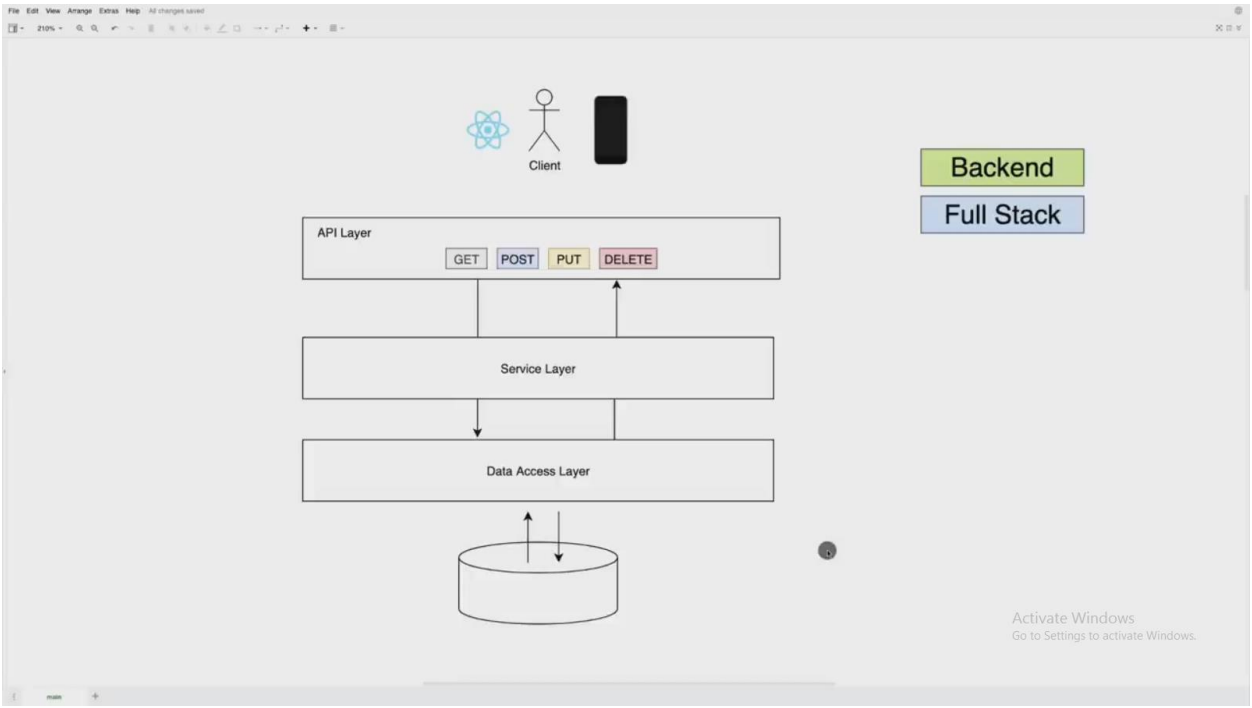
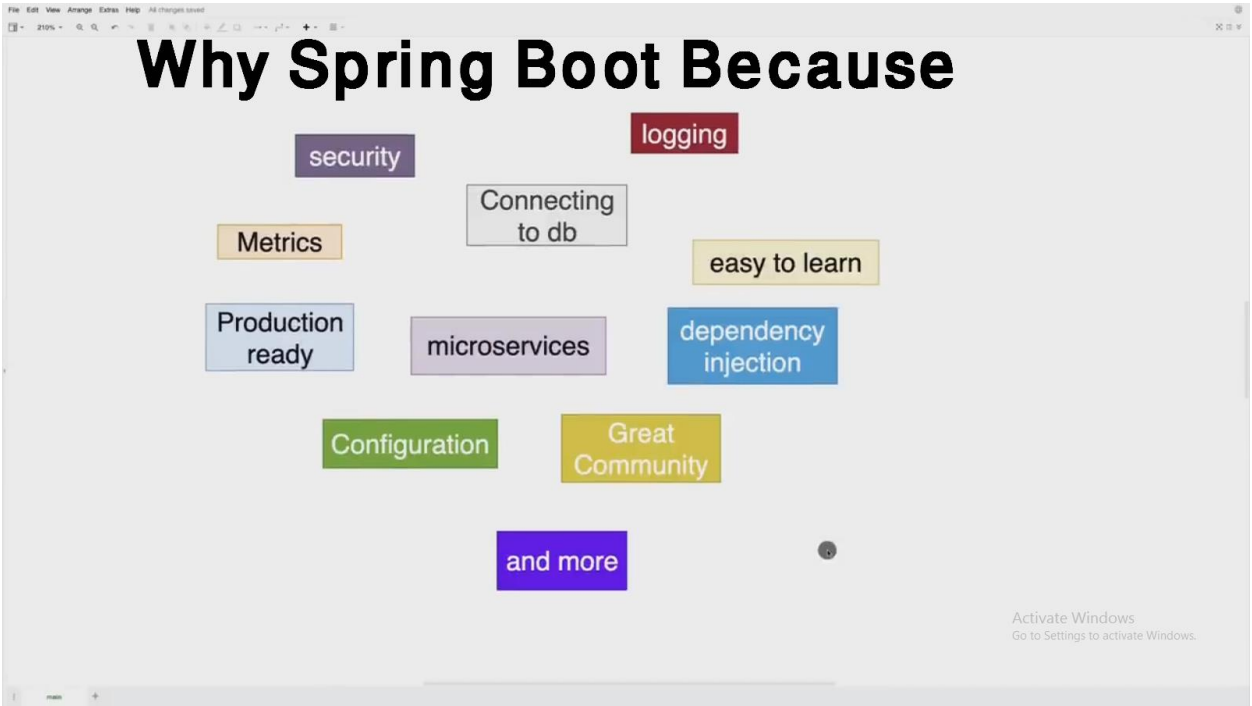
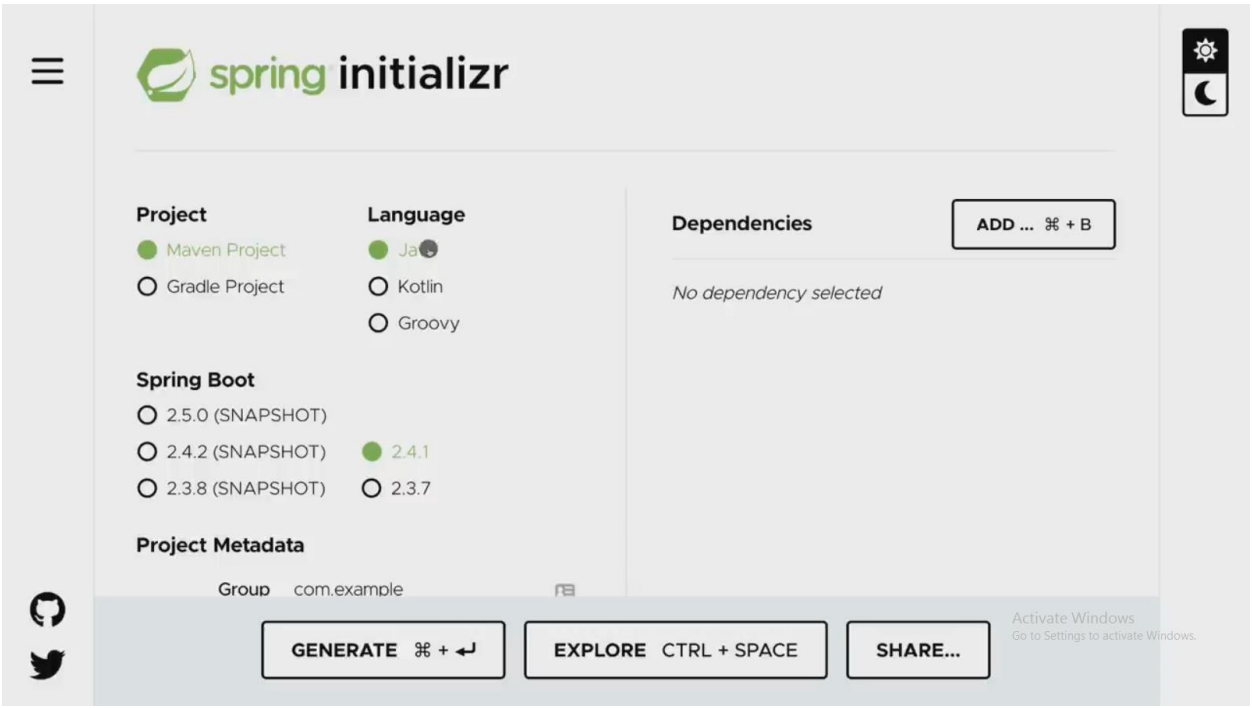
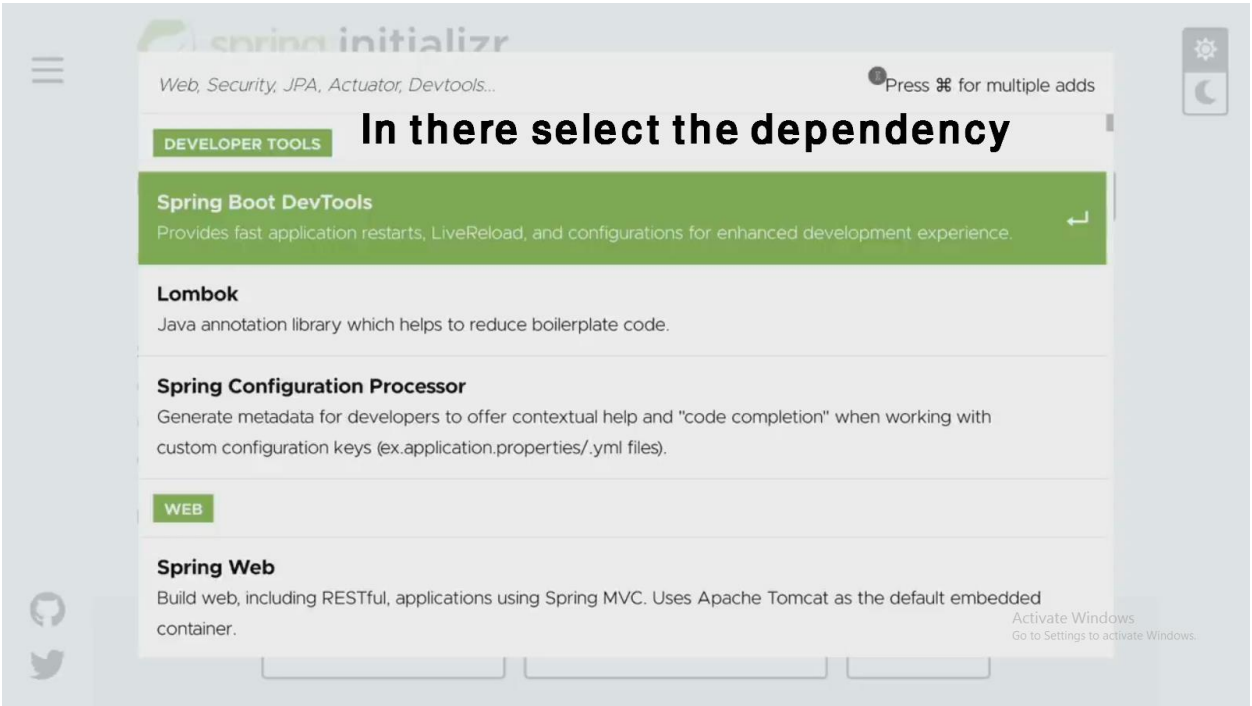
SPRING BOOT FULL COURSE

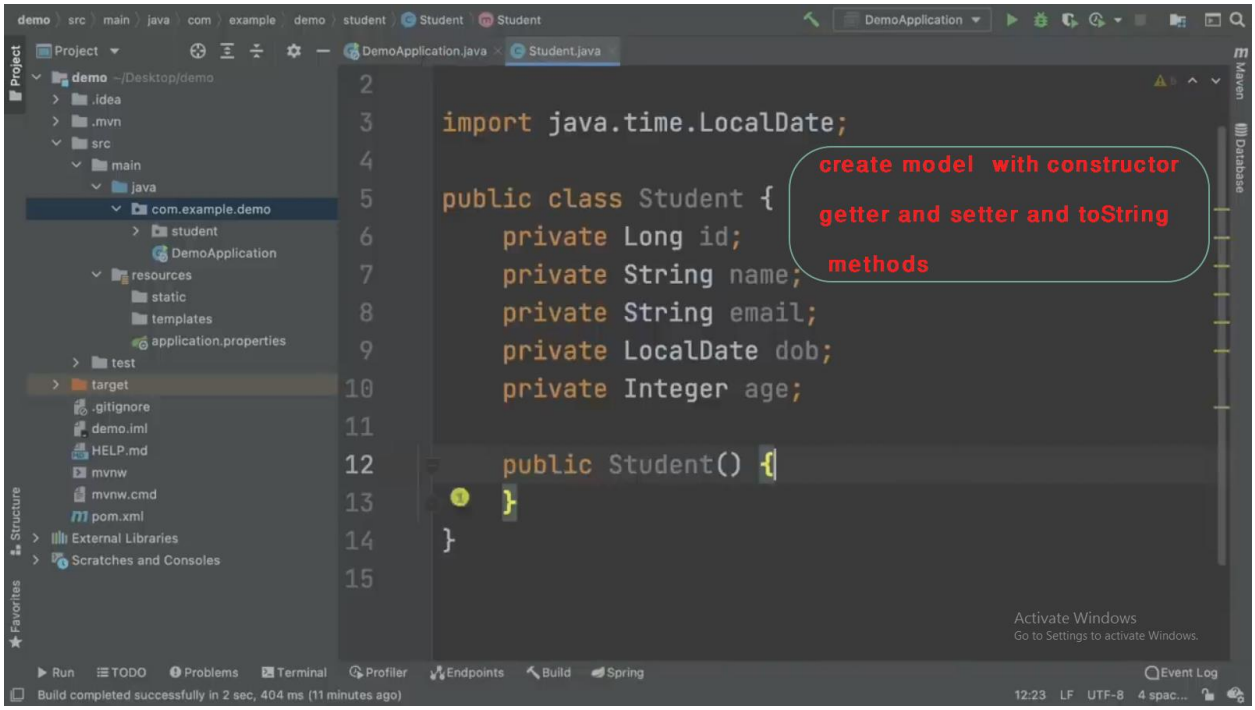

Client
Backend
Full Stack
API Layer
GET
POST
PUT
DELETE
Service Layer
Data Access Layer
Activate Windows
Go to Settings to activate Windows.

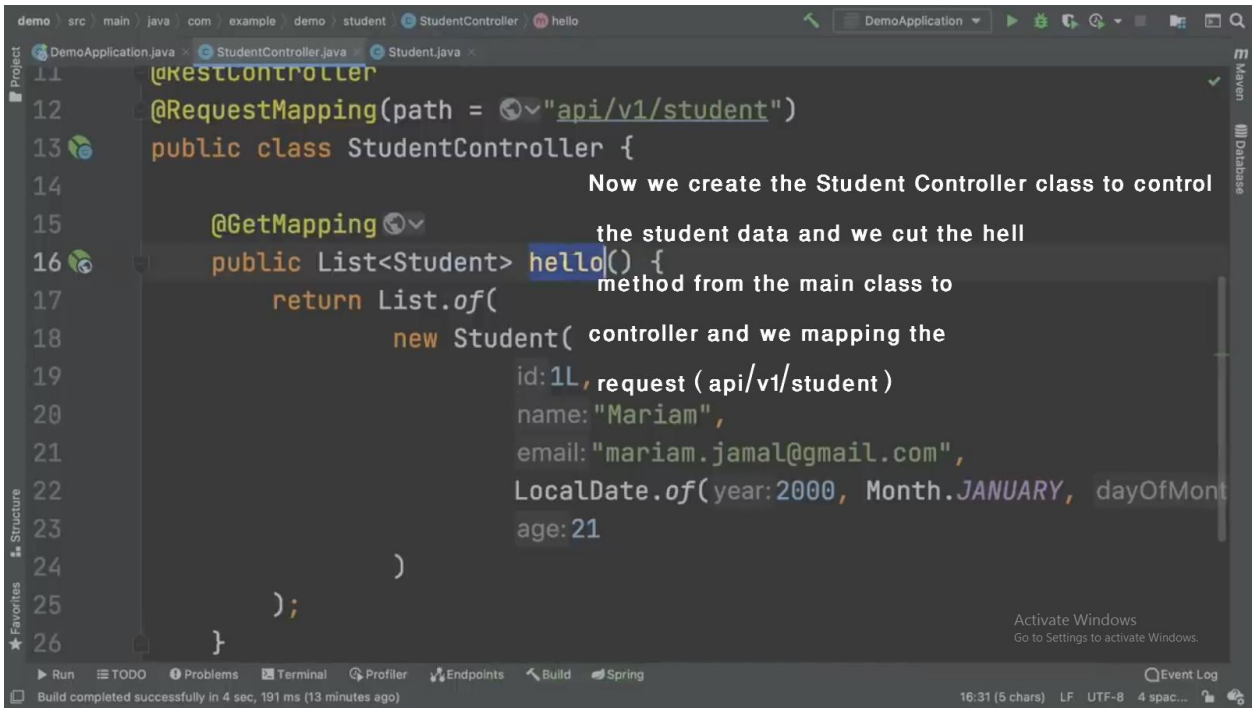

Why Spring Boot Because
security
logging
Connecting to db
Metrics
easy to learn
Production ready
microservices
dependency injection
Configuration
Great Community
and more
Activate Windows
Go to Settings to activate Windows.

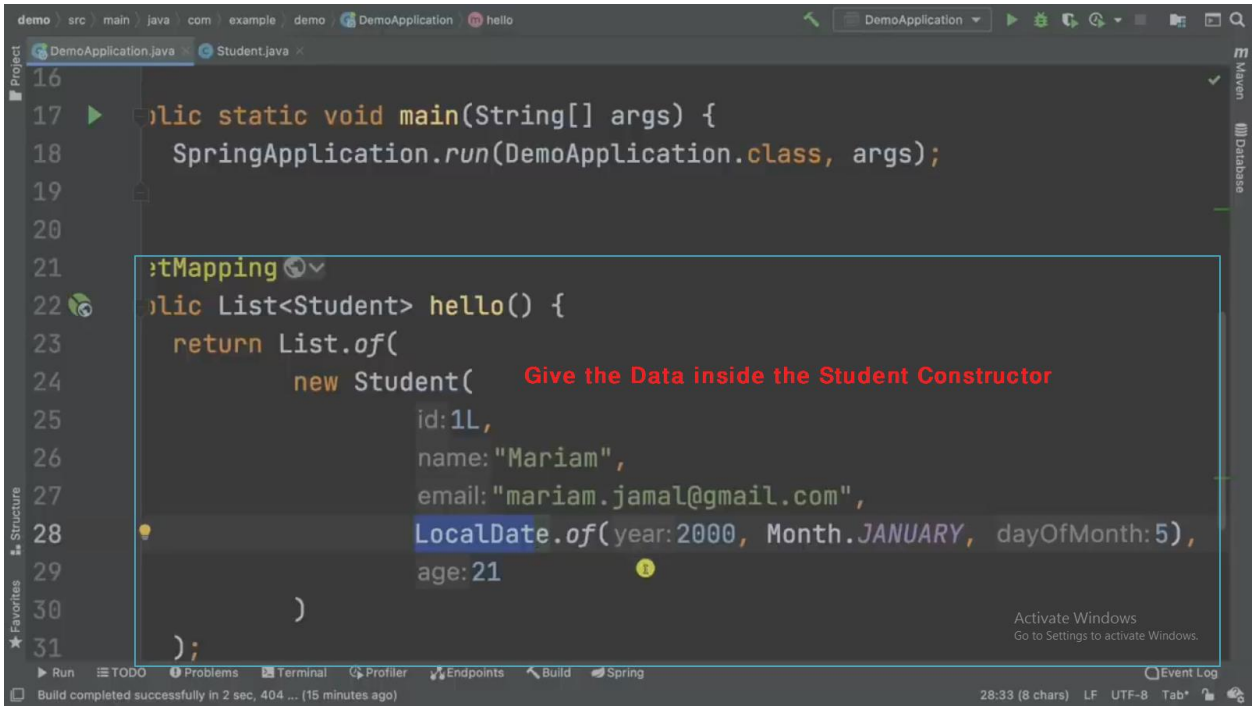


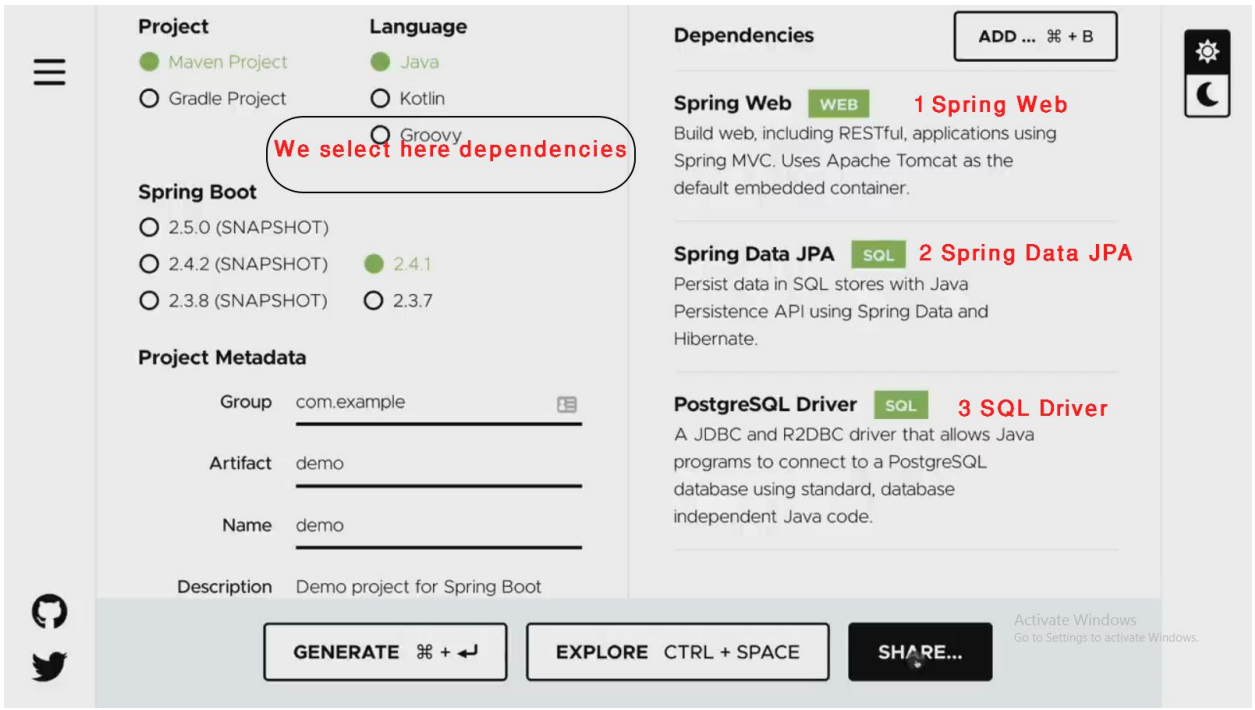


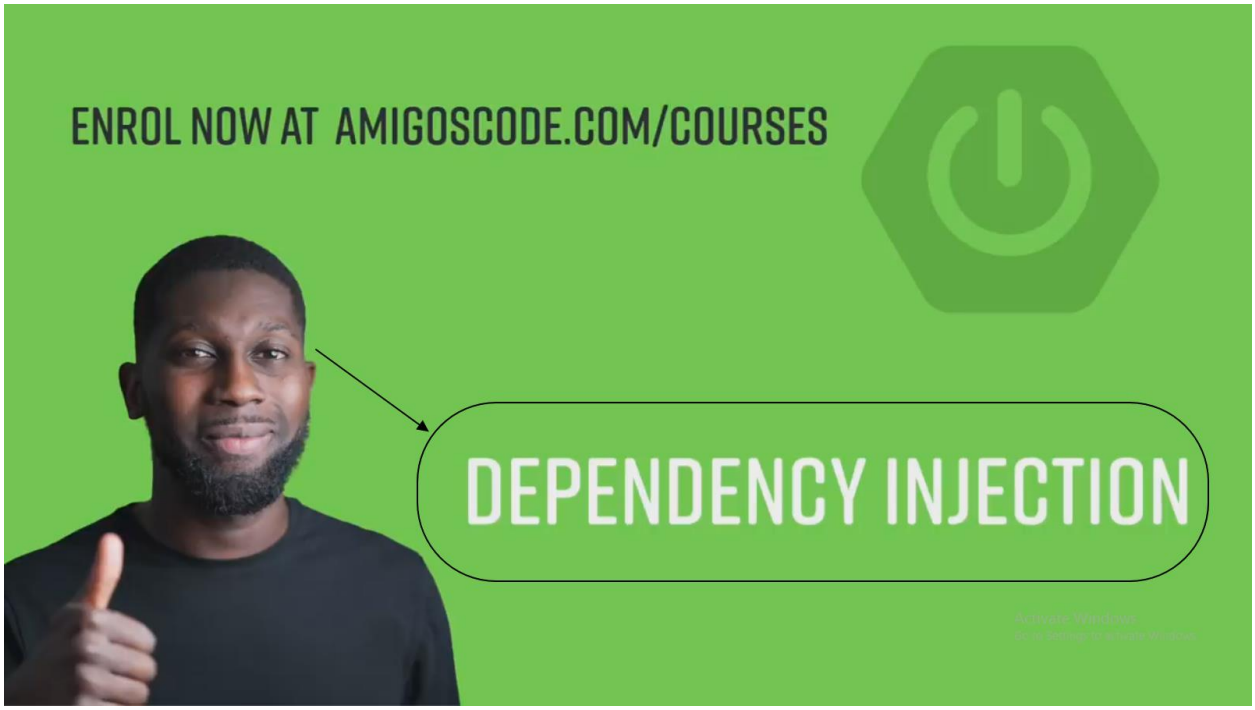
In there select the dependency

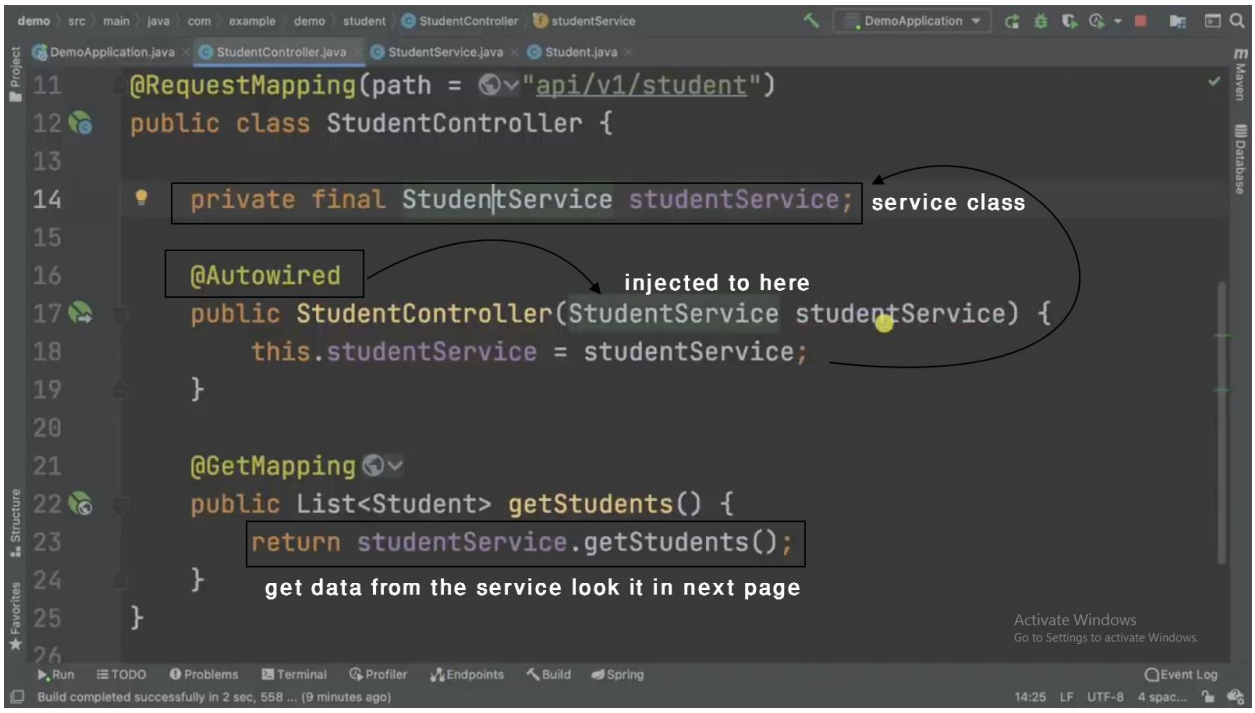

create model with constructor
getter and setter and toString
methods
import java.time.LocalDate;
public class Student {
private Long id;
private String name;
private String email;
private LocalDate dob;
private Integer age;
public Student() {
}
}

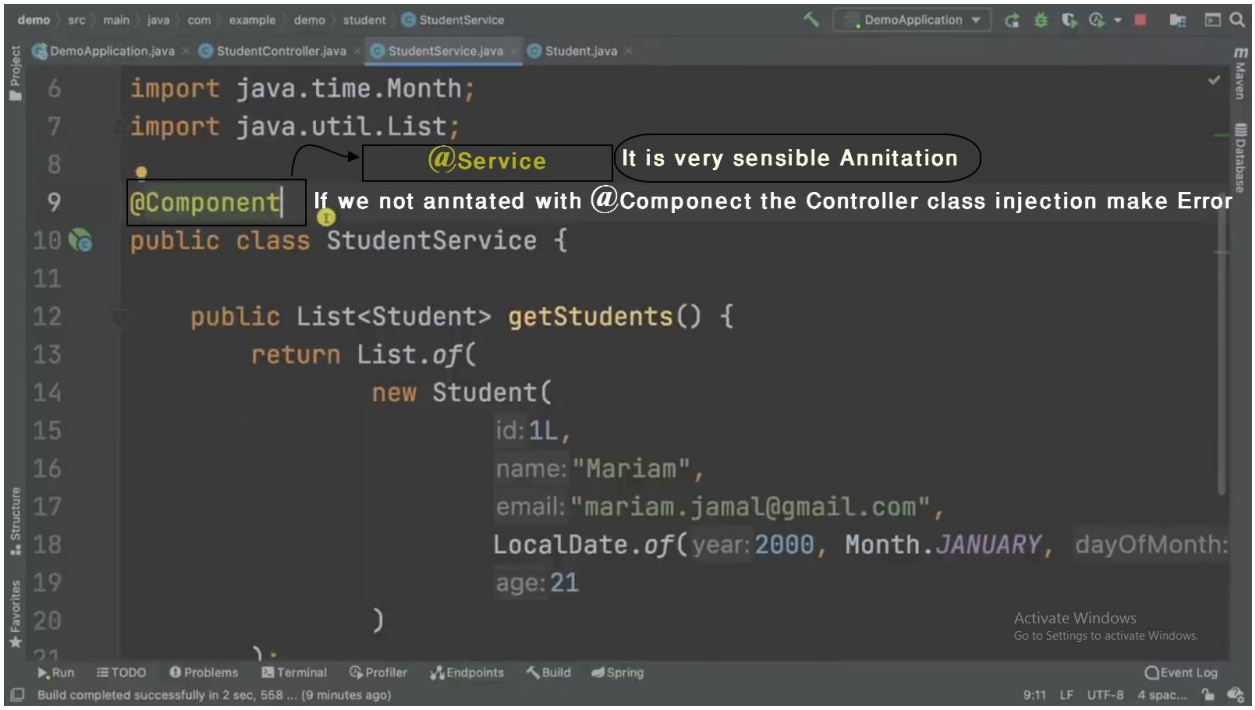

Now we create the Student Controller class to control
the student data and we cut the hell
method from the main class to
controller and we mapping the
request (api/v1/student)
@RestController
@RequestMapping(path = "api/v1/student")
public class StudentController {
@GetMapping
public List<Student> hello() {
return List.of(
new Student(
id:1L,
name:"Mariam",
email:"mariam.jamal@gmail.com",
LocalDate.of(year:2000, Month.JANUARY,
age:21
)
);
}

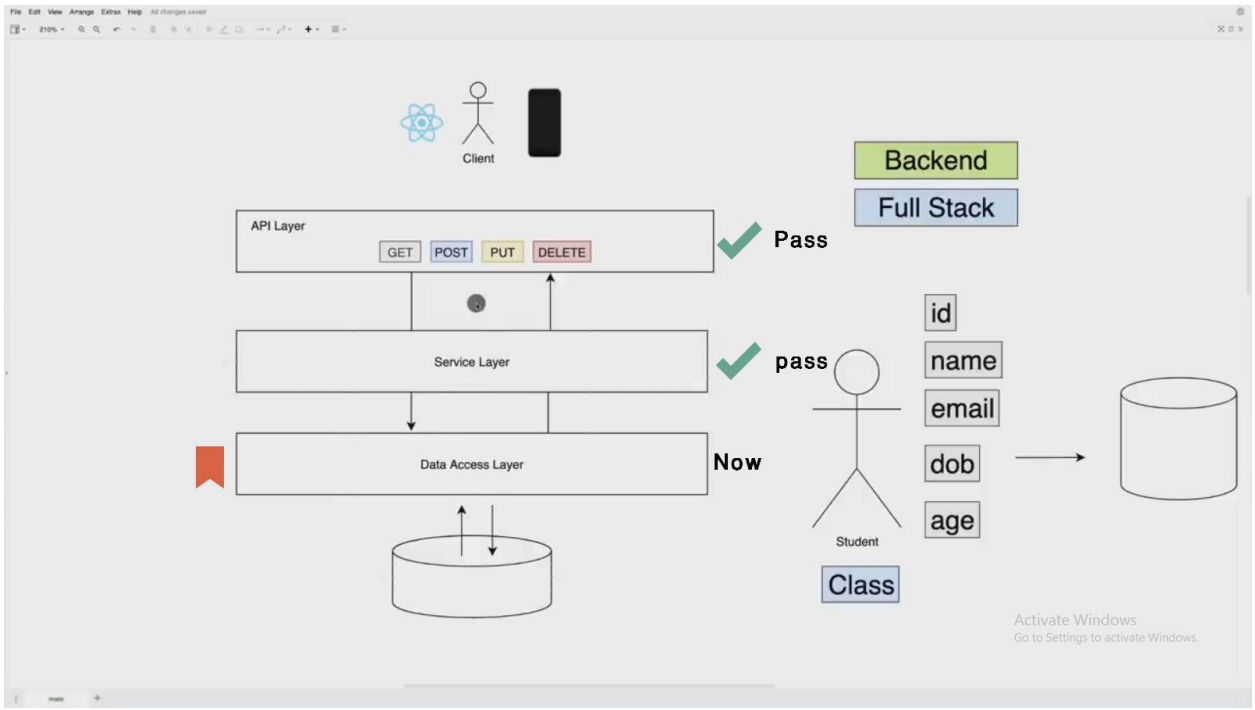

Give the Data inside the Student Constructor
Activate Windows
Go to Settings to activate Windows.

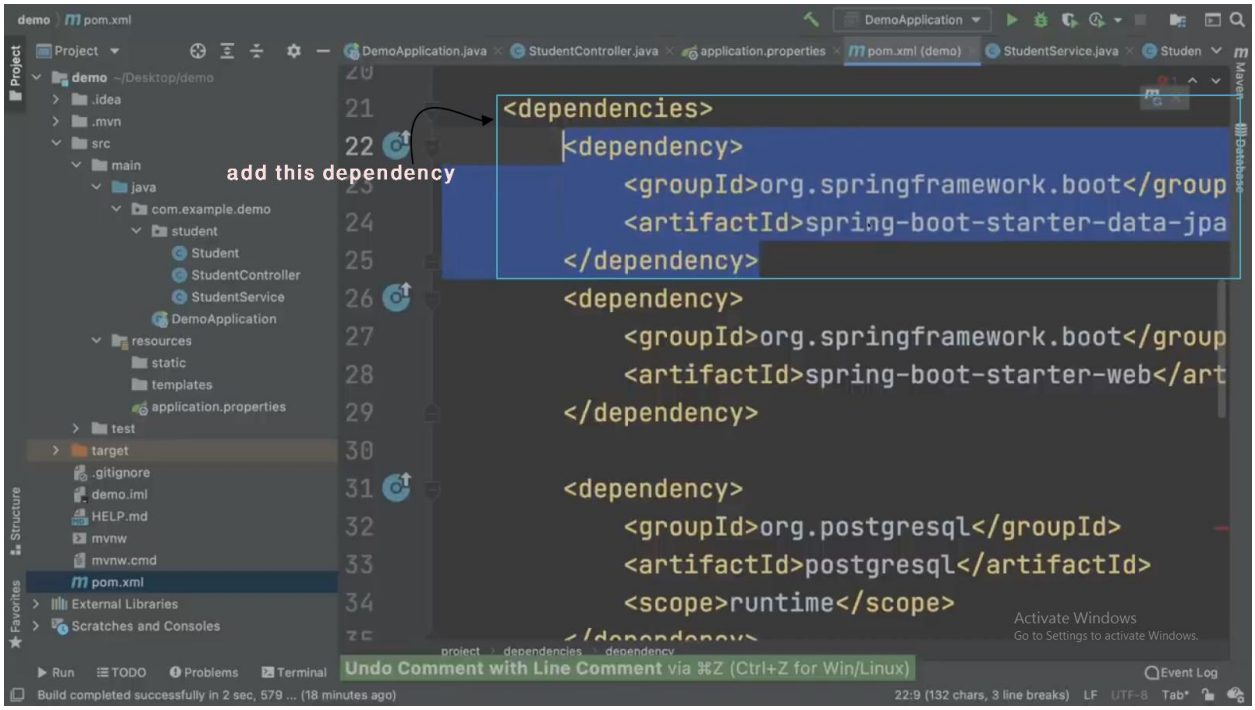

Project
Maven Project
Gradle Project
Language
Java
Kotlin
Groovy
We select here dependencies
Spring Boot
2.5.0 (SNAPSHOT)
2.4.2 (SNAPSHOT)
2.4.1
2.3.8 (SNAPSHOT)
2.3.7
Project Metadata
Group
com.example
Artifact
demo
Name
demo
Description
Demo project for Spring Boot
Dependencies
ADD ... ⌘ + B
Spring Web
WEB
1 Spring Web
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
Spring Data JPA
SQL
2 Spring Data JPA
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.
PostgreSQL Driver
SQL
3 SQL Driver
A JDBC and R2DBC driver that allows Java programs to connect to a PostgreSQL database using standard, database independent Java code.
GENERATE ⌘ + ↵
EXPLORE CTRL + SPACE
SHARE...
Activate Windows
Go to Settings to activate Windows.


ENROL NOW AT AMIGOSCODE.COM/COURSES
DEPENDENCY INJECTION

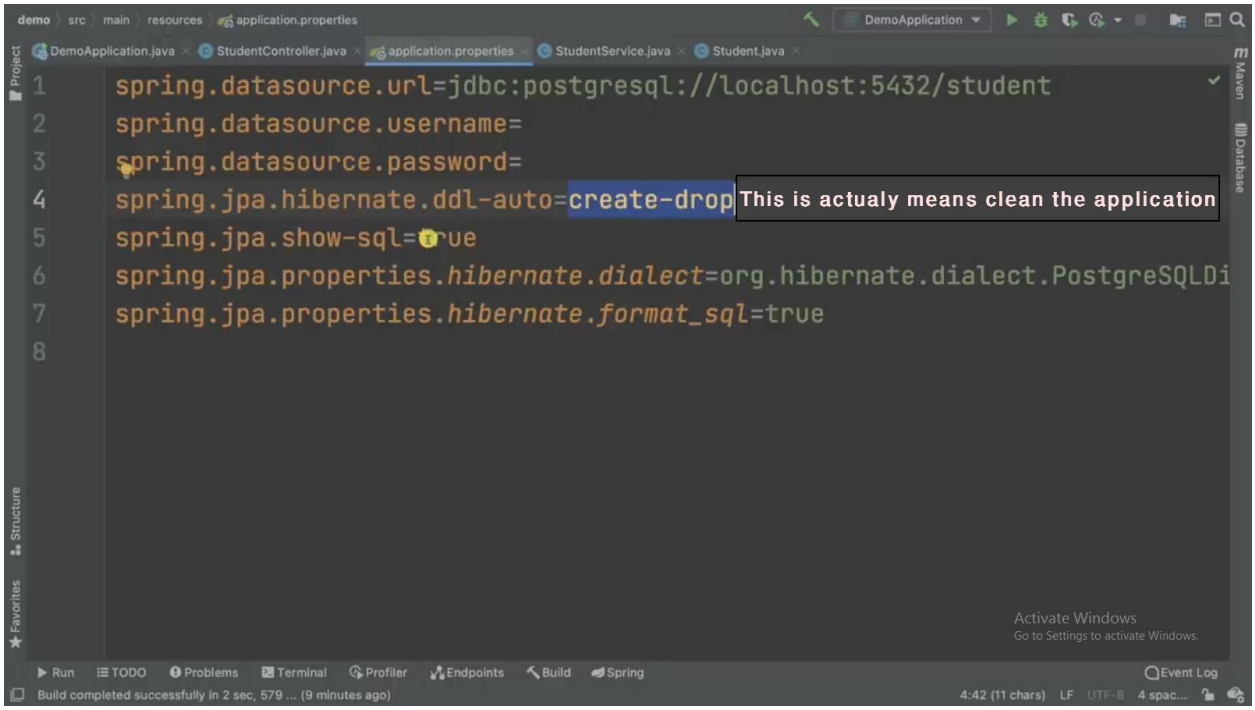

@RequestMapping(path = "api/v1/student")
public class StudentController {
private final StudentService studentService;
service class
@Autowired
injected to here
public StudentController(StudentService studentService) {
this.studentService = studentService;
}
@GetMapping
public List<Student> getStudents() {
return studentService.getStudents();
}
get data from the service look it in next page
}


@Service
It is very sensible Annitation
@Component
If we not anntated with @Componect the Controller class injection make Error
Activate Windows
Go to Settings to activate Windows.

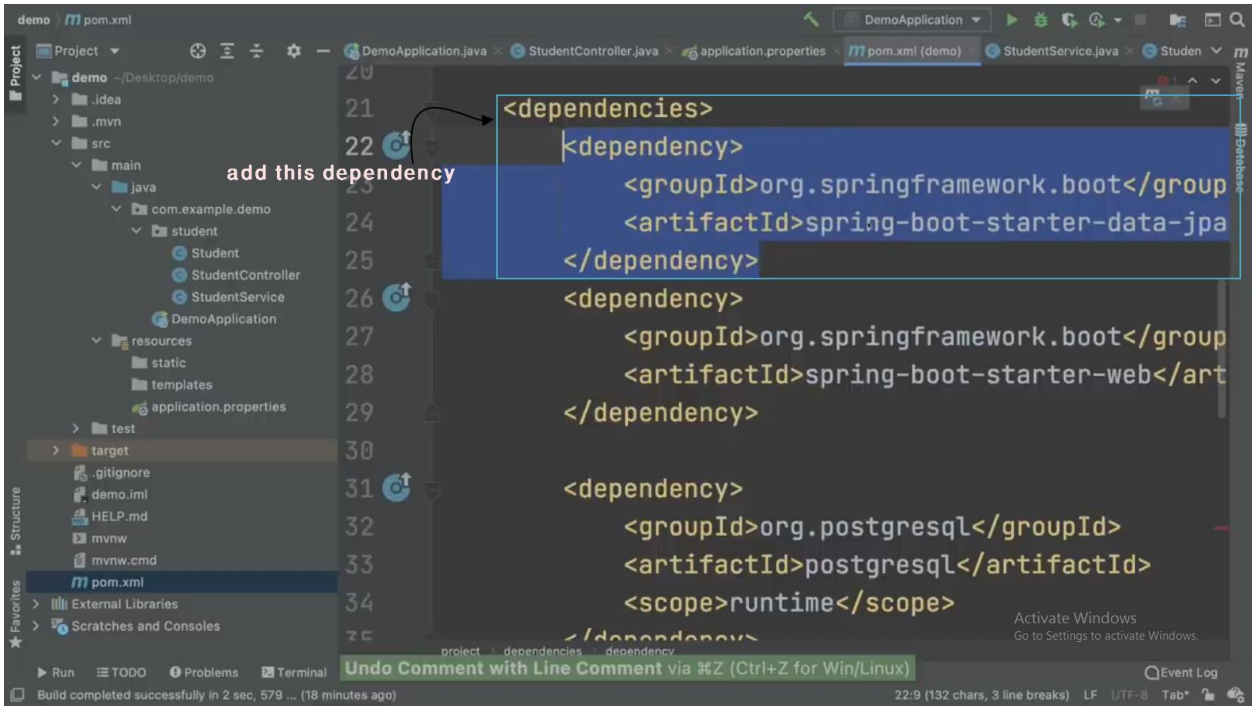

Client
Backend
Full Stack
API Layer
GET
POST
PUT
DELETE
Pass
Service Layer
pass
Data Access Layer
Now
id
name
email
dob
age
Student
Class
Activate Windows
Go to Settings to activate Windows.

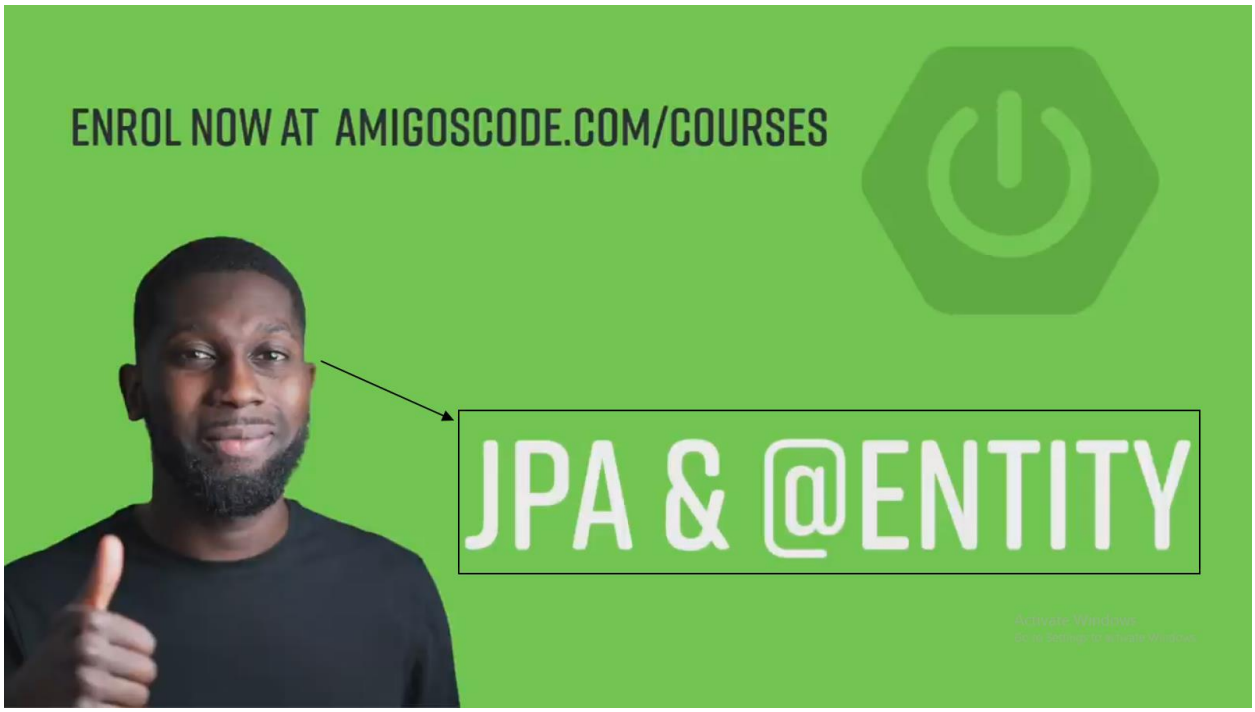

add this dependency
<dependencies>
<dependency>
<groupId>org.springframework.boot</group
<artifactId>spring-boot-starter-data-jpa
</dependency>
<dependency>
<groupId>org.springframework.boot</group
<artifactId>spring-boot-starter-web</art
</dependency>
<dependency>
<groupId>org.postgresql</groupId>
<artifactId>postgresql</artifactId>
<scope>runtime</scope>
Undo Comment with Line Comment via ⌘Z (Ctrl+Z for Win/Linux)
Build completed successfully in 2 sec, 579 ... (18 minutes ago)

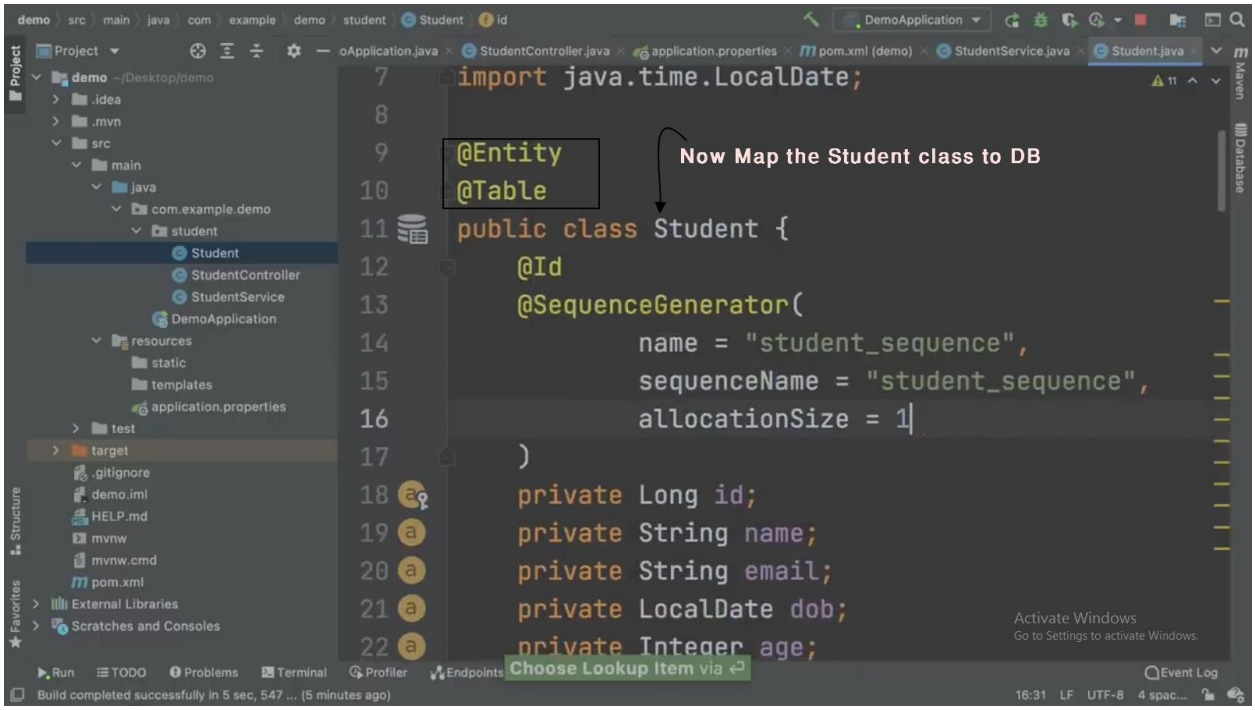

ENROL NOW AT AMIGOSCODE.COM/COURSES
PROPERTIES FILE

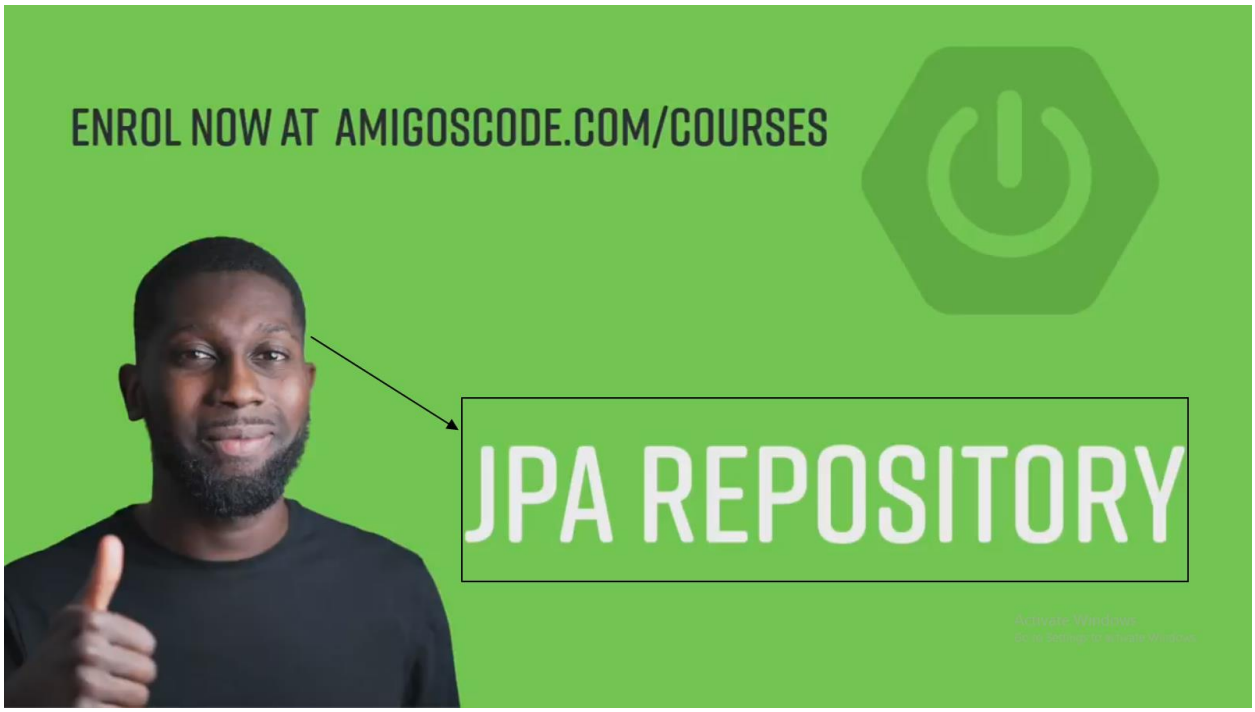

demo src main resources application.properties
DemoApplication
DemoApplication.java StudentController.java application.properties StudentService.java Student.java
spring.datasource.url=jdbc:postgresql://localhost:5432/student
spring.datasource.username=
spring.datasource.password=
spring.jpa.hibernate.ddl-auto=create-drop
This is actualy means clean the application
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDi
spring.jpa.properties.hibernate.format_sql=true
Activate Windows
Go to Settings to activate Windows.
Run TODO Problems Terminal Profiler Endpoints Build Spring
Event Log
Build completed successfully in 2 sec, 579 ... (9 minutes ago)
4:42 (11 chars) LF UTF-8 4 spac...

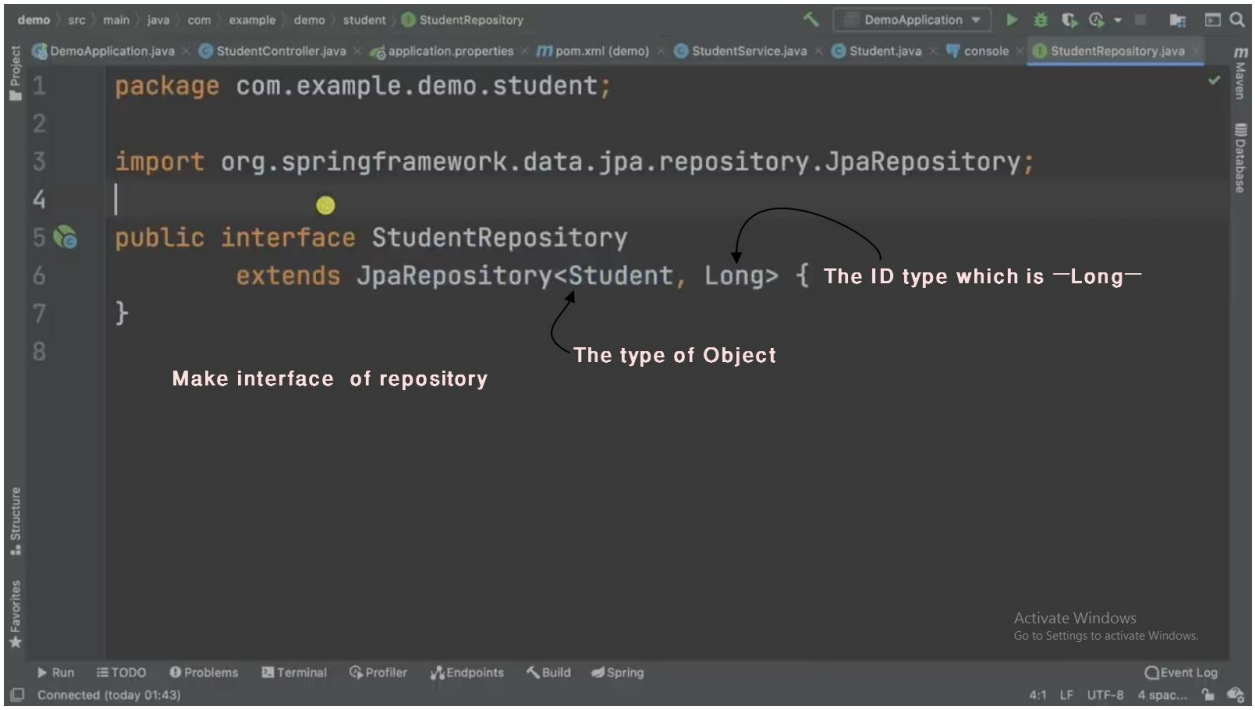

ENROL NOW AT AMIGOSCODE.COM/COURSES
CONNECTING TO DATABASE
Activate Windows

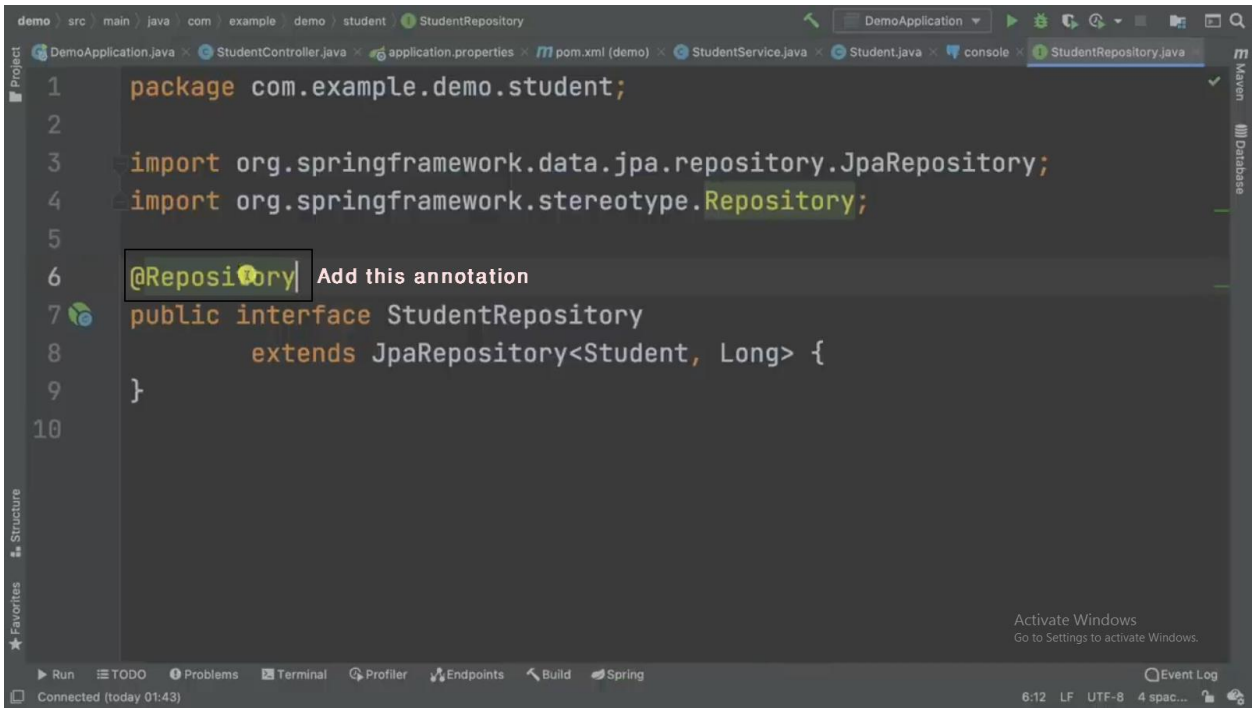

add this dependency
<dependencies>
<dependency>
<groupId>org.springframework.boot</group
<artifactId>spring-boot-starter-data-jpa
</dependency>
<dependency>
<groupId>org.springframework.boot</group
<artifactId>spring-boot-starter-web</art
</dependency>
<dependency>
<groupId>org.postgresql</groupId>
<artifactId>postgresql</artifactId>
<scope>runtime</scope>
Undo Comment with Line Comment via ⌘Z (Ctrl+Z for Win/Linux)
Build completed successfully in 2 sec, 579 ... (18 minutes ago)

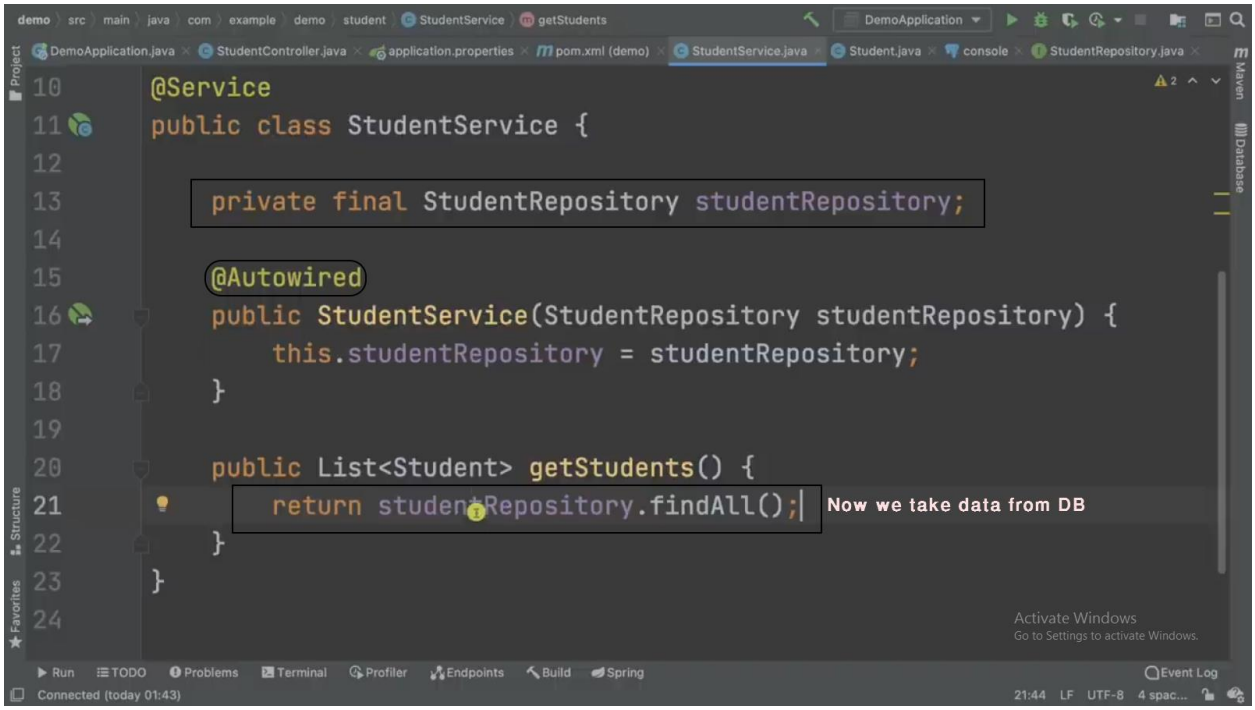

ENROL NOW AT AMIGOSCODE.COM/COURSES
JPA & @ENTITY


Now Map the Student class to DB
import java.time.LocalDate;
@Entity
@Table
public class Student {
@Id
@SequenceGenerator(
name = "student_sequence",
sequenceName = "student_sequence",
allocationSize = 1
)
private Long id;
private String name;
private String email;
private LocalDate dob;
private Integer age;
Activate Windows
Go to Settings to activate Windows.
Choose Lookup Item via
Build completed successfully in 5 sec, 547 ... (5 minutes ago)

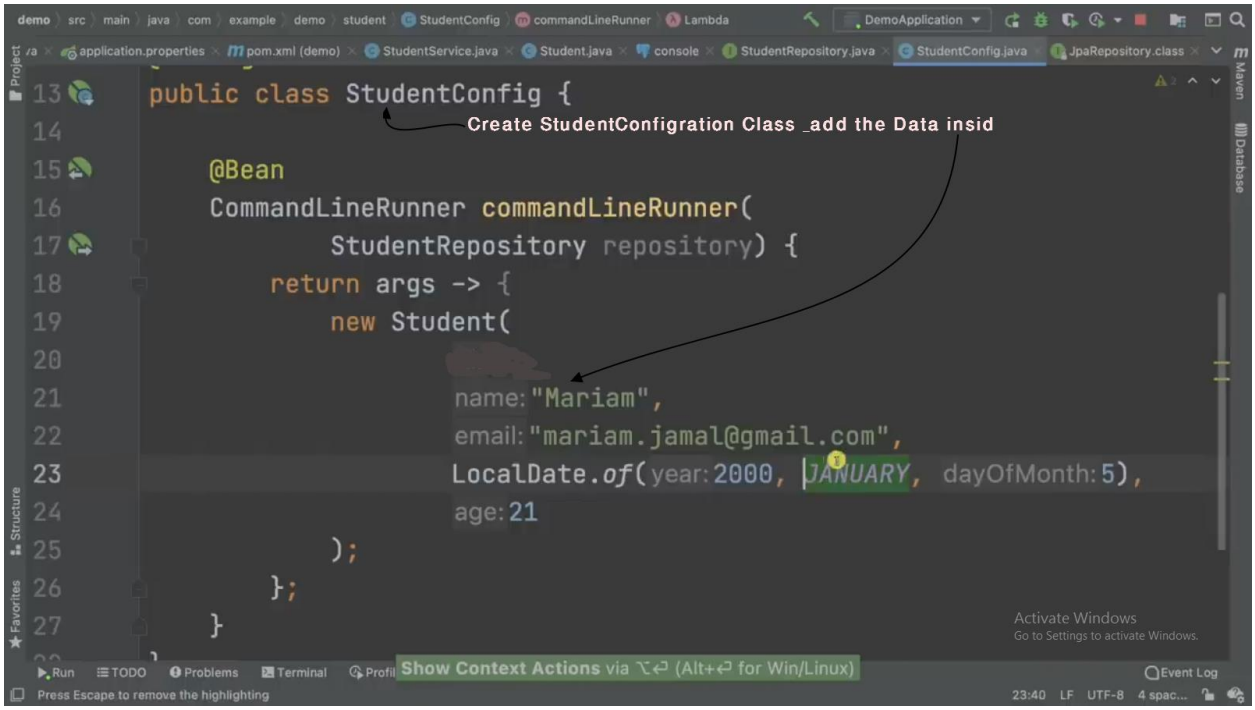

ENROL NOW AT AMIGOSCODE.COM/COURSES
JPA REPOSITORY

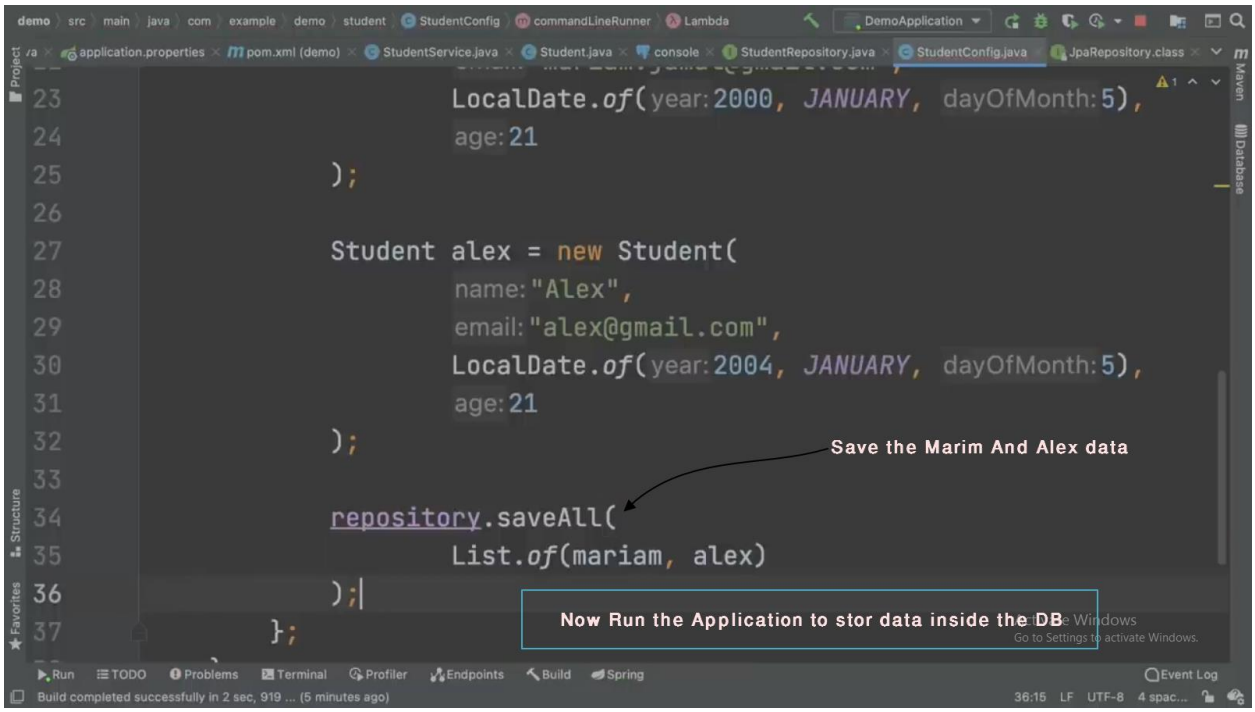

package com.example.demo.student;
import org.springframework.data.jpa.repository.JpaRepository;
public interface StudentRepository
extends JpaRepository<Student, Long> {
}
The ID type which is —Long—
The type of Object
Make interface of repository

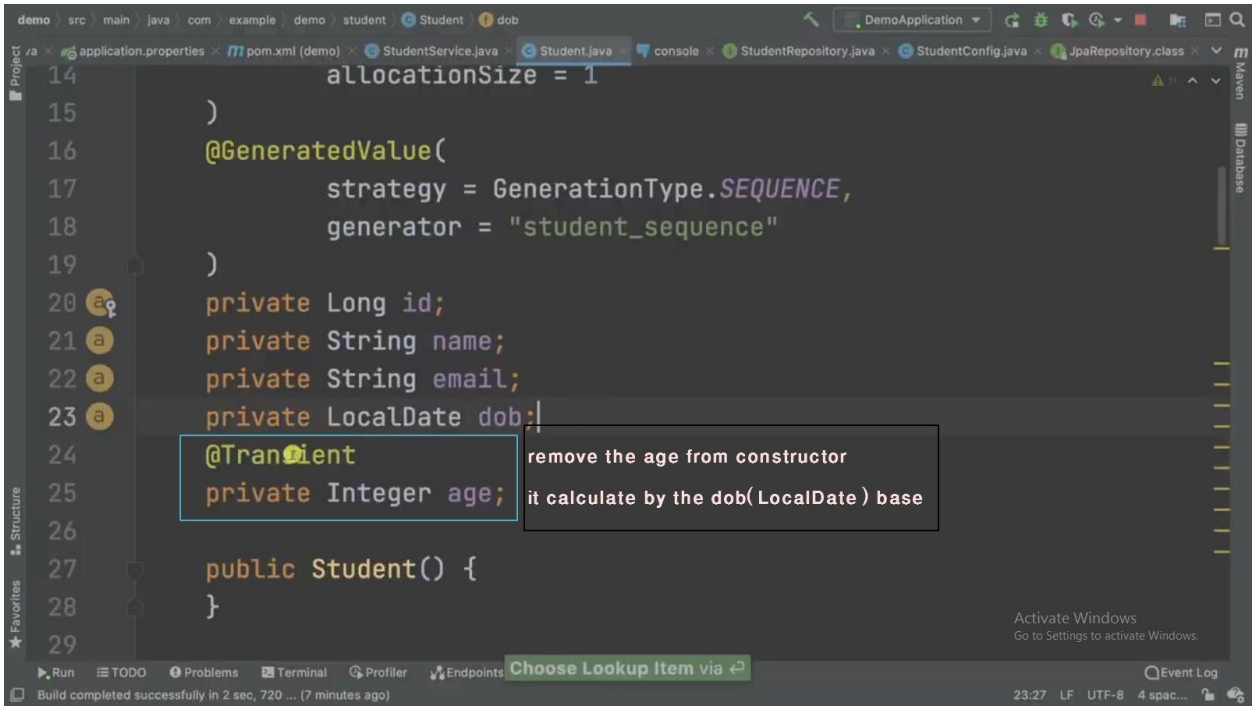

package com.example.demo.student;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
@Repository
Add this annotation
public interface StudentRepository
extends JpaRepository<Student, Long> {
}

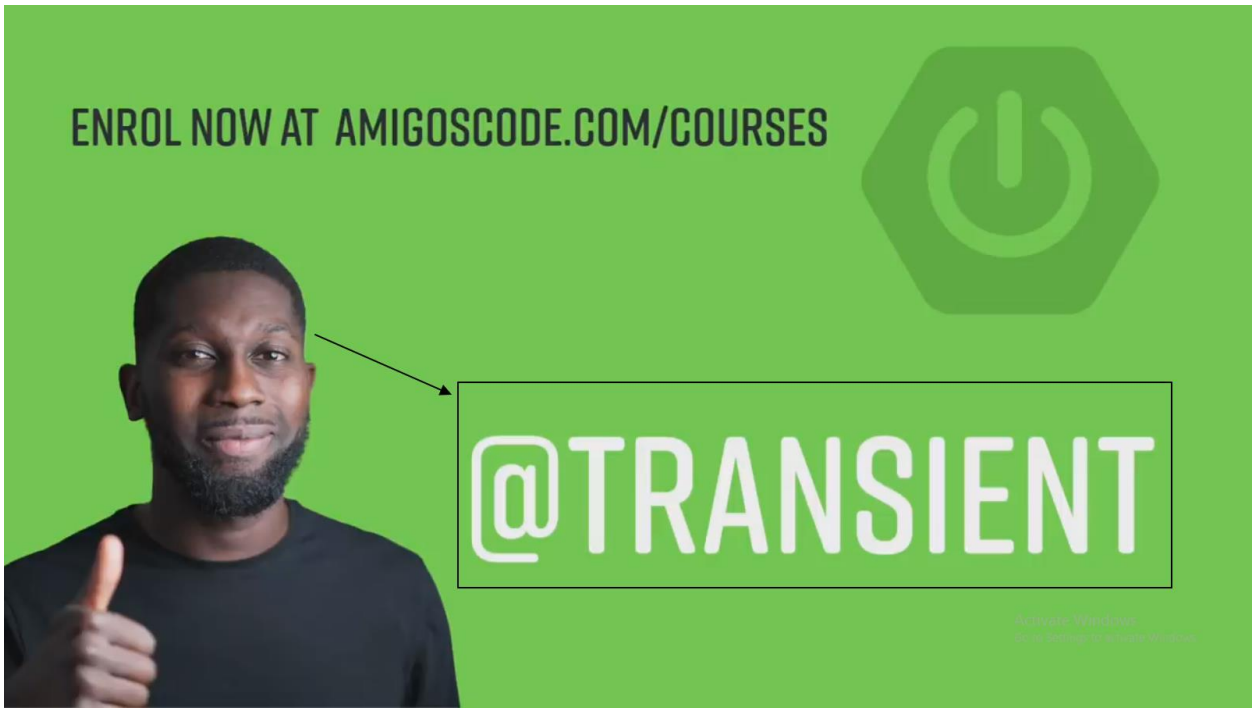

@Service
public class StudentService {
private final StudentRepository studentRepository;
@Autowired
public StudentService(StudentRepository studentRepository) {
this.studentRepository = studentRepository;
}
public List<Student> getStudents() {
return studentRepository.findAll();
}
}
Now we take data from DB
Activate Windows
Go to Settings to activate Windows.

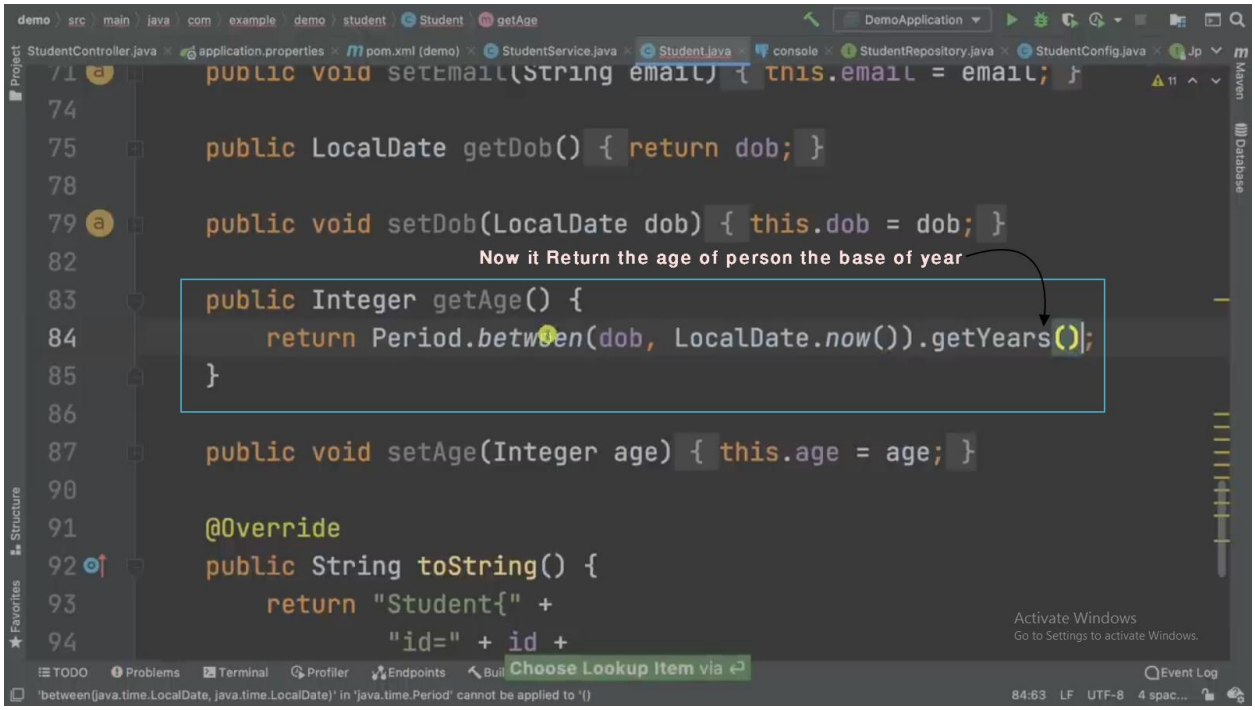

ENROL NOW AT AMIGOSCODE.COM/COURSES
SAVING STUDENTS

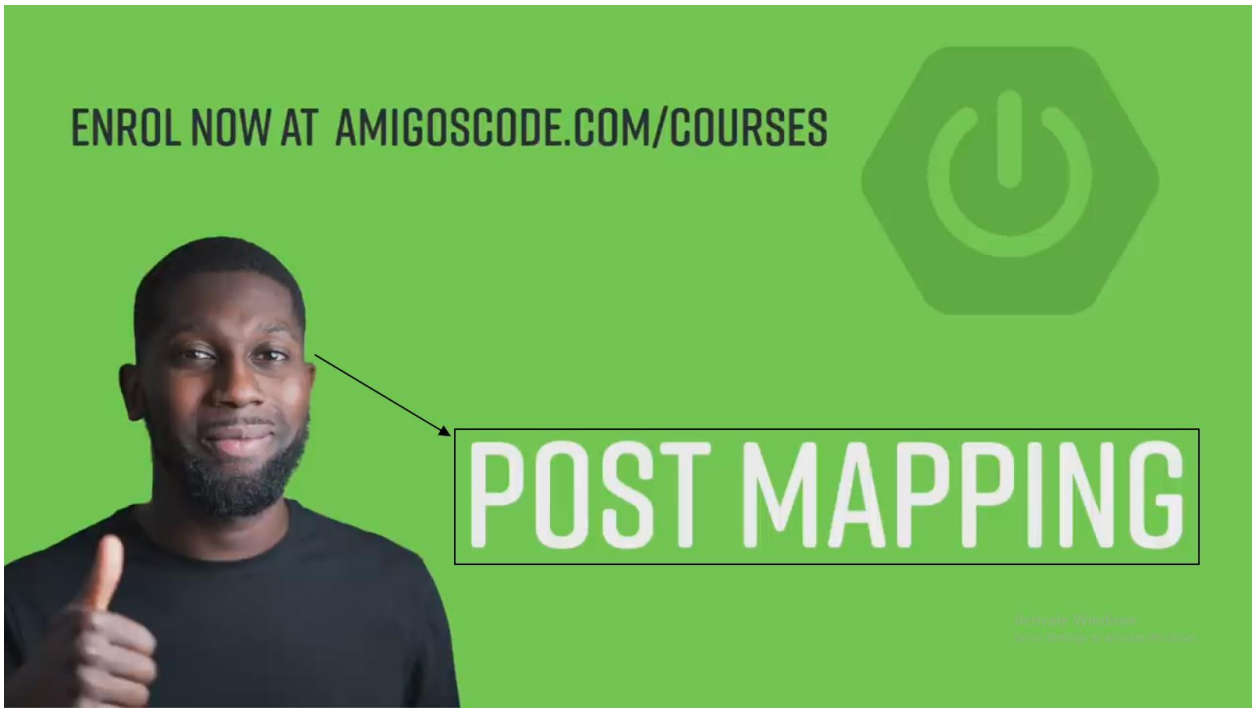

Create StudentConfigration Class _add the Data insid
public class StudentConfig {
@Bean
CommandLineRunner commandLineRunner(
StudentRepository repository) {
return args -> {
new Student(
name:"Mariam",
email:"mariam.jamal@gmail.com",
LocalDate.of(year:2000, JANUARY, dayOfMonth:5),
age:21
);
};
}
Activate Windows
Go to Settings to activate Windows.
Show Context Actions via ⌥⏎ (Alt+⏎ for Win/Linux)
Press Escape to remove the highlighting

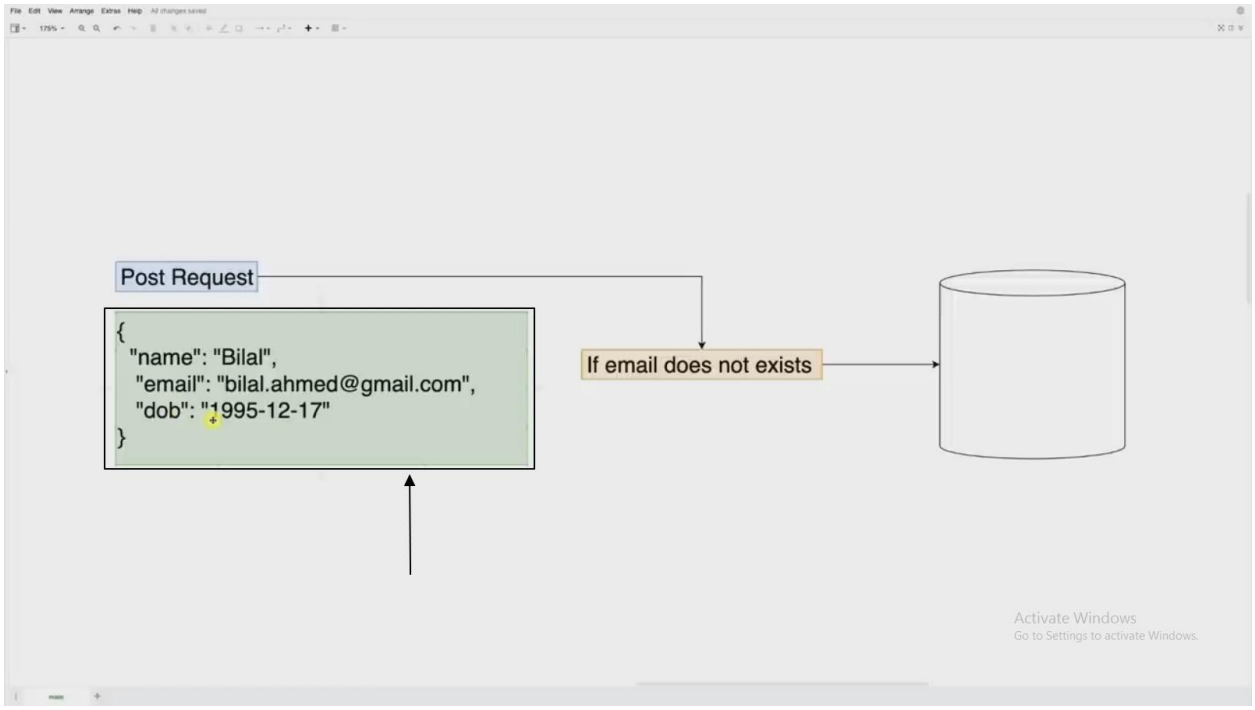

LocalDate.of(year:2000, JANUARY, dayOfMonth:5),
age:21
);
Student alex = new Student(
name:"Alex",
email:"alex@gmail.com",
LocalDate.of(year:2004, JANUARY, dayOfMonth:5),
age:21
);
Save the Marim And Alex data
repository.saveAll(
List.of(mariam, alex)
);
};
Now Run the Application to stor data inside the DB
Activate Windows
Go to Settings to activate Windows.
Build completed successfully in 2 sec, 919 ... (5 minutes ago)

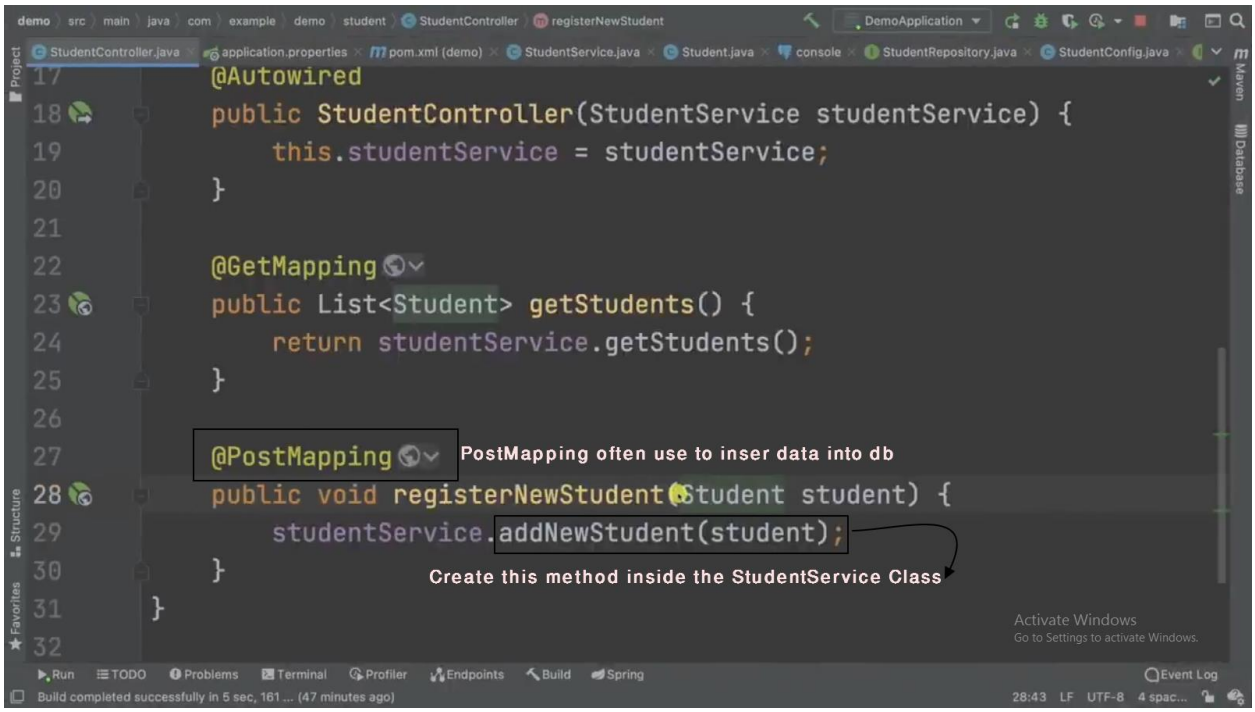

allocationSize = 1
)
@GeneratedValue(
strategy = GenerationType.SEQUENCE,
generator = "student_sequence"
)
private Long id;
private String name;
private String email;
private LocalDate dob;
@Transient
private Integer age;
public Student() {
}
remove the age from constructor
it calculate by the dob(LocalDate) base
Activate Windows
Go to Settings to activate Windows.
Choose Lookup Item via
Build completed successfully in 2 sec, 720 ... (7 minutes ago)

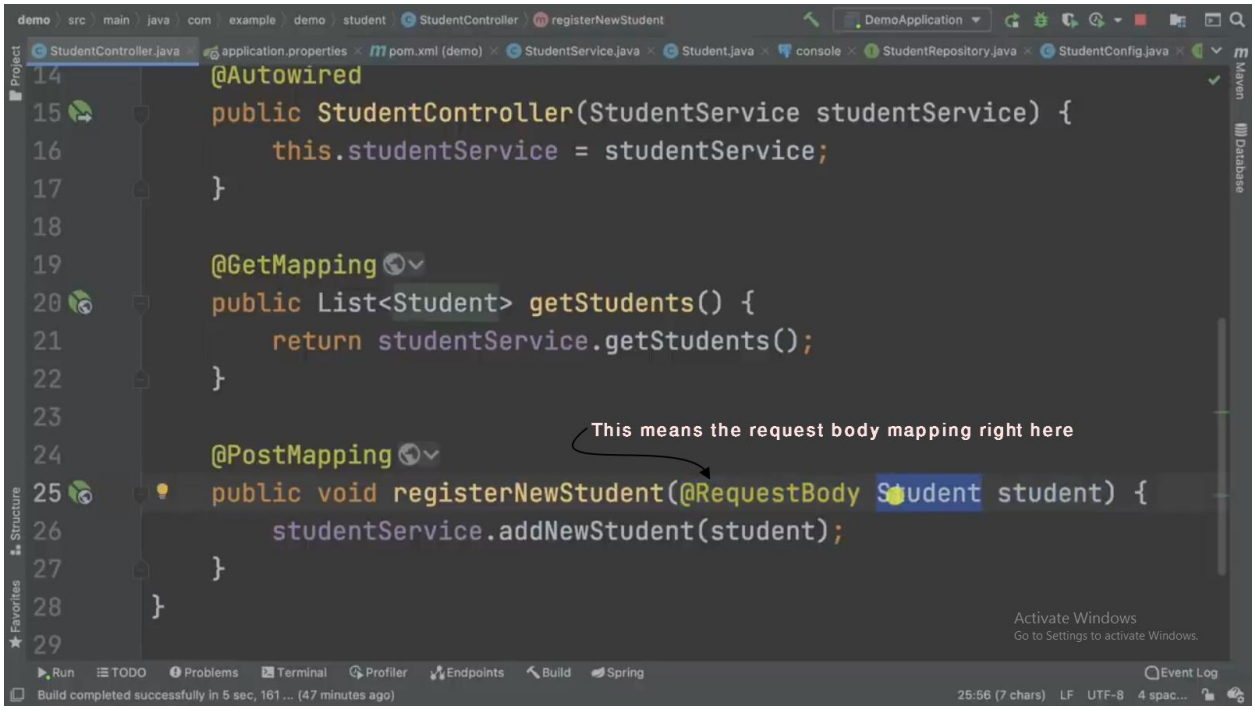

ENROL NOW AT AMIGOSCODE.COM/COURSES
@TRANSIENT


Now it Return the age of person the base of year
public LocalDate getDob() { return dob; }
public void setDob(LocalDate dob) { this.dob = dob; }
public Integer getAge() {
return Period.between(dob, LocalDate.now()).getYears();
}
public void setAge(Integer age) { this.age = age; }
@Override
public String toString() {
return "Student{" +
"id=" + id +
Activate Windows
Go to Settings to activate Windows.
Choose Lookup Item via
'between(java.time.LocalDate, java.time.LocalDate)' in 'java.time.Period' cannot be applied to '()

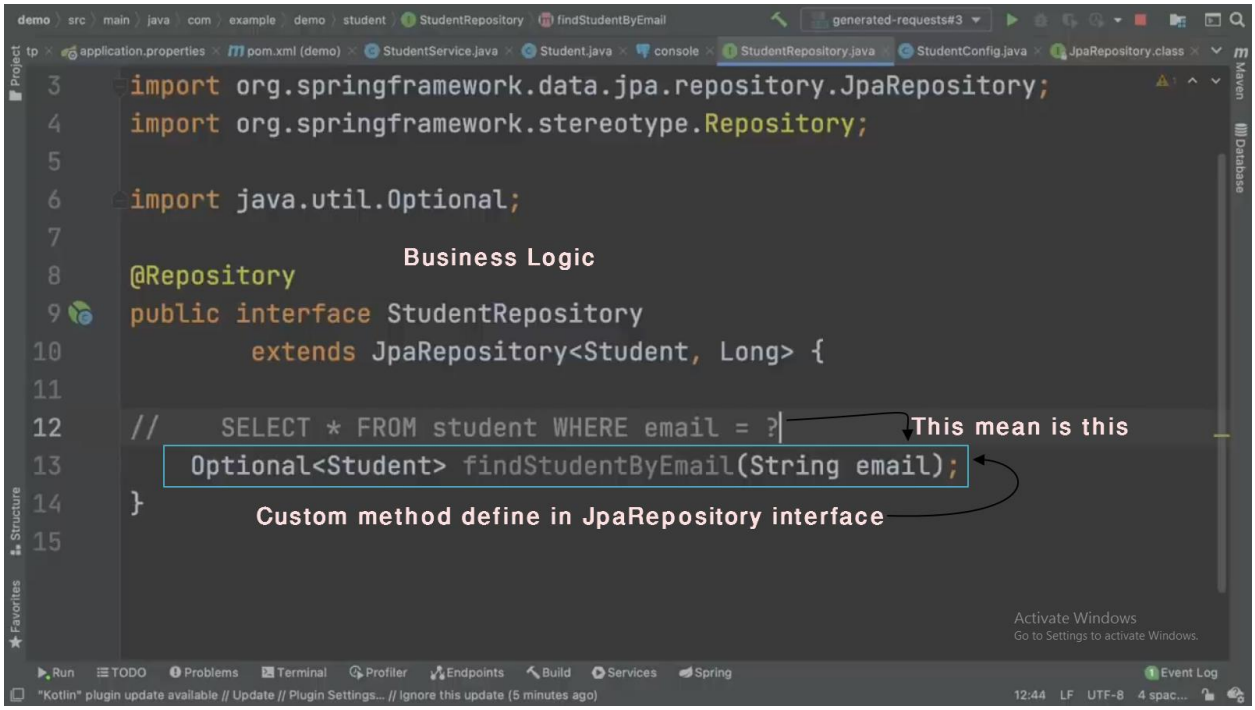

ENROL NOW AT AMIGOSCODE.COM/COURSES
POST MAPPING

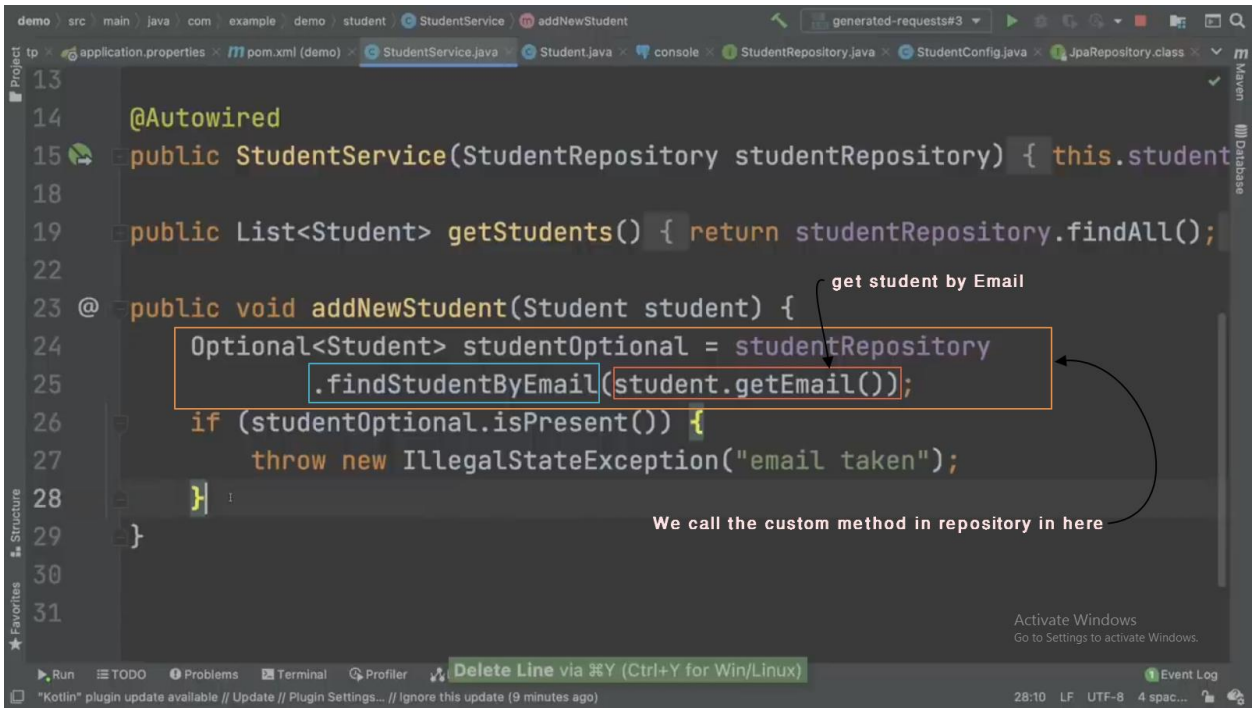

Post Request
{
"name": "Bilal",
"email": "bilal.ahmed@gmail.com",
"dob": "1995-12-17"
}
If email does not exists
Activate Windows
Go to Settings to activate Windows.

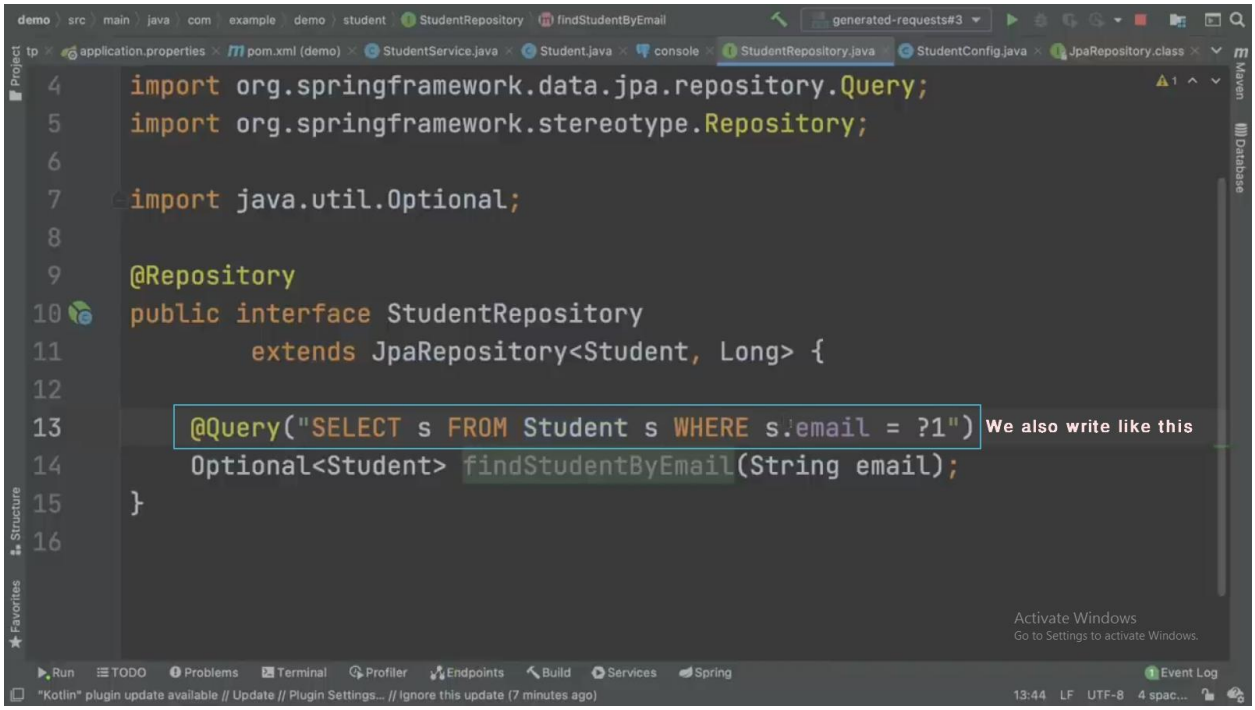

@Autowired
public StudentController(StudentService studentService) {
this.studentService = studentService;
}
@GetMapping
public List<Student> getStudents() {
return studentService.getStudents();
}
@PostMapping
PostMapping often use to inser data into db
public void registerNewStudent(Student student) {
studentService.addNewStudent(student);
}
Create this method inside the StudentService Class
}
Activate Windows
Go to Settings to activate Windows.

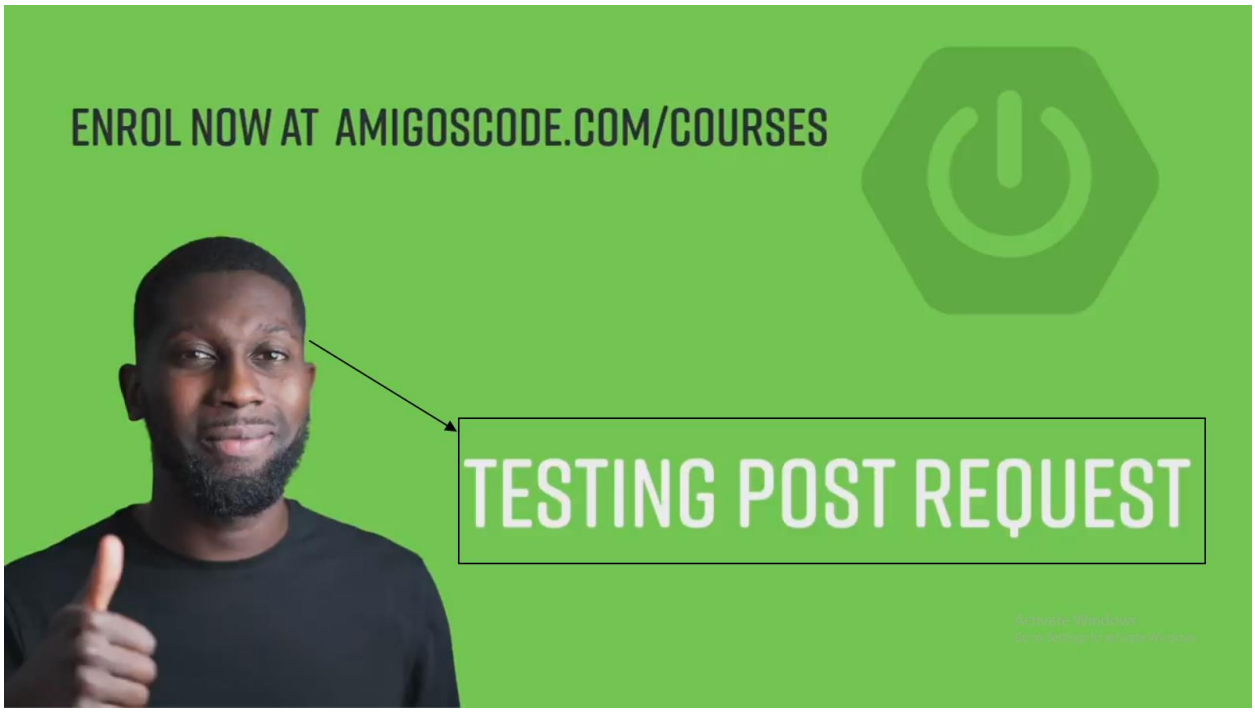

@Autowired
public StudentController(StudentService studentService) {
this.studentService = studentService;
}
@GetMapping
public List<Student> getStudents() {
return studentService.getStudents();
}
This means the request body mapping right here
@PostMapping
public void registerNewStudent(@RequestBody Student student) {
studentService.addNewStudent(student);
}
}
Activate Windows
Go to Settings to activate Windows.
Build completed successfully in 5 sec, 161 ... (47 minutes ago)


ENROL NOW AT AMIGOSCODE.COM/COURSES
WRITING BUSINESS LOGIC


import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import java.util.Optional;
Business Logic
@Repository
public interface StudentRepository
extends JpaRepository<Student, Long> {
// SELECT * FROM student WHERE email = ?
This mean is this
Optional<Student> findStudentByEmail(String email);
}
Custom method define in JpaRepository interface


@Autowired
public StudentService(StudentRepository studentRepository) { this.student
public List<Student> getStudents() { return studentRepository.findAll();
get student by Email
public void addNewStudent(Student student) {
Optional<Student> studentOptional = studentRepository
.findStudentByEmail(student.getEmail());
if (studentOptional.isPresent()) {
throw new IllegalStateException("email taken");
}
}
We call the custom method in repository in here

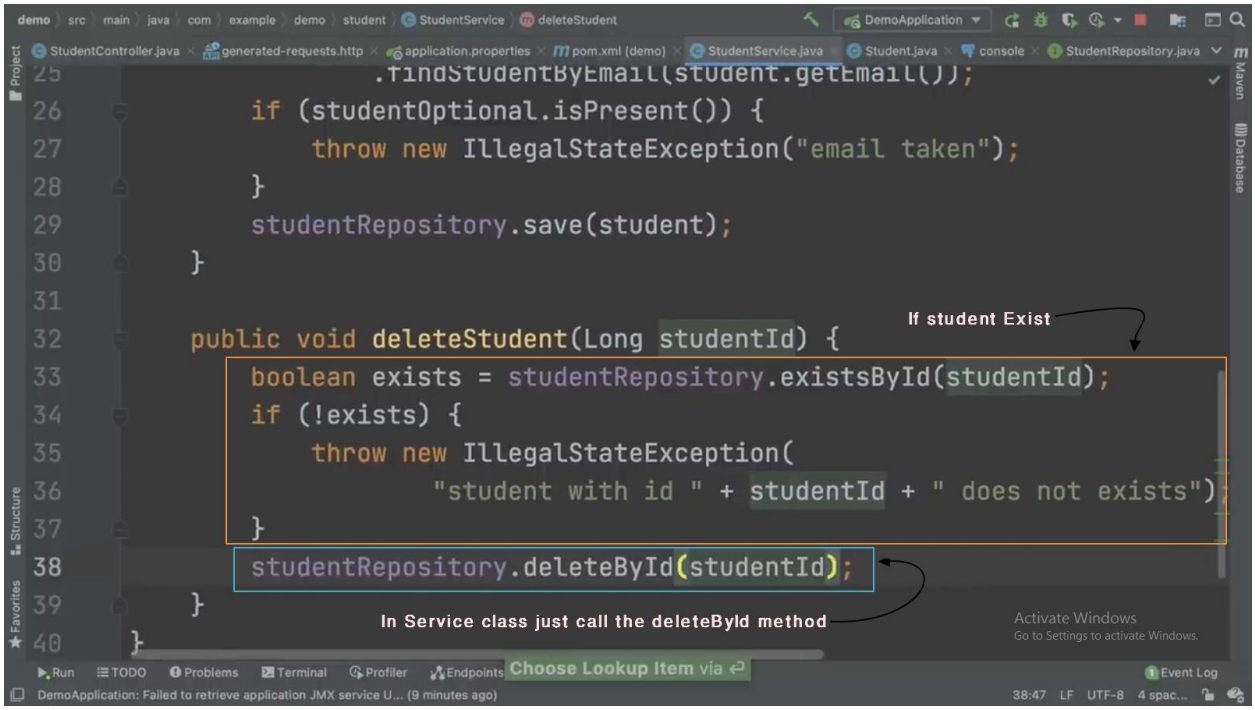

demo src main java com example demo student StudentRepository findStudentByEmail
generated-requests#3
application.properties pom.xml (demo) StudentService.java Student.java console StudentRepository.java StudentConfig.java JpaRepository.class
import org.springframework.data.jpa.repository.Query;
import org.springframework.stereotype.Repository;
import java.util.Optional;
@Repository
public interface StudentRepository
extends JpaRepository<Student, Long> {
@Query("SELECT s FROM Student s WHERE s.email = ?1")
Optional<Student> findStudentByEmail(String email);
}
We also write like this
Activate Windows
Go to Settings to activate Windows.
Run TODO Problems Terminal Profiler Endpoints Build Services Spring
Event Log
"Kotlin" plugin update available // Update // Plugin Settings... // Ignore this update (7 minutes ago)
13:44 LF UTF-8 4 spac...

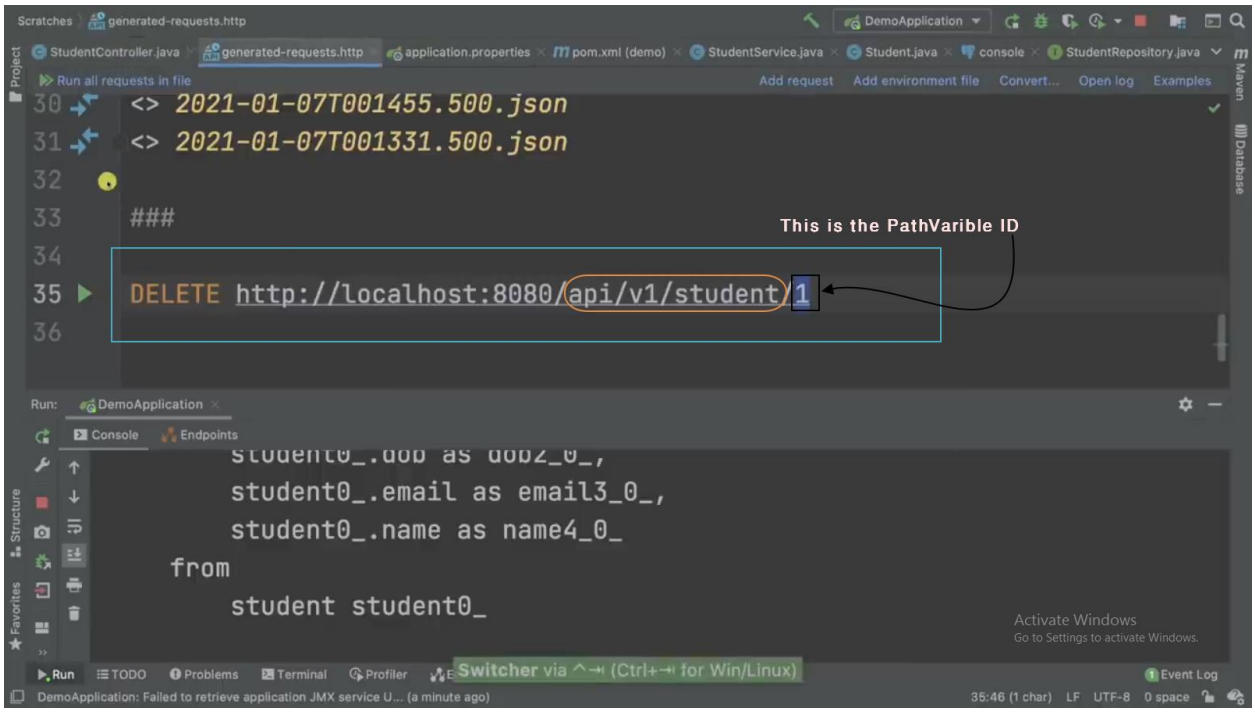

ENROL NOW AT AMIGOSCODE.COM/COURSES
TESTING POST REQUEST


```
21          return studentService.getStudents();
22      }
23
24      @PostMapping
25      public void registerNewStudent(@RequestBody Student student) {
26          studentService.addNewStudent(student);
27      }
28
29      @DeleteMapping(path = "{studentId}")
30      public void deleteStudent(@PathVariable("studentId") Long id) {
31          studentService.deleteStudent(id);
32      }
33  }
34
```

Match this student id to the below student id

Create deleteStudent Method iniside the Service

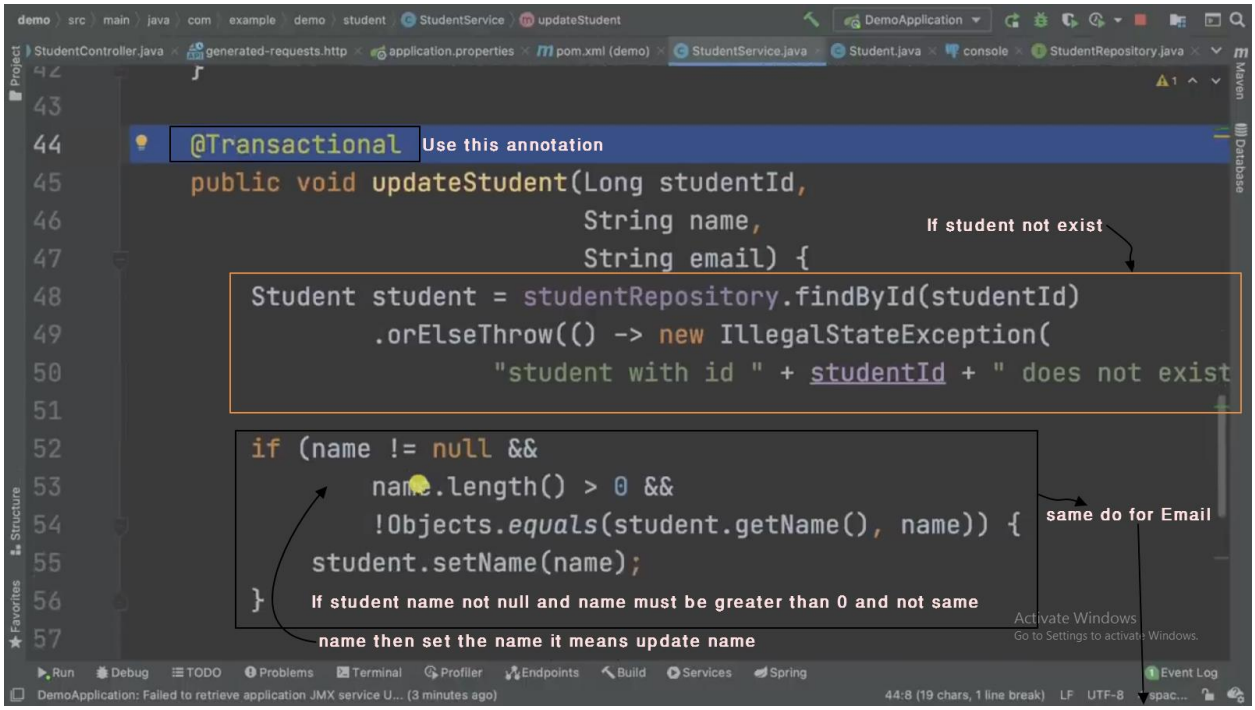

.findStudentByEmail(student.getEmail());
if (studentOptional.isPresent()) {
throw new IllegalStateException("email taken");
}
studentRepository.save(student);
}
public void deleteStudent(Long studentId) {
boolean exists = studentRepository.existsById(studentId);
if (!exists) {
throw new IllegalStateException(
"student with id " + studentId + " does not exists");
}
studentRepository.deleteById(studentId);
}
}
If student Exist
In Service class just call the deleteByld method

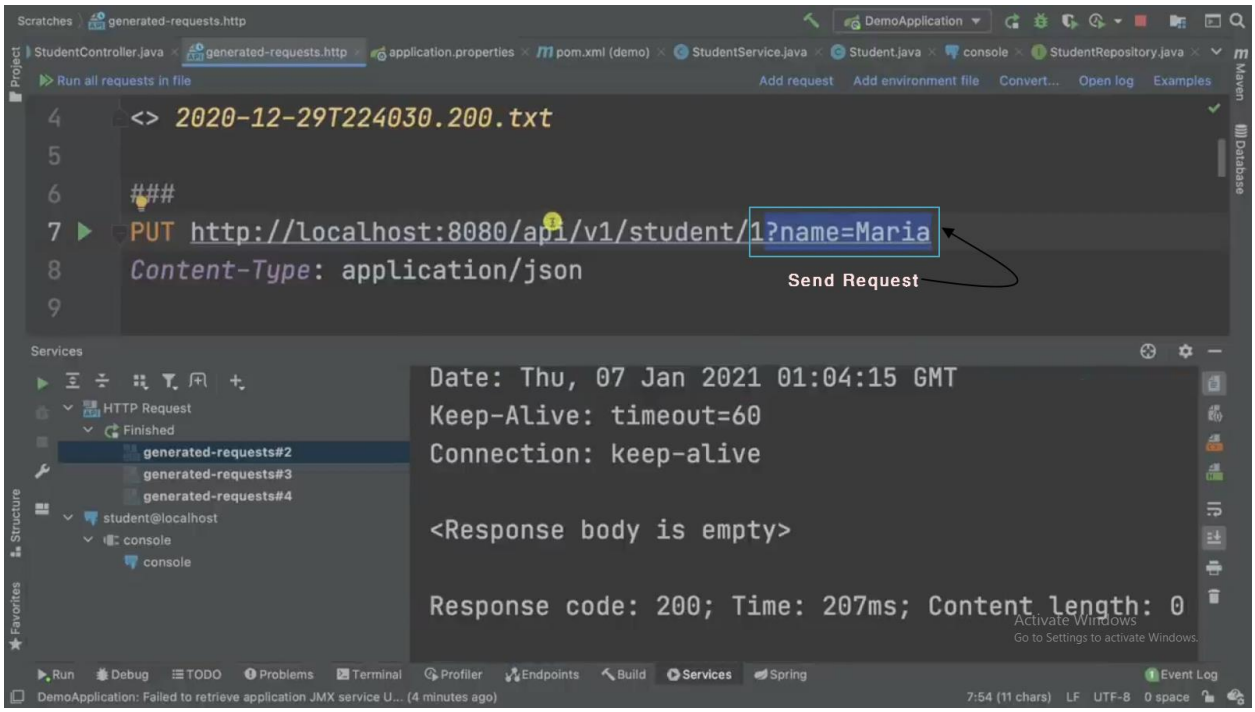


DELETE http://localhost:8080/api/v1/student/1
This is the PathVarible ID
student0_.email as email3_0_,
student0_.name as name4_0_
from
student student0_


ENROL NOW AT AMIGOSCODE.COM/COURSES
EXERCISE

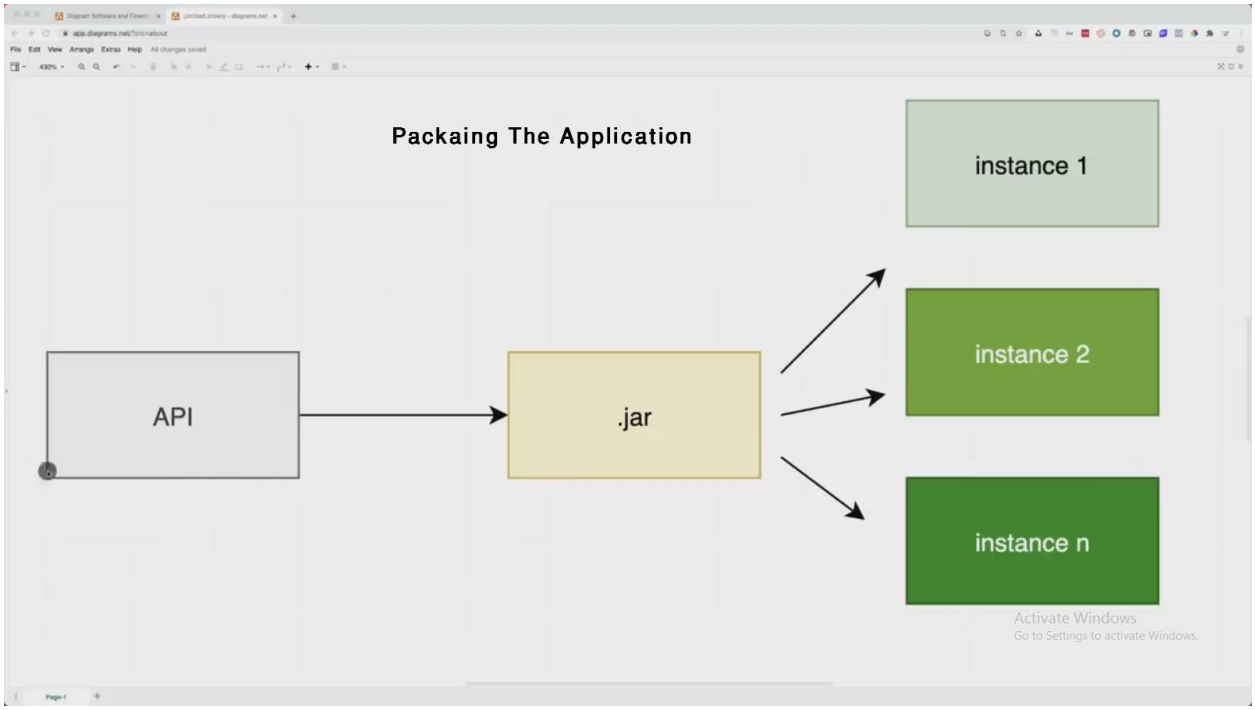

ENROL NOW AT AMIGOSCODE.COM/COURSES
SOLUTION

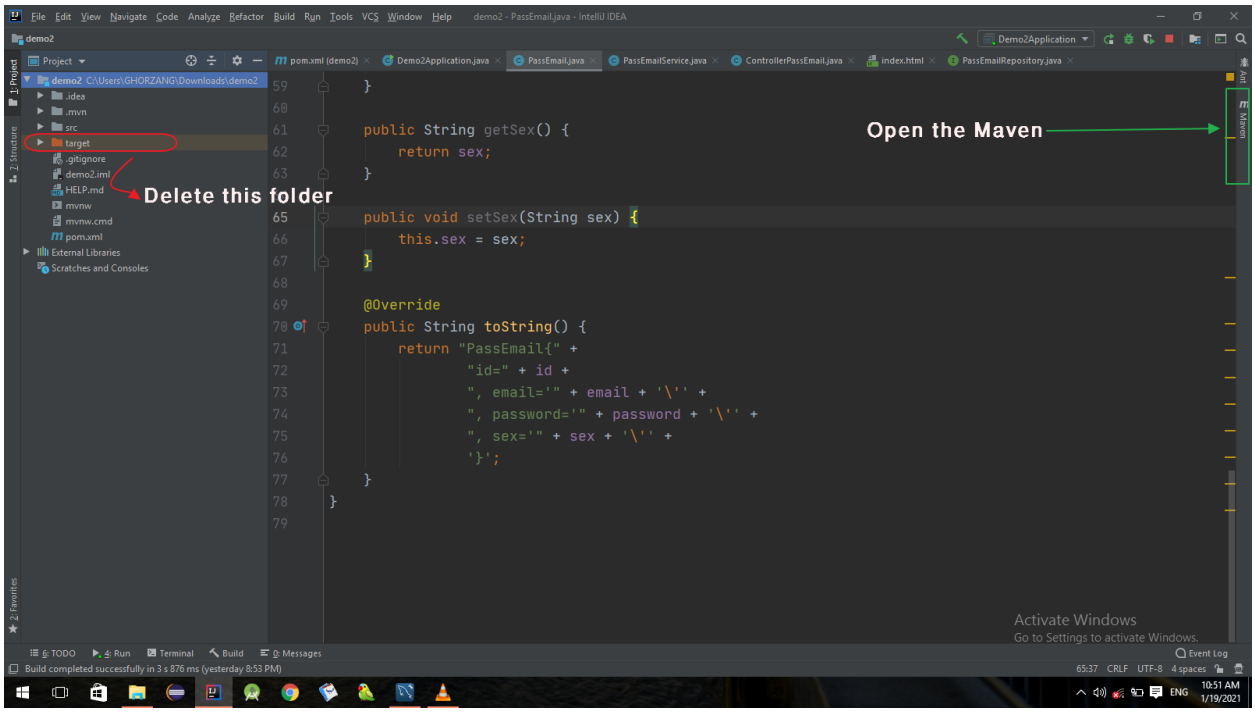

@Transactional
Use this annotation
public void updateStudent(Long studentId,
String name,
String email) {
If student not exist
Student student = studentRepository.findById(studentId)
.orElseThrow(() -> new IllegalStateException(
"student with id " + studentId + " does not exist
if (name != null &&
name.length() > 0 &&
!Objects.equals(student.getName(), name)) {
same do for Email
student.setName(name);
}
If student name not null and name must be greater than 0 and not same
name then set the name it means update name

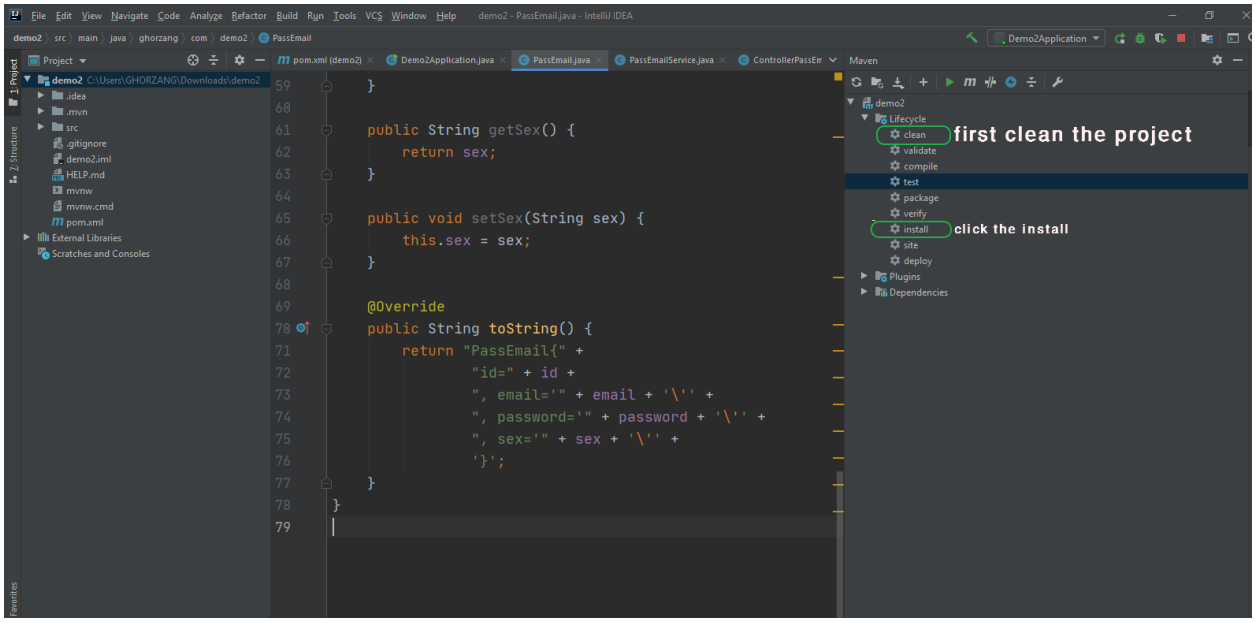

<> 2020-12-29T224030.200.txt
###
PUT http://localhost:8080/api/v1/student/1?name=Maria
Content-Type: application/json
Send Request
Date: Thu, 07 Jan 2021 01:04:15 GMT
Keep-Alive: timeout=60
Connection: keep-alive
<Response body is empty>
Response code: 200; Time: 207ms; Content length: 0


ENROL NOW AT AMIGOSCODE.COM/COURSES
TESTING


ENROL NOW AT AMIGOSCODE.COM/COURSES
PACKAGING AND RUNNING APPLICATION


Packaing The Application
instance 1
instance 2
instance n
API
.jar
Activate Windows
Go to Settings to activate Windows.


Delete this folder
Open the Maven
Activate Windows
Go to Settings to activate Windows.


first clean the project
click the install


ENROL NOW AT AMIGOSCODE.COM/COURSES
NEXT STEPS