

Student ID : 2200032247

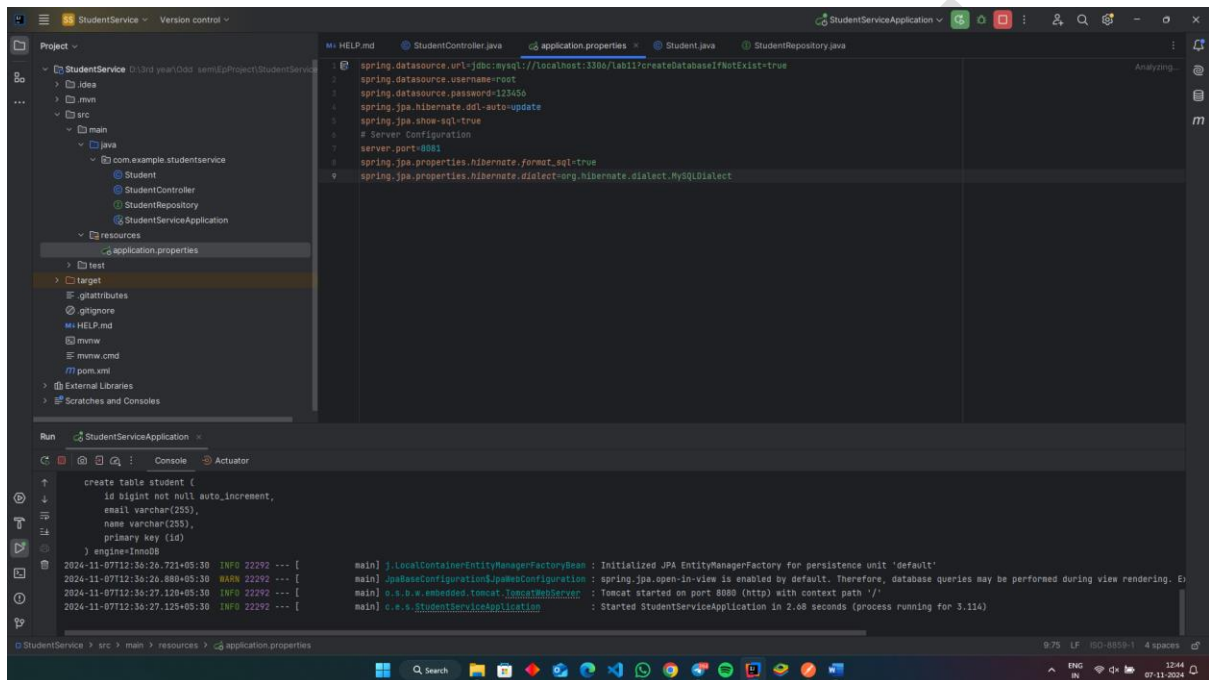
Name : N Pranay Vardhan

Lab experiment number : 11

Section number : 36

## StudentService

### 1.Application.properties



The screenshot shows an IDE with the following components:

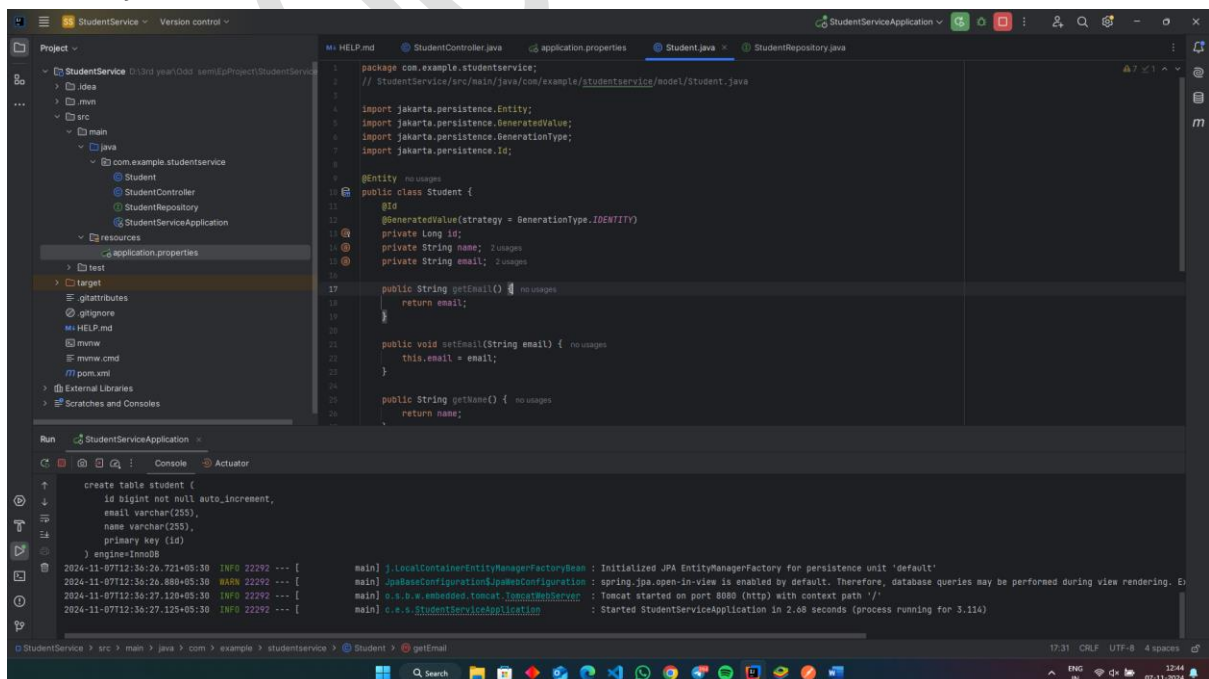
- Project Explorer:** Shows the project structure with folders for `src`, `main`, `resources`, `test`, and `target`. The `resources` folder contains `application.properties`.
- Editor:** Displays the `application.properties` file with the following content:

```
spring.datasource.url=jdbc:mysql://localhost:3306/lab11?createDatabaseIfNotExist=true
spring.datasource.username=root
spring.datasource.password=123456
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
# Server Configuration
server.port=8080
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```
- Run Console:** Shows the output of the application. The first part shows the creation of the `student` table:

```
create table student (
  id bigint not null auto_increment,
  email varchar(255),
  name varchar(255),
  primary key (id)
) engine=InnoDB
```

Subsequent lines show log messages from the application, including the initialization of the JPA EntityManagerFactory and the start of the Tomcat web server on port 8080.

### Student.java



The screenshot shows an IDE with the following components:

- Project Explorer:** Shows the project structure with folders for `src`, `main`, `resources`, `test`, and `target`. The `src/main/java/com/example/studentService/model` folder contains `Student.java`.
- Editor:** Displays the `Student.java` file with the following content:

```
package com.example.studentService;
// StudentService/src/main/java/com/example/studentService/model/Student.java

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    private String email;

    public String getEmail() {
        return email;
    }

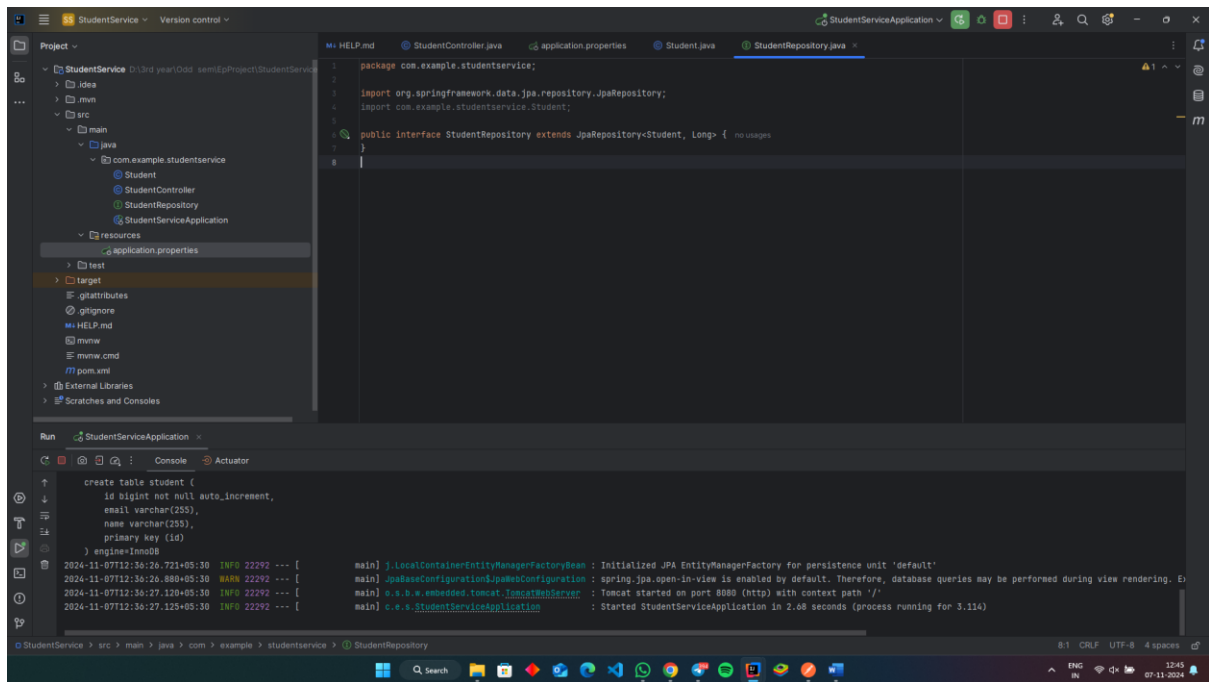
    public void setEmail(String email) {
        this.email = email;
    }

    public String getName() {
        return name;
    }
}
```
- Run Console:** Shows the output of the application. The first part shows the creation of the `student` table:

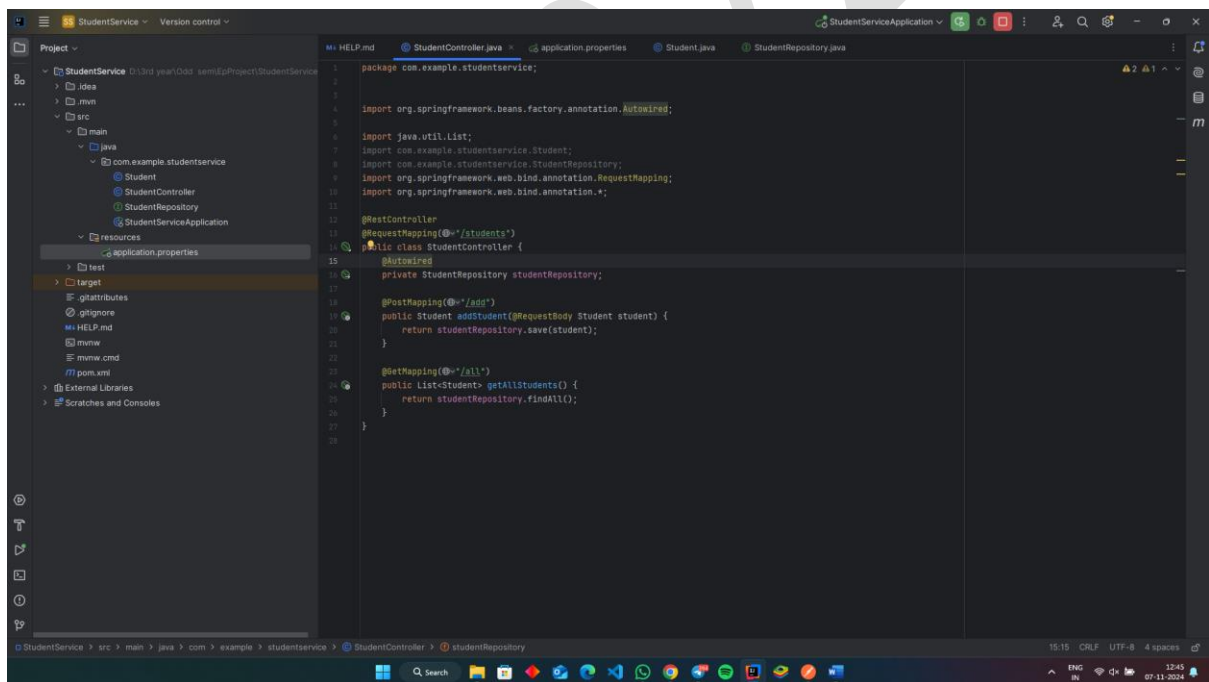
```
create table student (
  id bigint not null auto_increment,
  email varchar(255),
  name varchar(255),
  primary key (id)
) engine=InnoDB
```

Subsequent lines show log messages from the application, including the initialization of the JPA EntityManagerFactory and the start of the Tomcat web server on port 8080.

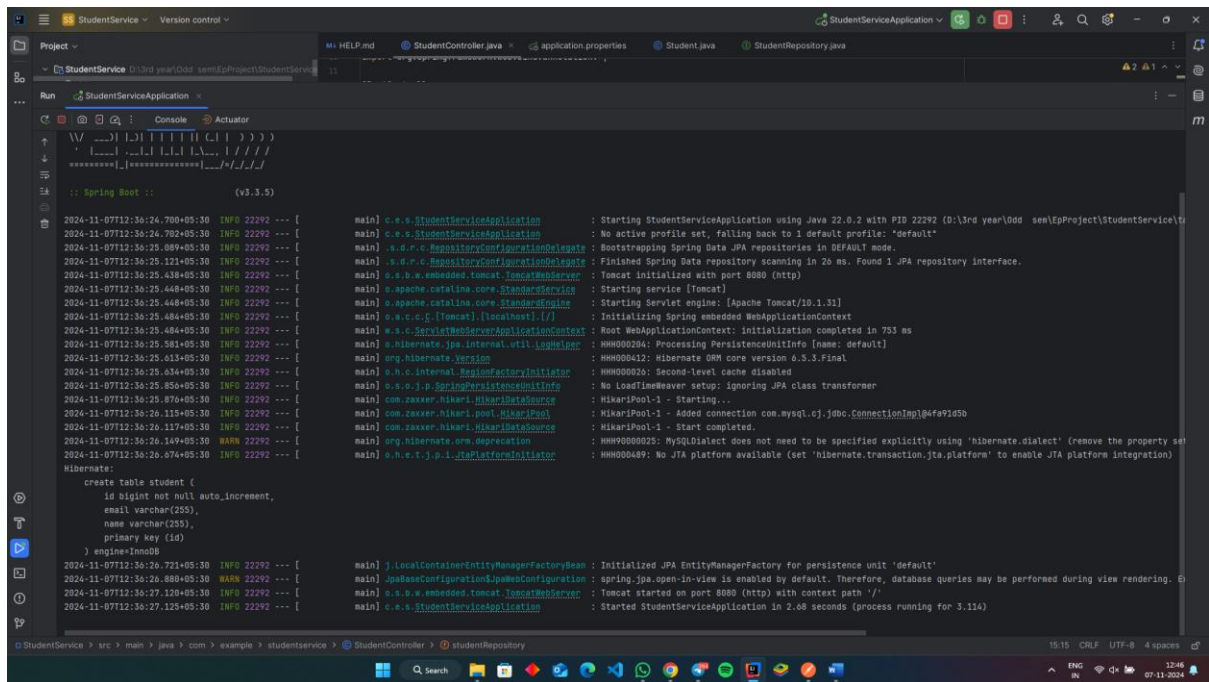
### 3.StudentRepository



### 4.StudentController



## 5.Terminal



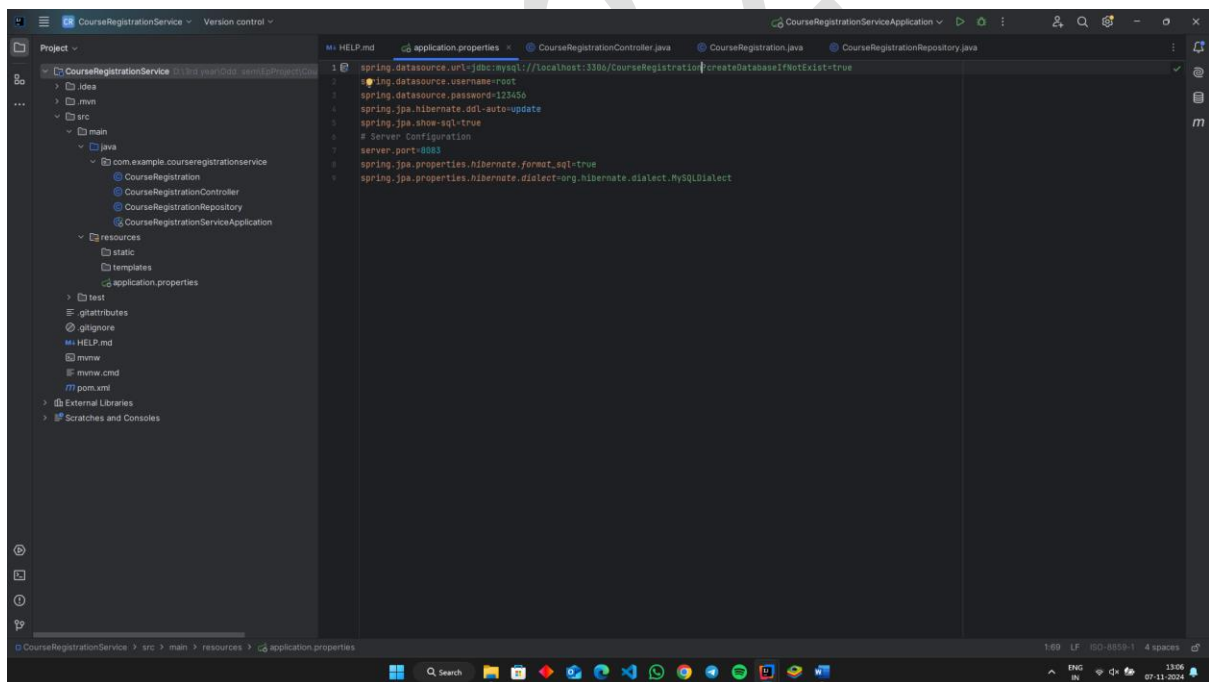
```
2024-11-07T12:36:24.700+05:30 INFO 22292 --- [main] c.e.s.StudentServiceApplication : Starting StudentServiceApplication using Java 22.0.2 with PID 22292 (D:\3rd year\Ods sem\IpProject\StudentService\src\main\java\com\example\studentservice\StudentServiceApplication.class)
2024-11-07T12:36:24.702+05:30 INFO 22292 --- [main] c.e.s.StudentServiceApplication : No active profile set, falling back to 1 default profile: "default"
2024-11-07T12:36:25.089+05:30 INFO 22292 --- [main] s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2024-11-07T12:36:25.121+05:30 INFO 22292 --- [main] s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 26 ms. Found 1 JPA repository interface.
2024-11-07T12:36:25.438+05:30 INFO 22292 --- [main] o.s.b.w.e.tomcat.TomcatWebServer : Tomcat initialized with port 8080 (http)
2024-11-07T12:36:25.448+05:30 INFO 22292 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2024-11-07T12:36:25.448+05:30 INFO 22292 --- [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.31]
2024-11-07T12:36:25.484+05:30 INFO 22292 --- [main] o.s.c.r.j.TomcatWebServer : Initializing Spring embedded WebApplicationContext
2024-11-07T12:36:25.484+05:30 INFO 22292 --- [main] o.s.c.r.j.TomcatWebServer : Root WebApplicationContext: initialization completed in 753 ms
2024-11-07T12:36:25.581+05:30 INFO 22292 --- [main] o.hibernate.jpa.internal.util.LogHelper : HH000026: Processing PersistenceUnitInfo [name: default]
2024-11-07T12:36:25.613+05:30 INFO 22292 --- [main] org.hibernate.version : HH000042: Hibernate ORM version 6.5.3.Final
2024-11-07T12:36:25.634+05:30 INFO 22292 --- [main] o.h.c.i.internal.RegionFactoryInitiator : HH000020: Second-level cache disabled
2024-11-07T12:36:25.856+05:30 INFO 22292 --- [main] o.s.c.r.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup; ignoring JPA class transformer
2024-11-07T12:36:25.876+05:30 INFO 22292 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2024-11-07T12:36:26.115+05:30 INFO 22292 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@4af91d5b
2024-11-07T12:36:26.117+05:30 INFO 22292 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2024-11-07T12:36:26.149+05:30 WARN 22292 --- [main] org.hibernate.orm.deprecation : HH000002: MySQLdialect does not need to be specified explicitly using 'hibernate.dialect' (remove the property set)
2024-11-07T12:36:26.174+05:30 INFO 22292 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HH000049: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integration)

Hibernate:
create table student (
  id bigint not null auto_increment,
  email varchar(255),
  name varchar(255),
  primary key (id)
) engine=InnoDB

2024-11-07T12:36:26.721+05:30 INFO 22292 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2024-11-07T12:36:26.880+05:30 WARN 22292 --- [main] jpabaseconfiguration.jpawebconfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering.
2024-11-07T12:36:27.120+05:30 INFO 22292 --- [main] o.s.b.w.e.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'
2024-11-07T12:36:27.125+05:30 INFO 22292 --- [main] c.e.s.StudentServiceApplication : Started StudentServiceApplication in 2.08 seconds (process running for 3.114)
```

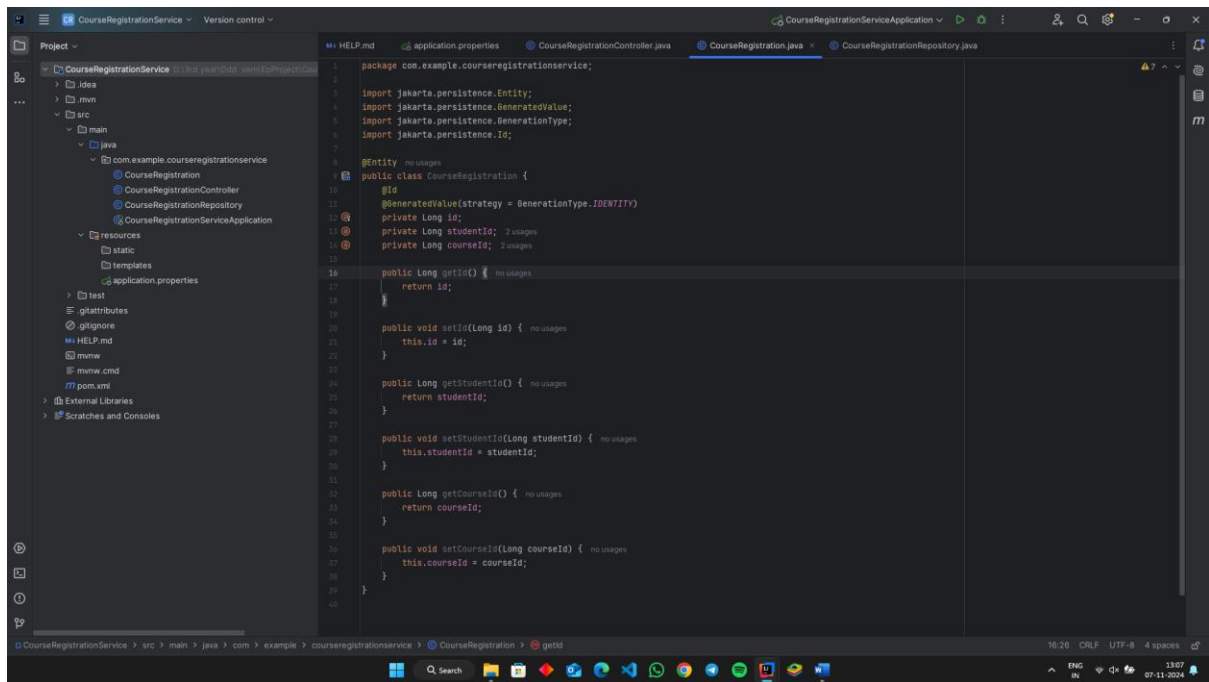
## CourseRegistrationService

### 1.Application.Properties

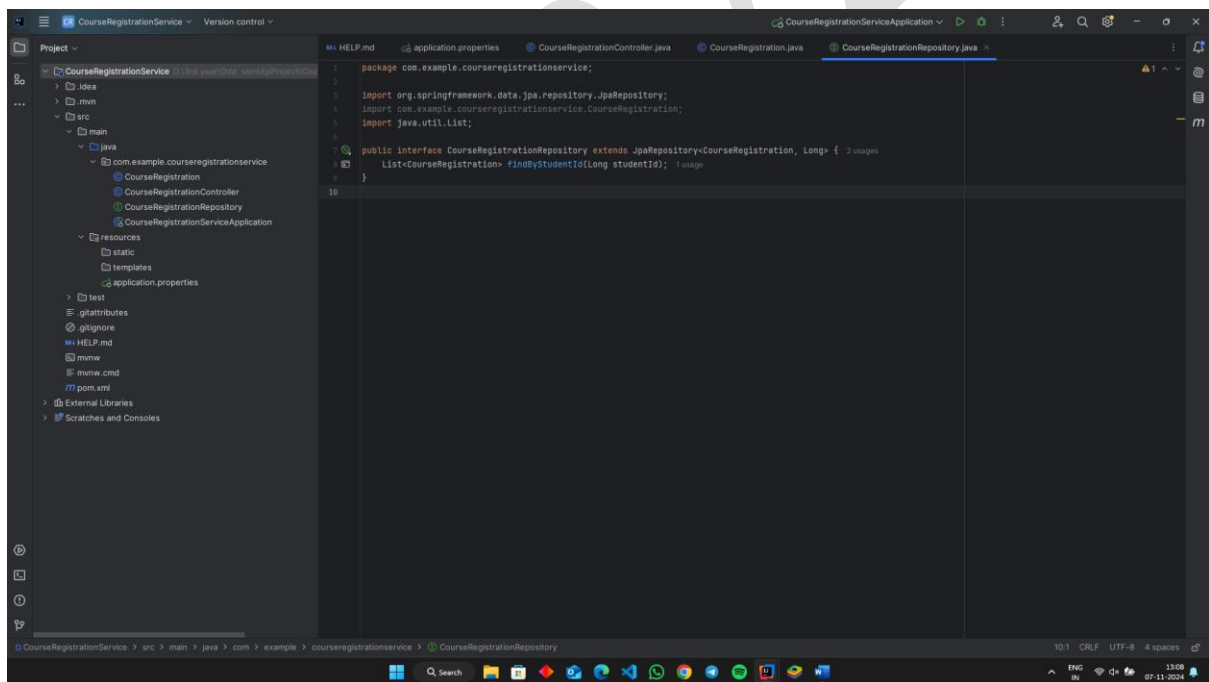


```
spring.datasource.url=jdbc:mysql://localhost:3306/CourseRegistration?createDatabaseIfNotExist=true
spring.datasource.username=root
spring.datasource.password=123456
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
# Server Configuration
server.port=8080
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

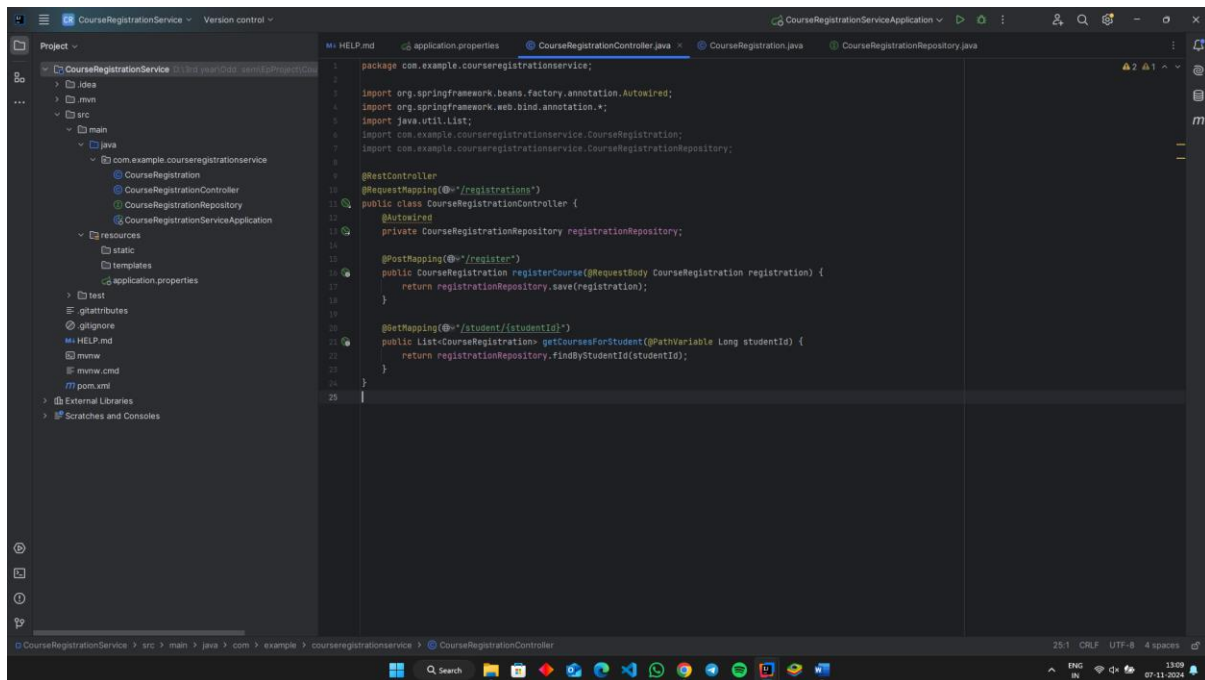
## 2.CourseRegistration.java



## 3.CourseRegistrationRepository

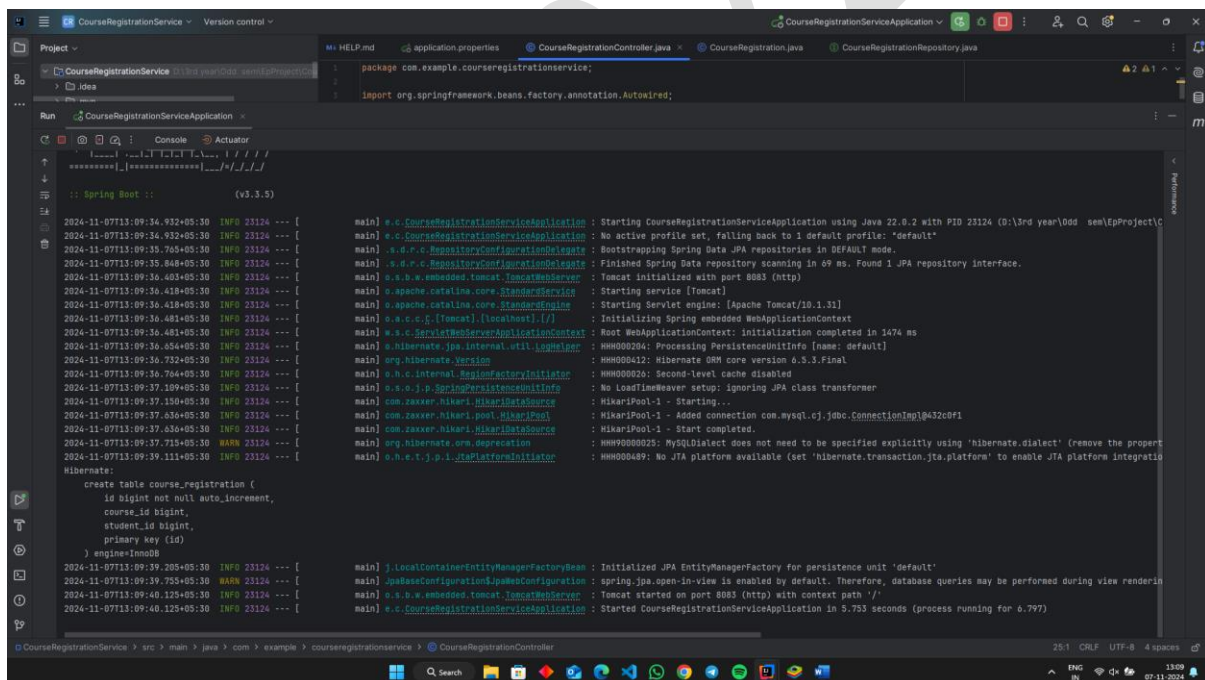


## 4.CourseRegistrationController



```
1 package com.example.courseregistration;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.web.bind.annotation.*;
5 import java.util.List;
6 import com.example.courseregistration.CourseRegistration;
7 import com.example.courseregistration.CourseRegistrationRepository;
8
9 @RestController
10 @RequestMapping("/registrations")
11 public class CourseRegistrationController {
12     @Autowired
13     private CourseRegistrationRepository registrationRepository;
14
15     @PostMapping("/register")
16     public CourseRegistration registerCourse(@RequestBody CourseRegistration registration) {
17         return registrationRepository.save(registration);
18     }
19
20     @GetMapping("/{student}/{studentId}")
21     public List<CourseRegistration> getCoursesForStudent(@PathVariable Long studentId) {
22         return registrationRepository.findByStudentId(studentId);
23     }
24 }
25
```

## 5.Terminal



```
2024-11-07T13:09:34.932+05:30 INFO 23124 --- [main] e.c.CourseRegistrationServiceApplication : Starting CourseRegistrationServiceApplication using Java 22.0.2 with PID 23124 (D:\3rd year\Ods sem\EpProject\CourseRegistrationService\src\main\java\com\example\courseregistration\CourseRegistrationServiceApplication.class)
2024-11-07T13:09:34.932+05:30 INFO 23124 --- [main] e.c.CourseRegistrationServiceApplication : No active profile set, falling back to default profile: 'default'
2024-11-07T13:09:35.765+05:30 INFO 23124 --- [main] e.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2024-11-07T13:09:35.848+05:30 INFO 23124 --- [main] e.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 69 ms. Found 1 JPA repository interface.
2024-11-07T13:09:36.403+05:30 INFO 23124 --- [main] e.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8083 (http)
2024-11-07T13:09:36.418+05:30 INFO 23124 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2024-11-07T13:09:36.418+05:30 INFO 23124 --- [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.11]
2024-11-07T13:09:36.481+05:30 INFO 23124 --- [main] e.s.c.s.g.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2024-11-07T13:09:36.481+05:30 INFO 23124 --- [main] e.s.c.s.g.[Tomcat].[localhost].[/] : Root WebApplicationContext: initialization completed in 1474 ms
2024-11-07T13:09:36.654+05:30 INFO 23124 --- [main] e.hibernate.jpa.internal.util.LogHelper : HH0000204: Processing PersistenceUnitInfo [name: default]
2024-11-07T13:09:36.732+05:30 INFO 23124 --- [main] org.hibernate.Version : HH0000412: Hibernate ORM core version 6.5.3.Final
2024-11-07T13:09:36.764+05:30 INFO 23124 --- [main] e.s.c.internal.RegionFactoryInitiator : HH0000026: Second-level cache disabled
2024-11-07T13:09:37.109+05:30 INFO 23124 --- [main] e.s.c.j.p.SpringPersistenceUnitInfo : No LossOfMemorySetup: ignoring JPA class transformer
2024-11-07T13:09:37.150+05:30 INFO 23124 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2024-11-07T13:09:37.636+05:30 INFO 23124 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@432c0f1
2024-11-07T13:09:37.636+05:30 INFO 23124 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2024-11-07T13:09:37.715+05:30 WARN 23124 --- [main] org.hibernate.orm.deprecation : HH00000025: MySQLDialect does not need to be specified explicitly using 'hibernate.dialect' (remove the property 'hibernate.dialect')
2024-11-07T13:09:39.111+05:30 INFO 23124 --- [main] e.h.e.t.j.p.i.JpaPlatformInitiator : HH0000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integration)
2024-11-07T13:09:39.205+05:30 INFO 23124 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2024-11-07T13:09:39.755+05:30 WARN 23124 --- [main] jpaBaseConfiguration.jpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering.
2024-11-07T13:09:40.125+05:30 INFO 23124 --- [main] e.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8083 (http) with context path '/'
2024-11-07T13:09:40.125+05:30 INFO 23124 --- [main] e.c.CourseRegistrationServiceApplication : Started CourseRegistrationServiceApplication in 5.753 seconds (process running for 6.797)
```

## Outputs in postman

## Update Student

The screenshot shows the Postman application interface. On the left, a sidebar displays a collection of API requests under 'My Workspace'. The 'UpdateStudent' request is selected. The main panel shows the details of this request:

- Method:** PUT
- URL:** http://localhost:8081/students/update/1
- Body:** The request body is a JSON object: 

```
{  "name": "Dheeraj keta"}
```
- Response:** The response is a 200 OK status, indicating the update was successful. The response body is a JSON object: 

```
{  "id": 1,  "name": "Dheeraj keta",  "email": "rs@klu.edu"}
```

## Delete Student

The screenshot shows the Postman application interface. On the left, a sidebar displays a collection of API requests under 'My Workspace'. The 'DeleteStudents' request is selected. The main panel shows the details of this request:

- Method:** DELETE
- URL:** http://localhost:8081/students/delete/1
- Response:** The response is a 200 OK status, indicating the deletion was successful. The response body is empty.

## UpdateCourse

The screenshot shows the Postman interface for a PUT request named 'UpdateCourse' at the URL 'http://localhost:8082/courses/update/1'. The request body is a JSON object: 

```
{  "courseId": 1,  "name": "Advanced Java",  "description": "Learn advanced topics in Java programming"}
```

. The response is a 200 OK status with a 242 ms response time and 257 B of data. The response body is a JSON object: 

```
{  "id": 1,  "courseName": "Data Structures",  "courseDescription": "Introduction to Data Structures"}
```

.

My Workspace

Lab 7

Lab 8

Lab 9

Lab 10

POST AddStudent

GET GetAllStudents

GET AddCourse

GET GetAllCourses

GET RegisterCourse

GET getallregisteredCourses

Lab 11

GET UpdateStudent

GET DeleteStudents

GET UpdateCourse

Lab 12

New Collection

PUT UpdateCourse

http://localhost:8082/courses/update/1

Params Authorization Headers (8) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "courseId": 1,
3   "name": "Advanced Java",
4   "description": "Learn advanced topics in Java programming"
5 }
6
```

Body Cookies Headers (8) Test Results

200 OK · 242 ms · 257 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "courseName": "Data Structures",
4   "courseDescription": "Introduction to Data Structures"
5 }
```

## DeleteCourse

The screenshot shows the Postman interface for a DELETE request named 'DeleteCourse' at the URL 'http://localhost:8082/courses/delete/2'. The response is a 200 OK status with a 65 ms response time and 123 B of data. The response body is a single character '1'.

My Workspace

Lab 7

Lab 8

Lab 9

Lab 10

POST AddStudent

GET GetAllStudents

GET AddCourse

GET GetAllCourses

GET RegisterCourse

GET getallregisteredCourses

Lab 11

GET UpdateStudent

GET DeleteStudents

GET UpdateCourse

Lab 12

New Collection

DELETE DeleteCourse

http://localhost:8082/courses/delete/2

Params Authorization Headers (8) Body Scripts Settings

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (4) Test Results

200 OK · 65 ms · 123 B

Pretty Raw Preview Visualize Text

```
1
```



## UpdateRegisteredCourse

The screenshot shows the Postman application interface. On the left, a sidebar displays a collection of API requests under 'My Workspace'. The 'UpdateRegisteredCourse' request is selected. The main panel shows the details of this request:

- Method:** PUT
- URL:** http://localhost:8083/registrations/update/1
- Body:** The request body is a JSON object: 

```
{  "studentId": 2,  "courseId": 2}
```
- Response:** The response is a 200 OK status, indicating a successful update. The response body is a JSON object: 

```
{  "id": 1,  "studentId": 2,  "courseId": 2}
```

## DeleteRegisteredCourse

The screenshot shows the Postman application interface. On the left, a sidebar displays a collection of API requests under 'My Workspace'. The 'DeleteRegisteredCourse' request is selected. The main panel shows the details of this request:

- Method:** DELETE
- URL:** http://localhost:8083/registrations/delete/1
- Response:** The response is a 200 OK status, indicating a successful deletion. The response body is a simple text response: 

```
1
```



2200032247