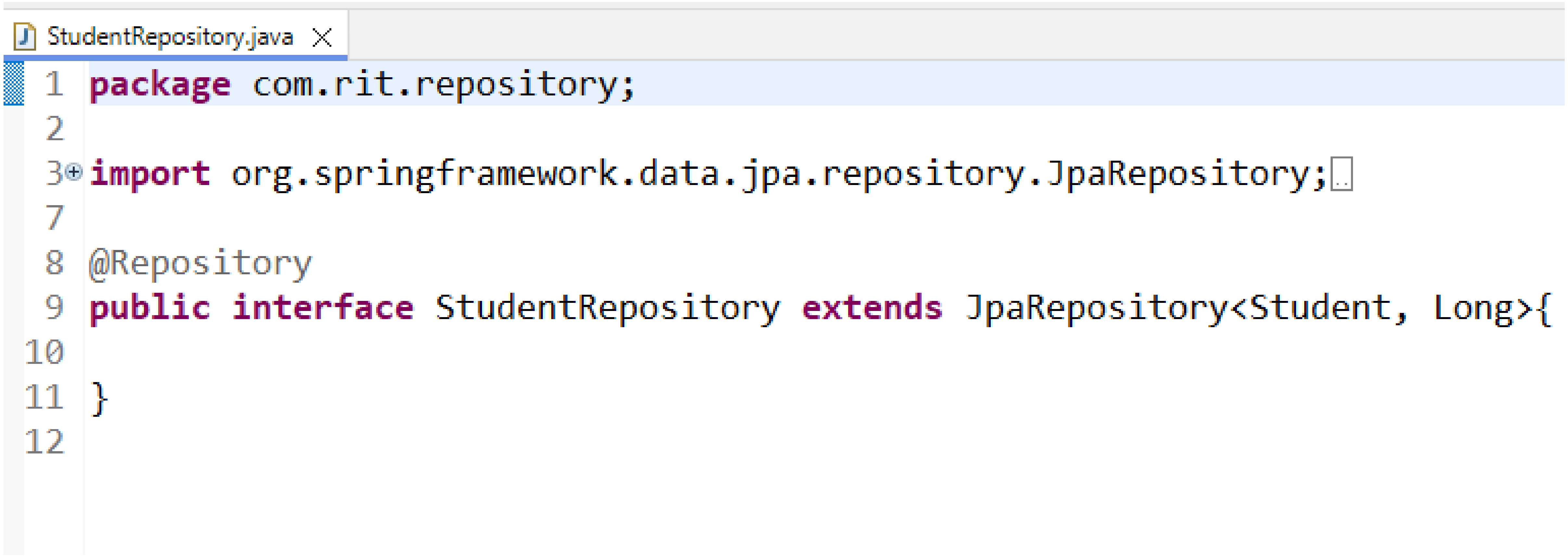
```
import jakarta.persistence.Entity;
@Entity
public class Student {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long studentId;
    private String firstName;
    private String lastName;
    private String email;

    //Getter & Setter
    public Long getStudentId() {
    public void setStudentId(Long studentId) {
    public String getFirstName() {
    public void setFirstName(String firstName) {
    public String getLastName() {
    public void setLastName(String lastName) {
    public String getEmail() {
    public void setEmail(String email) {
}
```



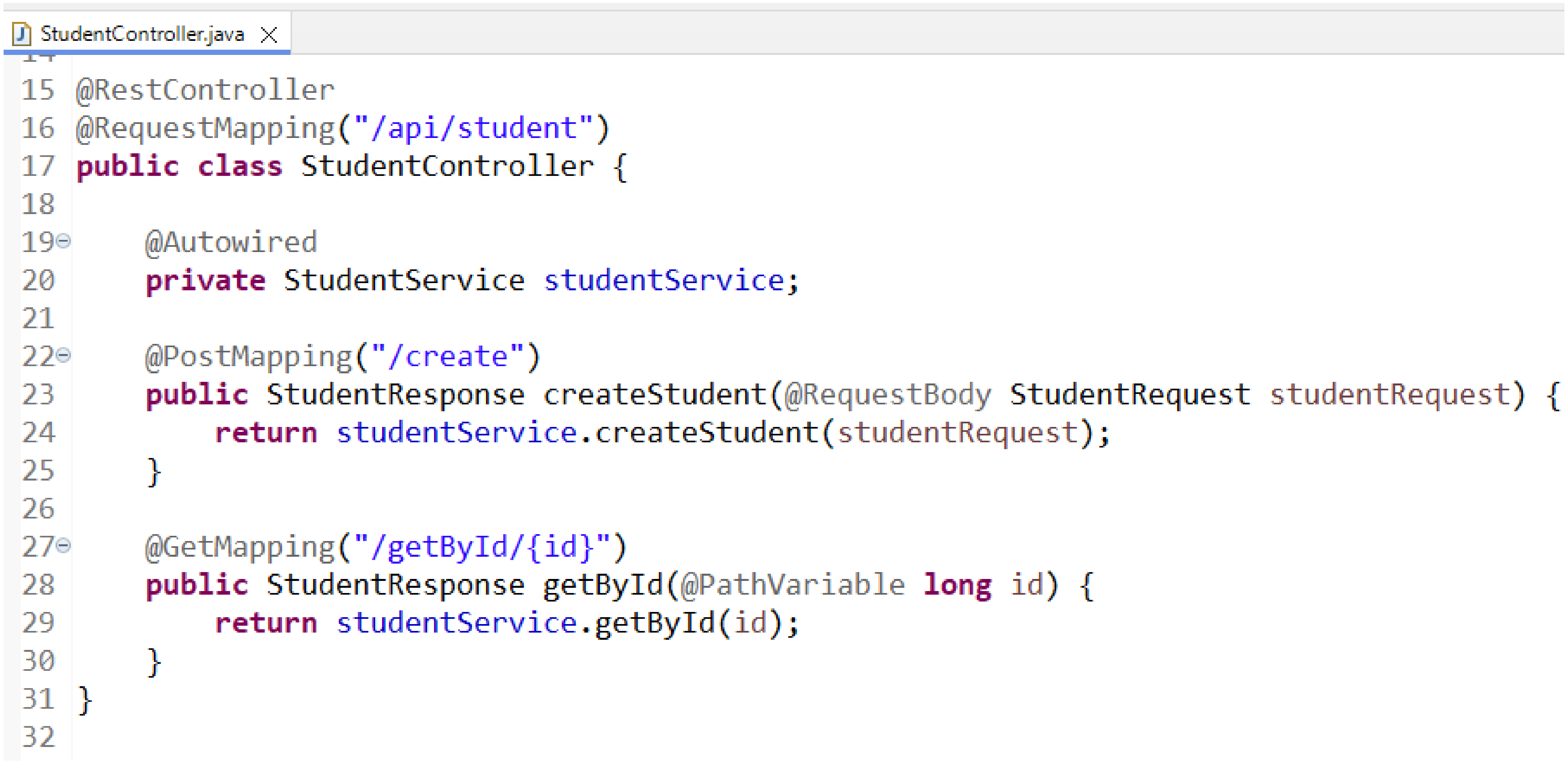
```java
package com.rit.request;

public class StudentRequest {

    private String firstName;
    private String lastName;
    private String email;

    // Getter & Setter
    public String getFirstName() {
    public void setFirstName(String firstName) {
    public String getLastName() {
    public void setLastName(String lastName) {
    public String getEmail() {
    public void setEmail(String email) {
}
```


StudentResponse.java

```java
public class StudentResponse {
    private Long studentId;
    private String firstName;
    private String lastName;
    private String email;

    //Constructor to Convert Student Entity to Response
    public StudentResponse(Student student) {
        super();
        this.studentId = student.getStudentId();
        this.firstName = student.getFirstName();
        this.lastName = student.getLastName();
        this.email = student.getEmail();
    }
    //Getter & Setter
    public Long getStudentId() {
    public void setStudentId(Long studentId) {
    public String getFirstName() {
    public void setFirstName(String firstName) {
    public String getLastName() {
    public void setLastName(String lastName) {
    public String getEmail() {
    public void setEmail(String email) {
}
```

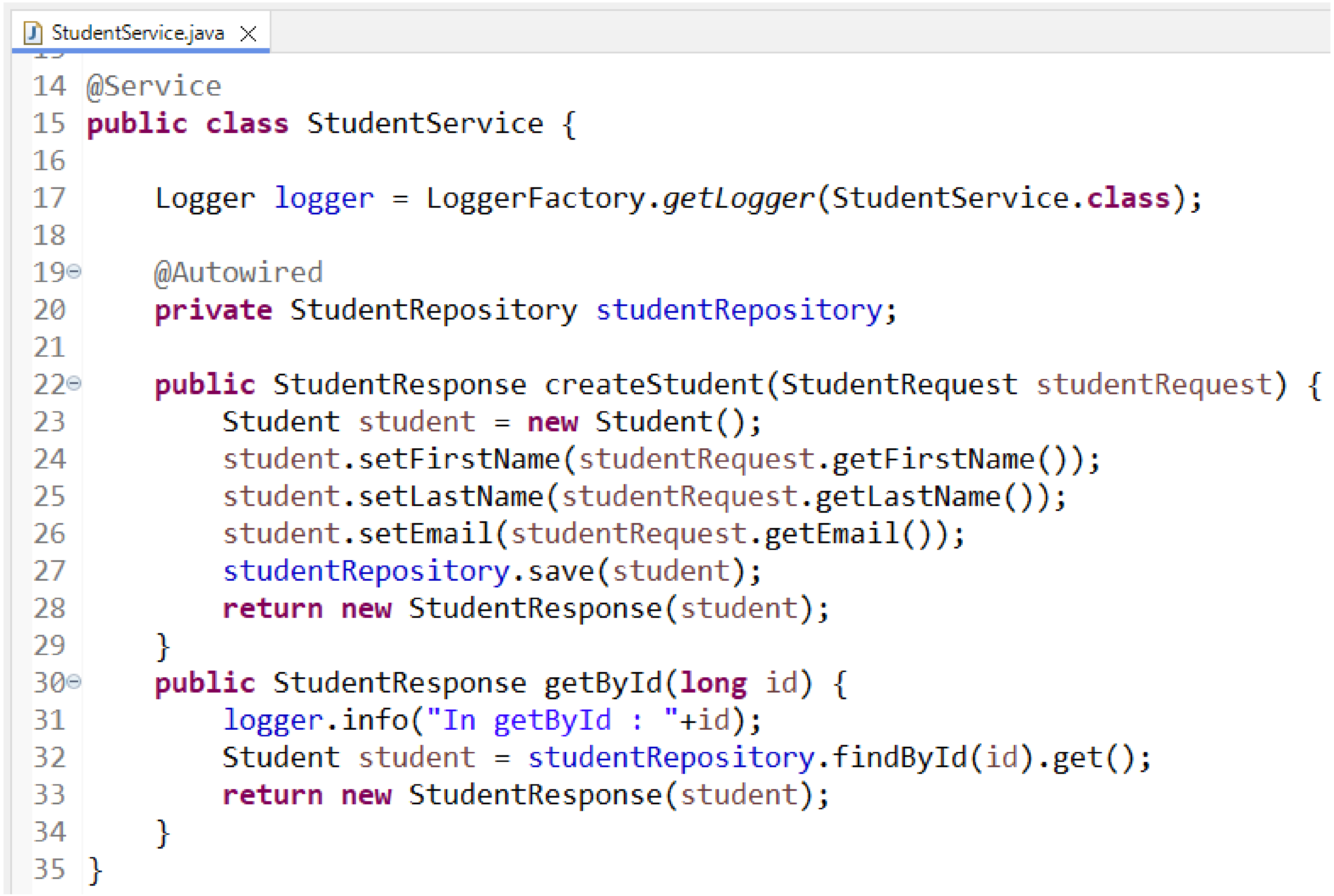

StudentRepository.java

```java
package com.rit.repository;

import org.springframework.data.jpa.repository.JpaRepository;

@Repository
public interface StudentRepository extends JpaRepository<Student, Long>{

}
```

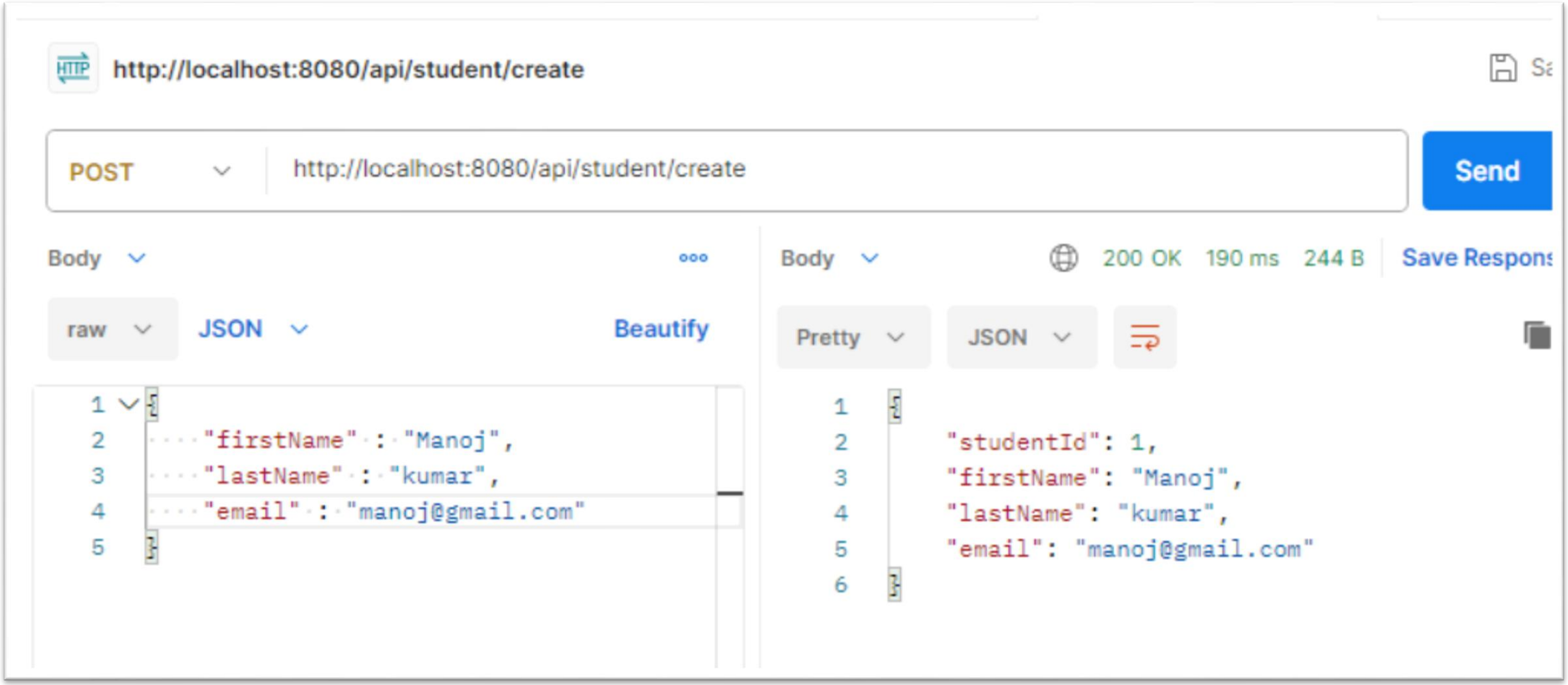


StudentController.java

```java
@RestController
@RequestMapping("/api/student")
public class StudentController {

    @Autowired
    private StudentService studentService;

    @PostMapping("/create")
    public StudentResponse createStudent(@RequestBody StudentRequest studentRequest) {
        return studentService.createStudent(studentRequest);
    }

    @GetMapping("/getById/{id}")
    public StudentResponse getById(@PathVariable long id) {
        return studentService.getById(id);
    }
}

```



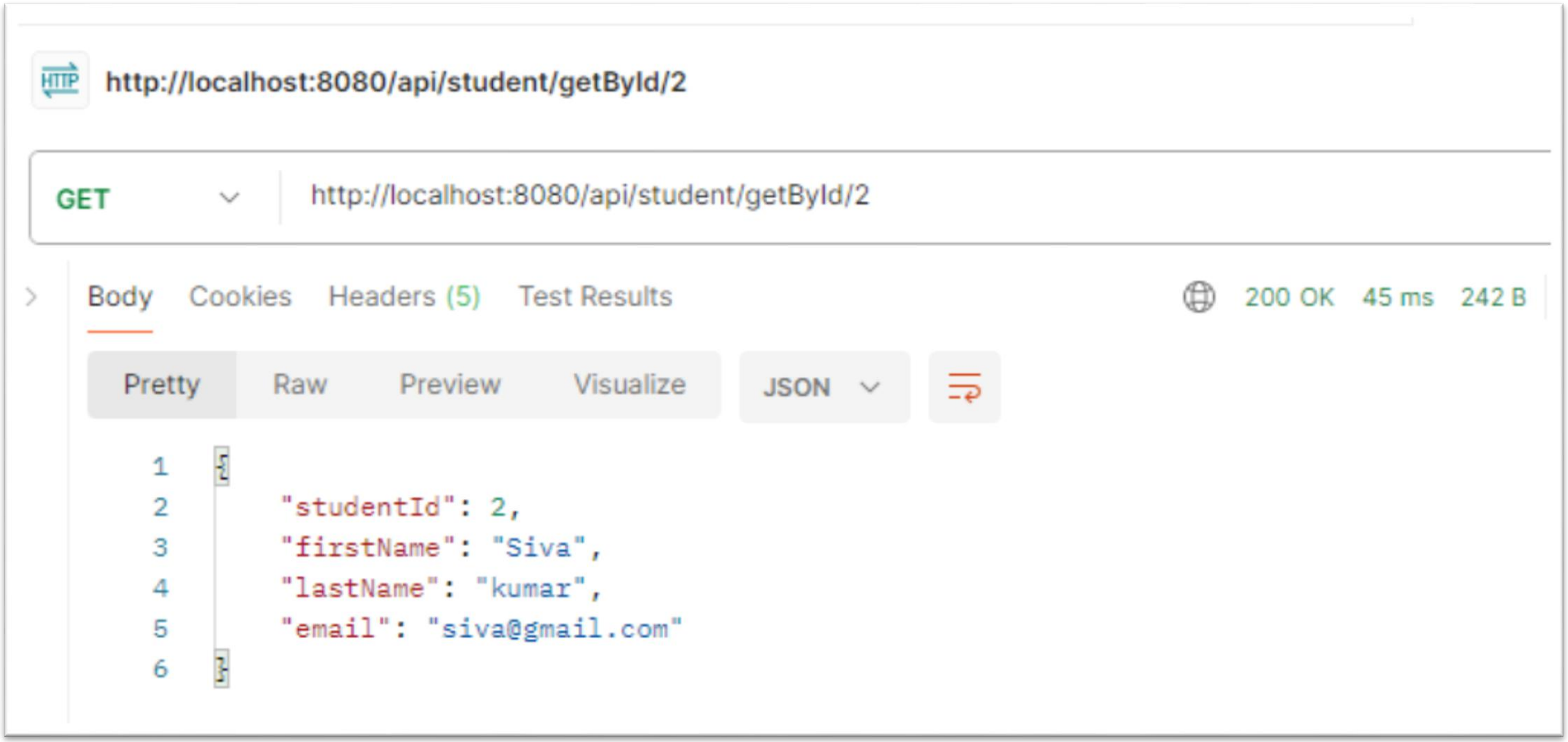
```java
@Service
public class StudentService {

    Logger logger = LoggerFactory.getLogger(StudentService.class);

    @Autowired
    private StudentRepository studentRepository;

    public StudentResponse createStudent(StudentRequest studentRequest) {
        Student student = new Student();
        student.setFirstName(studentRequest.getFirstName());
        student.setLastName(studentRequest.getLastName());
        student.setEmail(studentRequest.getEmail());
        studentRepository.save(student);
        return new StudentResponse(student);
    }
    public StudentResponse getById(long id) {
        logger.info("In getById : "+id);
        Student student = studentRepository.findById(id).get();
        return new StudentResponse(student);
    }
}
```



```
http://localhost:8080/api/student/create
```

```
{
    "firstName" : "Manoj",
    "lastName" : "kumar",
    "email" : "manoj@gmail.com"
}
```

```
{
    "studentId": 1,
    "firstName": "Manoj",
    "lastName": "kumar",
    "email": "manoj@gmail.com"
}
```



http://localhost:8080/api/student/getById/2

GET http://localhost:8080/api/student/getById/2

```
{
    "studentId": 2,
    "firstName": "Siva",
    "lastName": "kumar",
    "email": "siva@gmail.com"
}
```

## Get Course along with Student

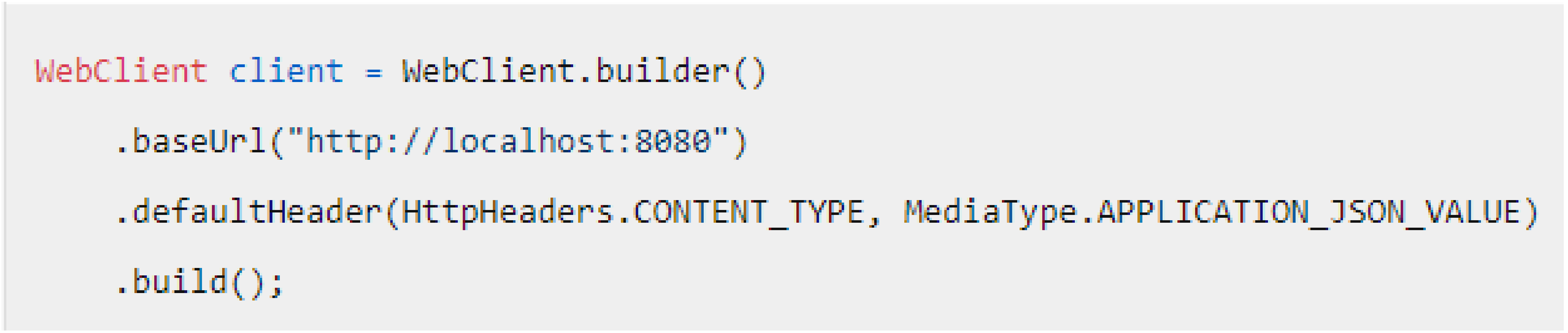
Using WebClient

## WebClient

WebClient is a reactive, non-blocking web client introduced in Spring 5 as part of the Spring WebFlux module.

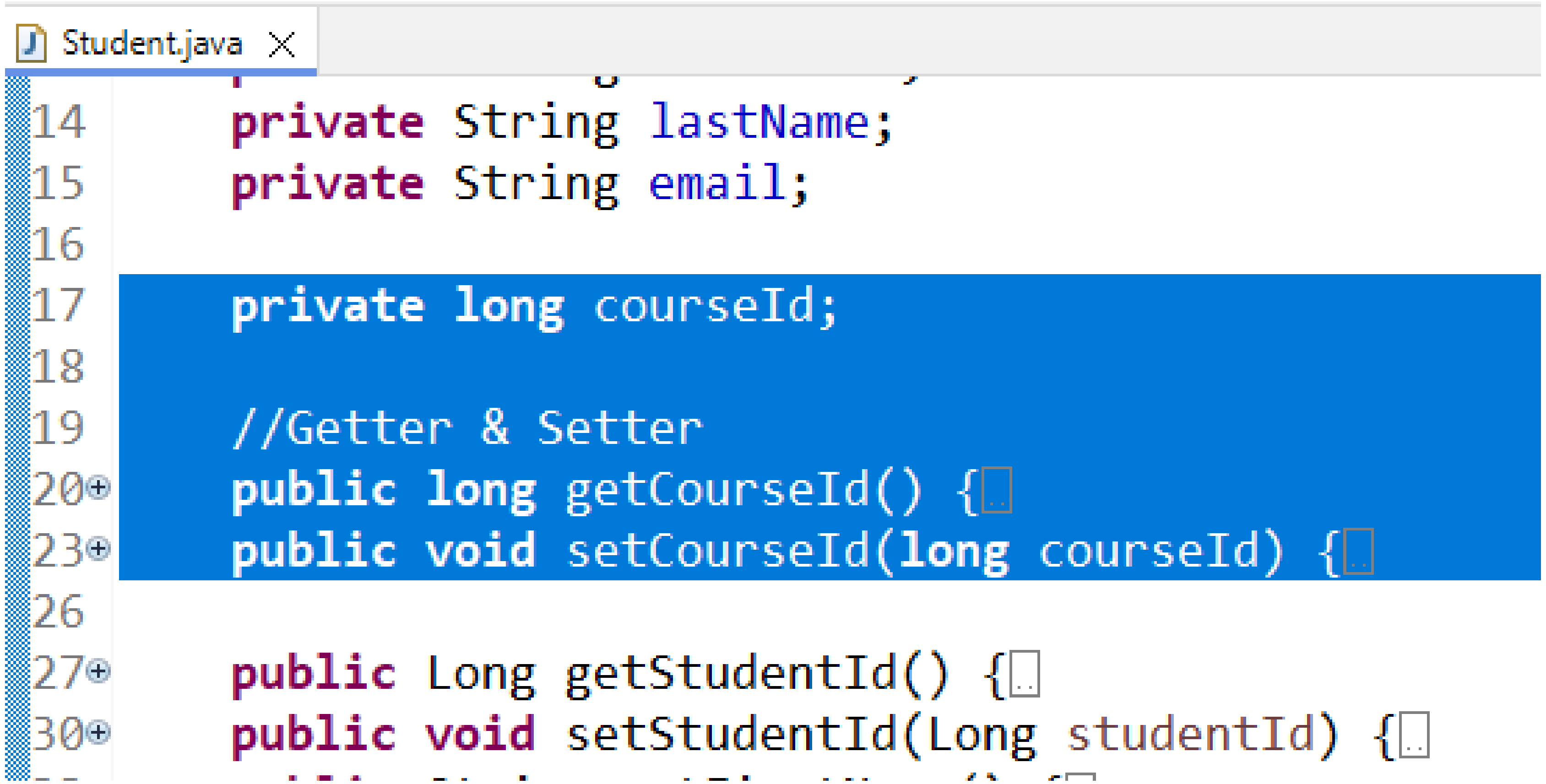
It is designed to replace the classic RestTemplate for making HTTP requests in a more modern and efficient way.

WebClient also supports synchronous and Asynchronous operations, making it versatile for different use cases



```java
WebClient client = WebClient.builder()

    .baseUrl("http://localhost:8080")

    .defaultHeader(HttpHeaders.CONTENT_TYPE, MediaType.APPLICATION_JSON_VALUE)

    .build();
```

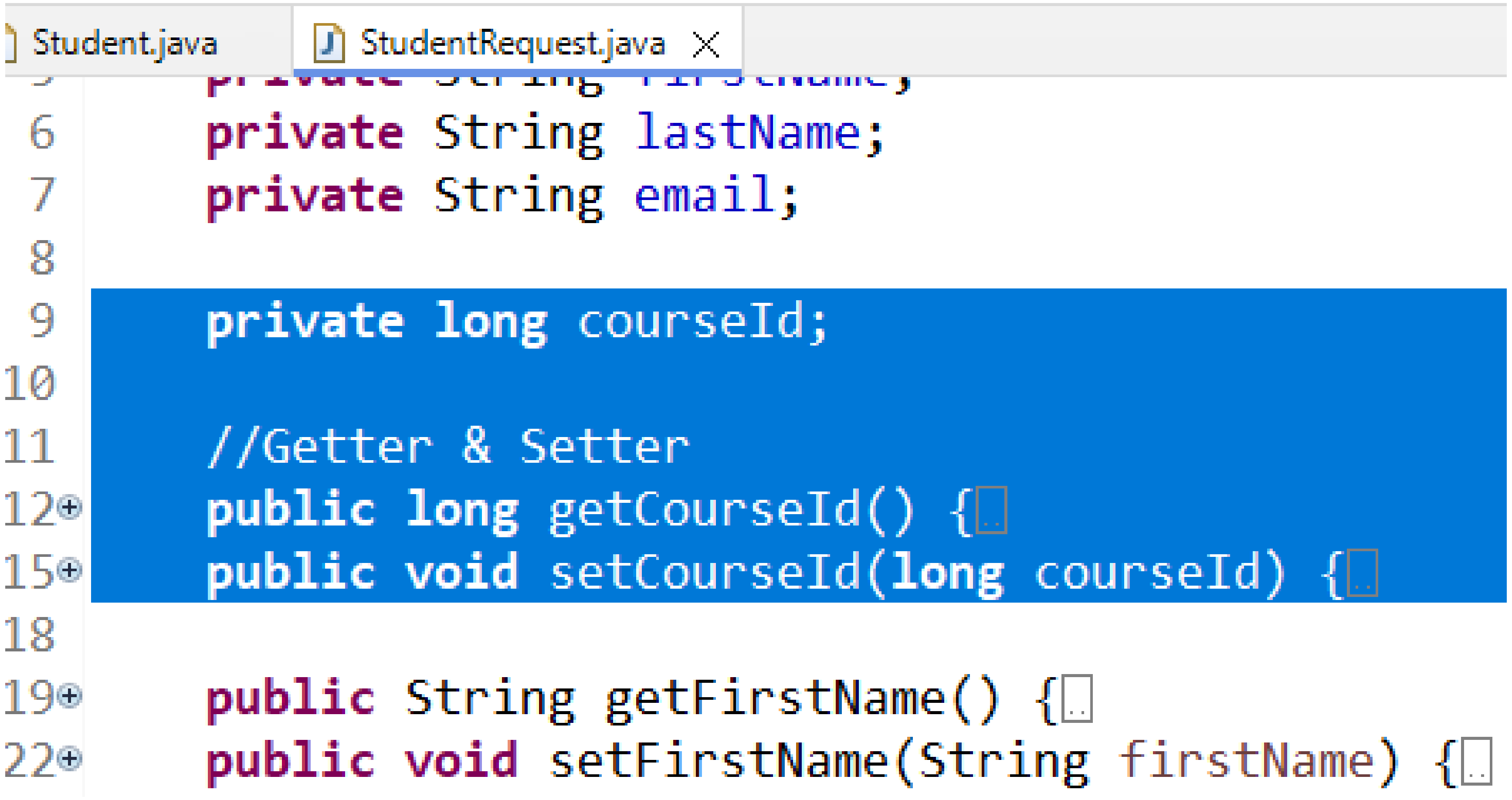
# Student: Add CourseId



Student.java

```
    private String lastName;
    private String email;

    private long courseId;

    //Getter & Setter
    public long getCourseId() {...
    public void setCourseId(long courseId) {...

    public Long getStudentId() {...
    public void setStudentId(Long studentId) {...
```

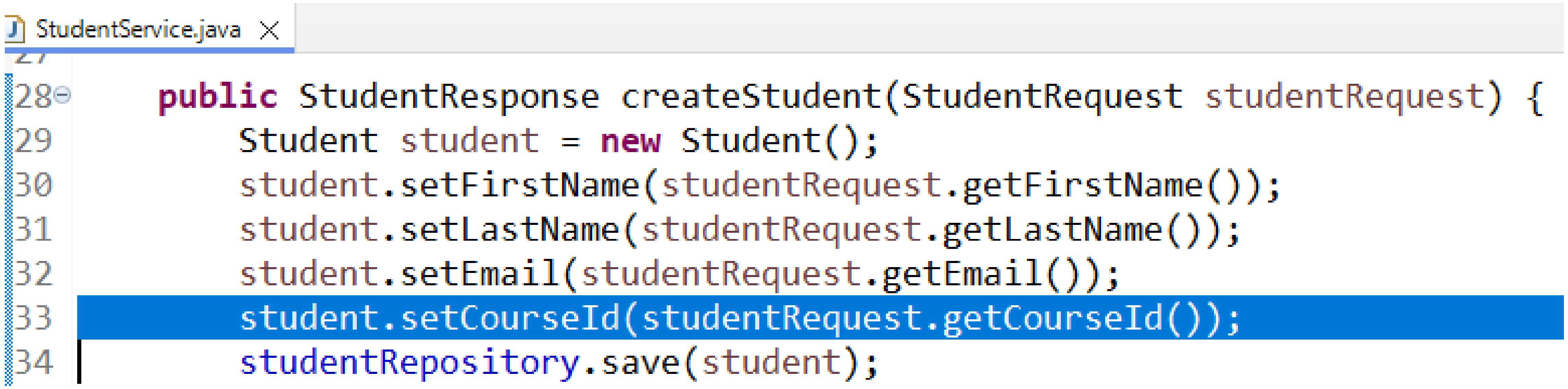
# StudentRequest: Add CourseId



Student.java | StudentRequest.java

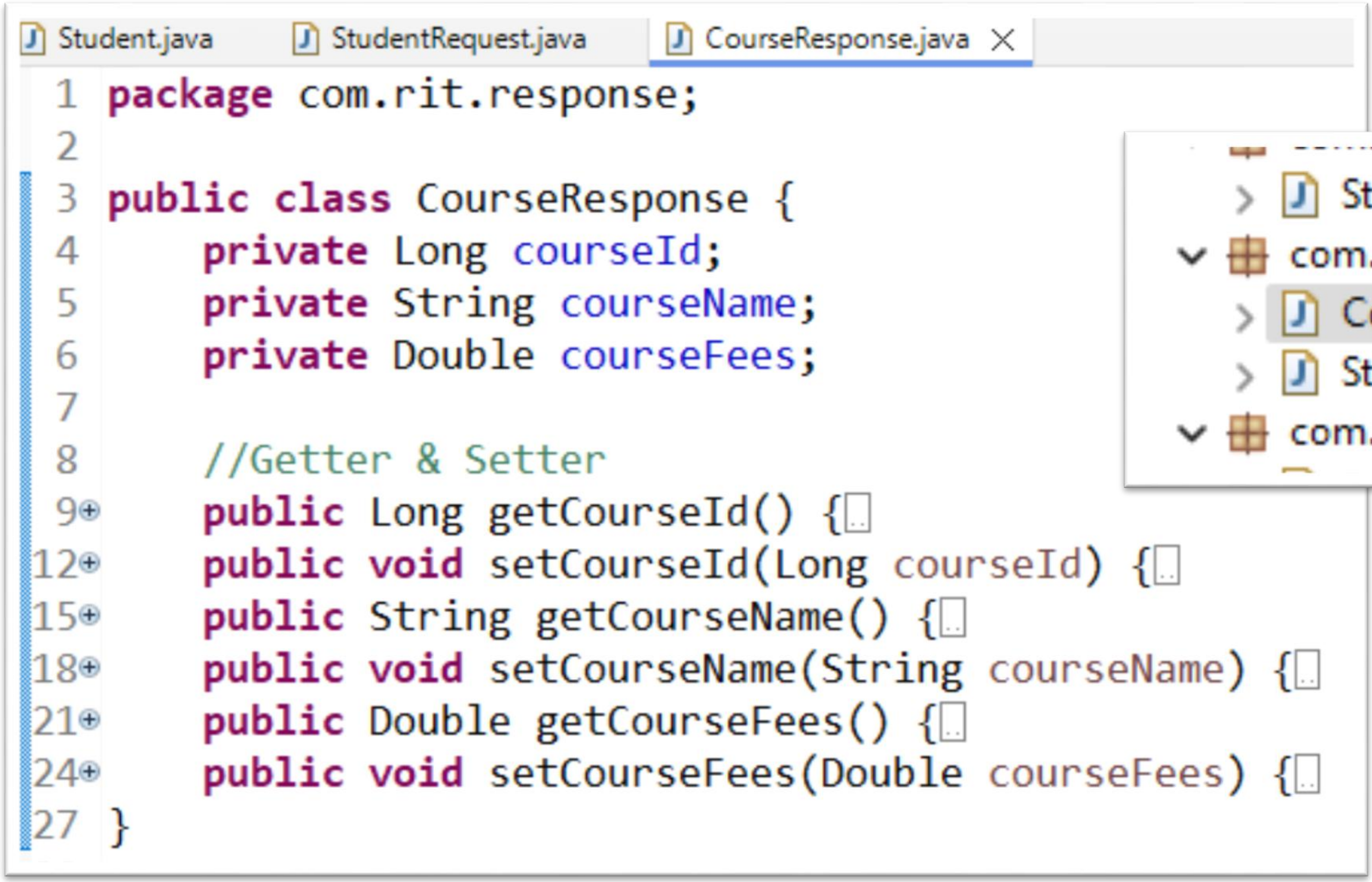
```
    private String lastName;
    private String email;

    private long courseId;

    //Getter & Setter
    public long getCourseId() {
    public void setCourseId(long courseId) {

    public String getFirstName() {
    public void setFirstName(String firstName) {
```

## Update it in Service class

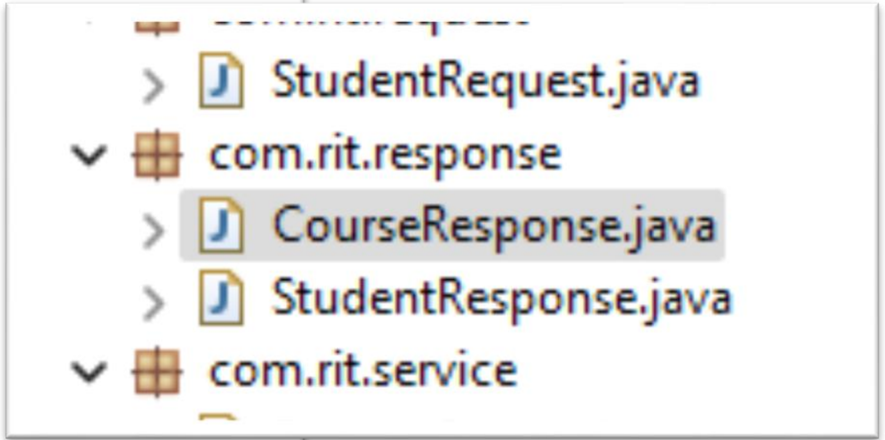

StudentService.java

```
    public StudentResponse createStudent(StudentRequest studentRequest) {
        Student student = new Student();
        student.setFirstName(studentRequest.getFirstName());
        student.setLastName(studentRequest.getLastName());
        student.setEmail(studentRequest.getEmail());
        student.setCourseId(studentRequest.getCourseId());
        studentRepository.save(student);
```

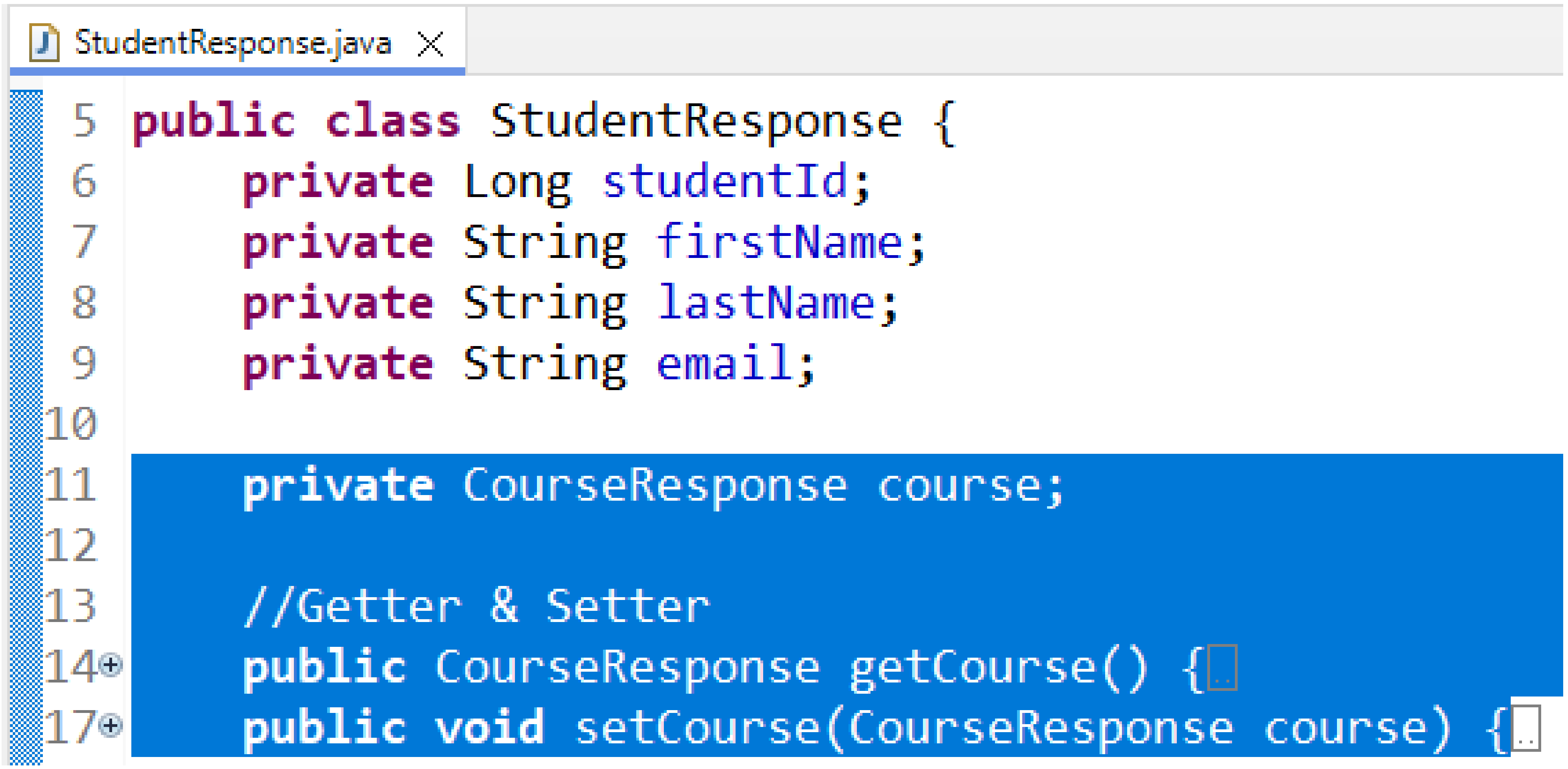
## Create a CourseResponse Class in response



```
package com.rit.response;

public class CourseResponse {
    private Long courseId;
    private String courseName;
    private Double courseFees;

    //Getter & Setter
    public Long getCourseId() {
    public void setCourseId(Long courseId) {
    public String getCourseName() {
    public void setCourseName(String courseName) {
    public Double getCourseFees() {
    public void setCourseFees(Double courseFees) {
}
```



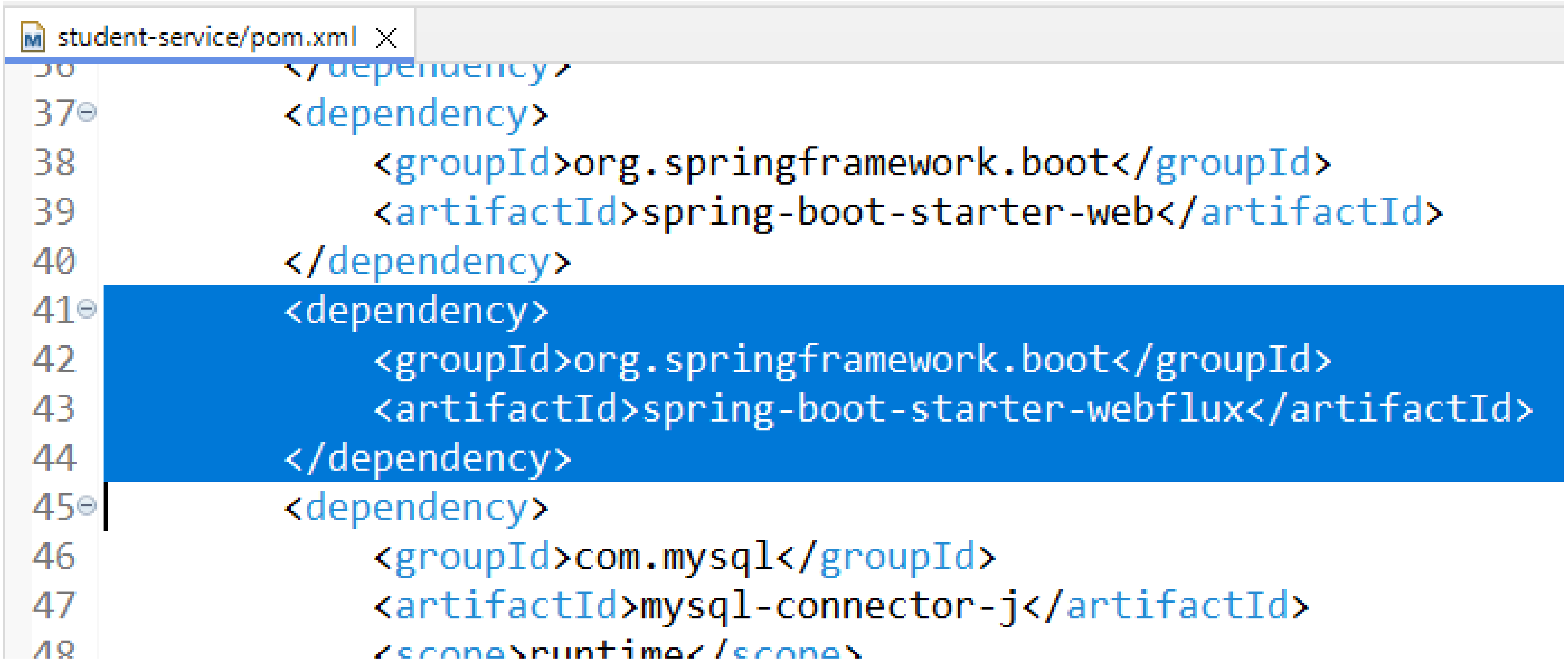
# Add CourseResponse in StudentResponse



StudentResponse.java

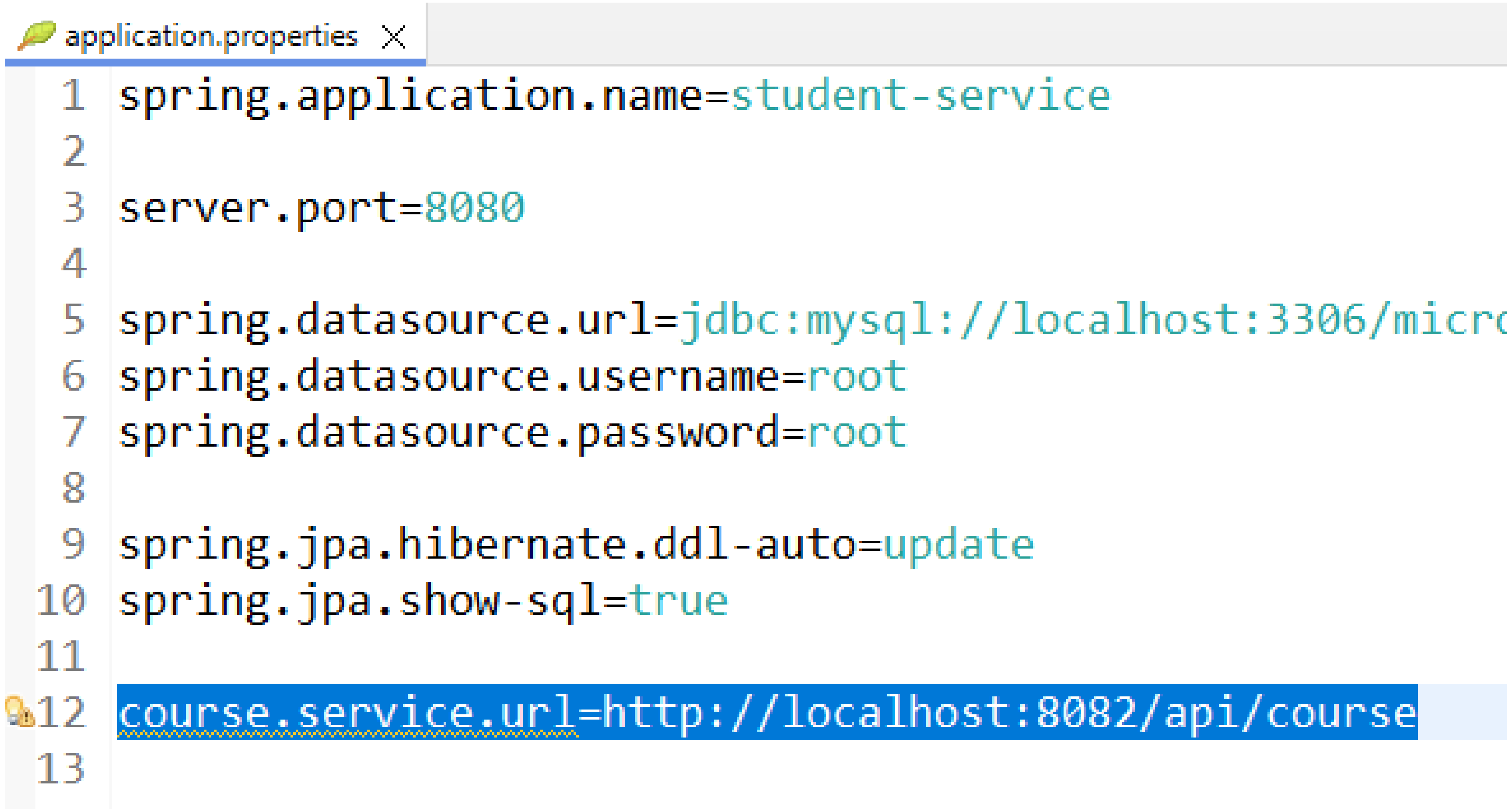
```
public class StudentResponse {
    private Long studentId;
    private String firstName;
    private String lastName;
    private String email;

    private CourseResponse course;

    //Getter & Setter
    public CourseResponse getCourse() {
    public void setCourse(CourseResponse course) {
```

# Add Webflux dependency (for WebClient)



```
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-webflux</artifactId>
        </dependency>
        <dependency>
            <groupId>com.mysql</groupId>
            <artifactId>mysql-connector-j</artifactId>
            <scope>runtime</scope>
```

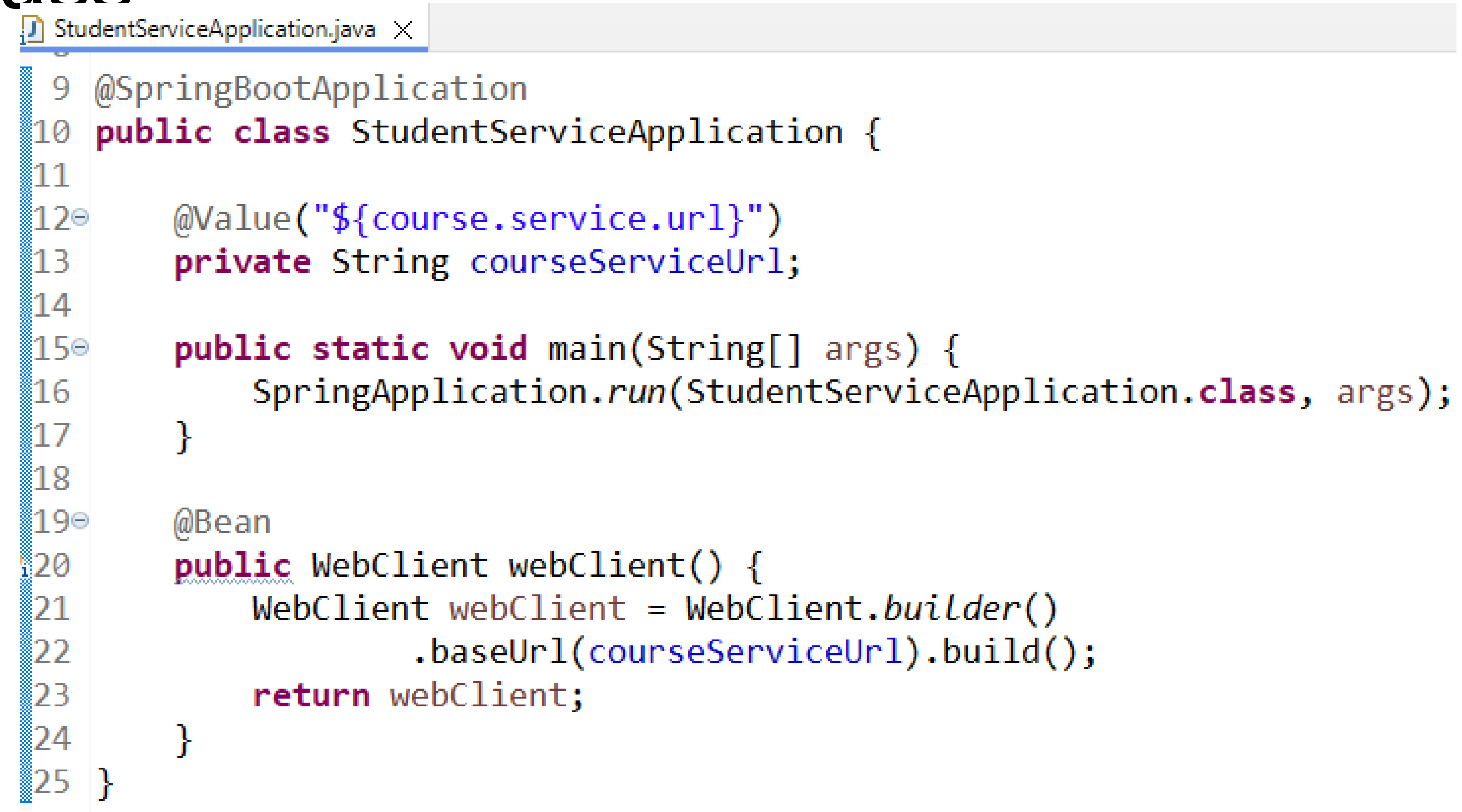
# Add course API URL in properties file



application.properties

```
spring.application.name=student-service

server.port=8080

spring.datasource.url=jdbc:mysql://localhost:3306/micr
spring.datasource.username=root
spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

course.service.url=http://localhost:8082/api/course

```

## Add url and WebClient Bean in Main class



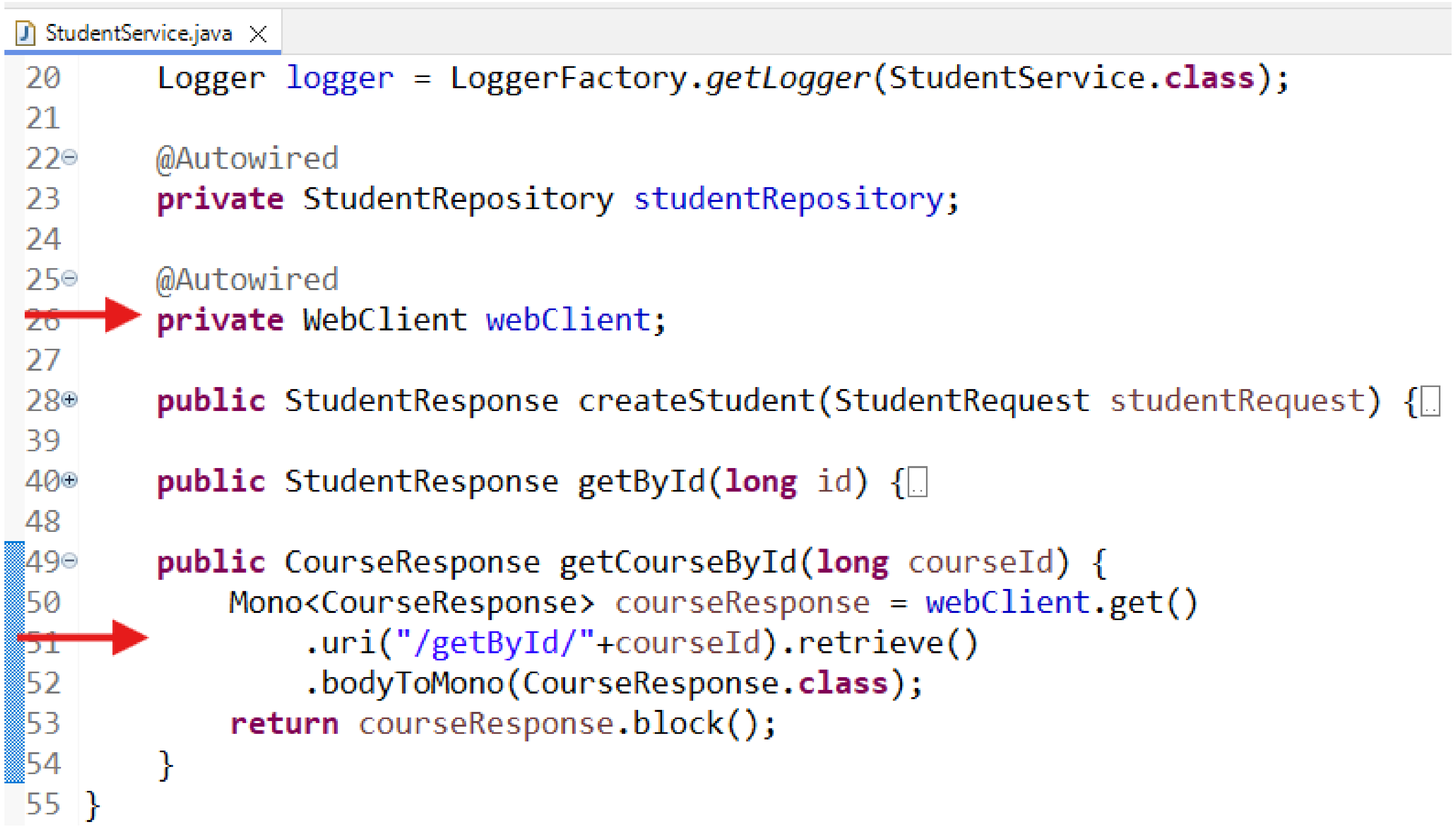
StudentServiceApplication.java

```
@SpringBootApplication
public class StudentServiceApplication {

    @Value("${course.service.url}")
    private String courseServiceUrl;

    public static void main(String[] args) {
        SpringApplication.run(StudentServiceApplication.class, args);
    }

    @Bean
    public WebClient webClient() {
        WebClient webClient = WebClient.builder()
                .baseUrl(courseServiceUrl).build();
        return webClient;
    }
}
```

## In Student Service

- Autowire WebClient
- Create a method to get CourseResponse
- Update StudentReponse in getById method
- Update StudentReponse in createStudent method

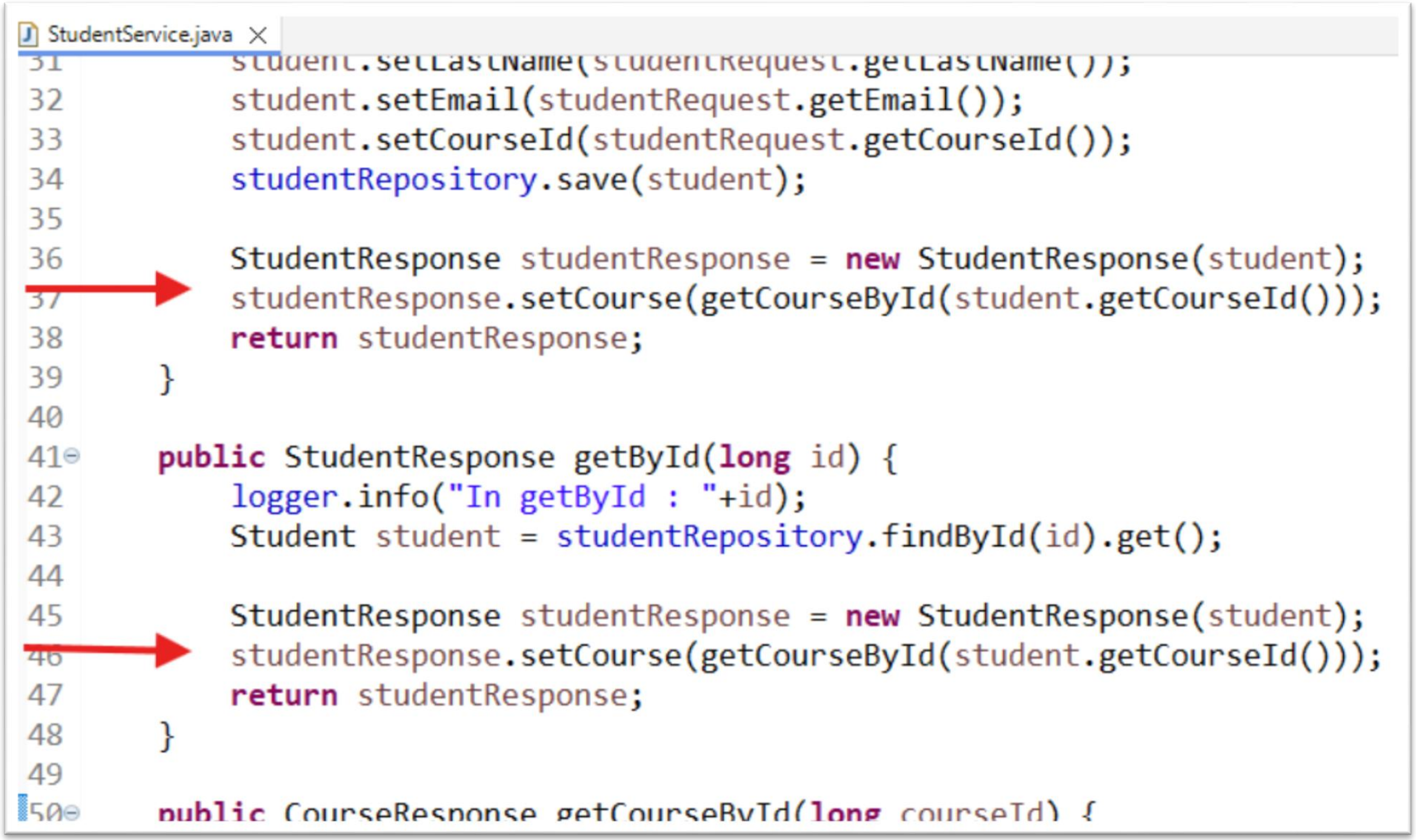
# Add WebClient and CourseReponse Method



StudentService.java

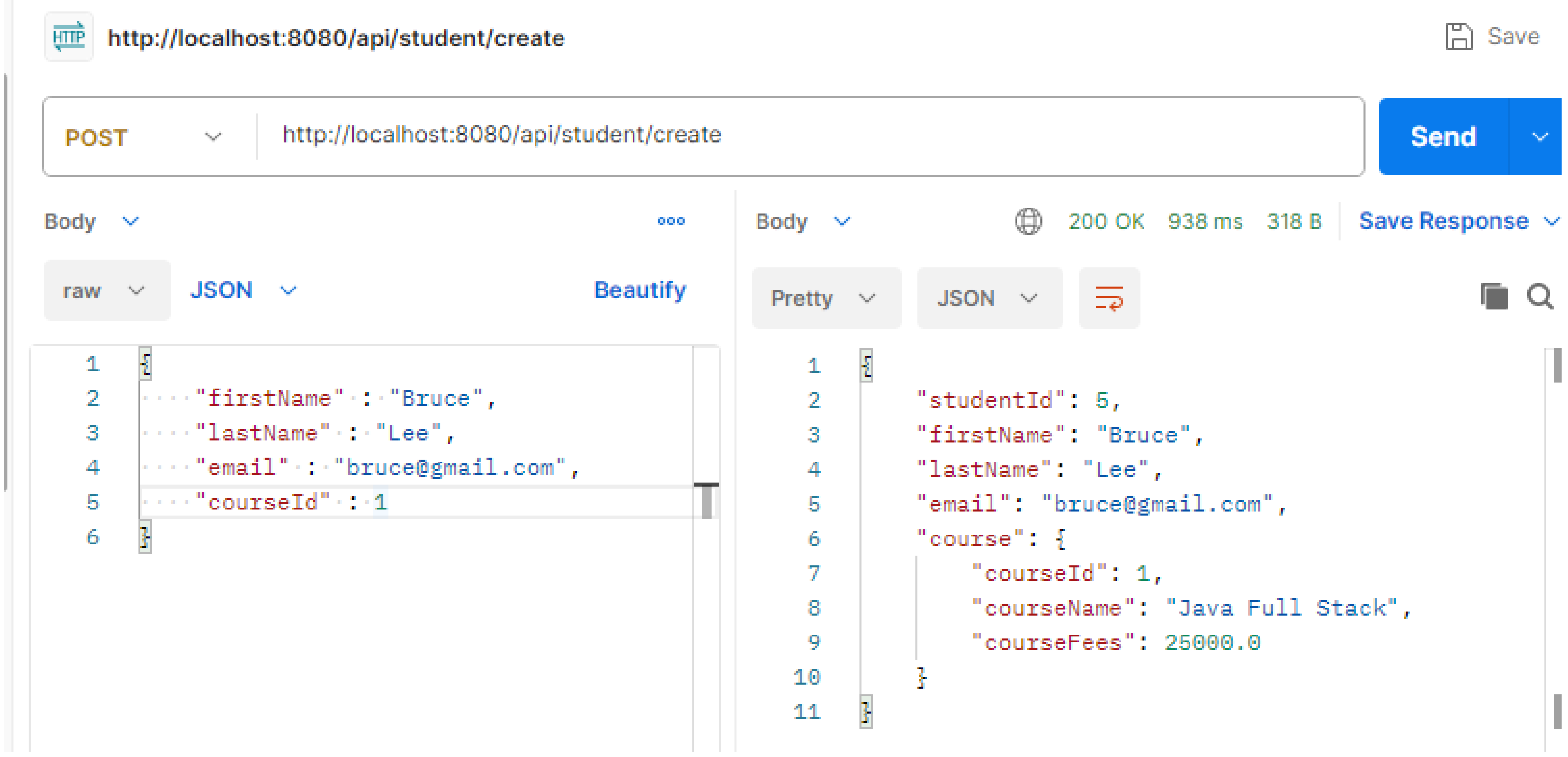
```
    Logger logger = LoggerFactory.getLogger(StudentService.class);

    @Autowired
    private StudentRepository studentRepository;

    @Autowired
    private WebClient webClient;

    public StudentResponse createStudent(StudentRequest studentRequest) {

    public StudentResponse getById(long id) {

    public CourseResponse getCourseById(long courseId) {
        Mono<CourseResponse> courseResponse = webClient.get()
            .uri("/getById/"+courseId).retrieve()
            .bodyToMono(CourseResponse.class);
        return courseResponse.block();
    }
}
```

# Update in create & getById methods



StudentService.java

```java
        student.setLastName(studentRequest.getLastName());
        student.setEmail(studentRequest.getEmail());
        student.setCourseId(studentRequest.getCourseId());
        studentRepository.save(student);

        StudentResponse studentResponse = new StudentResponse(student);
        studentResponse.setCourse(getCourseById(student.getCourseId()));
        return studentResponse;
    }

    public StudentResponse getById(long id) {
        logger.info("In getById : "+id);
        Student student = studentRepository.findById(id).get();

        StudentResponse studentResponse = new StudentResponse(student);
        studentResponse.setCourse(getCourseById(student.getCourseId()));
        return studentResponse;
    }

    public CourseResponse getCourseById(long courseId) {
```



http://localhost:8080/api/student/create

POST http://localhost:8080/api/student/create

```
{
    "firstName" : "Bruce",
    "lastName" : "Lee",
    "email" : "bruce@gmail.com",
    "courseId" : 1
}
```

200 OK 938 ms 318 B

```
{
    "studentId": 5,
    "firstName": "Bruce",
    "lastName": "Lee",
    "email": "bruce@gmail.com",
    "course": {
        "courseId": 1,
        "courseName": "Java Full Stack",
        "courseFees": 25000.0
    }
}
```