**WEEK - 3 : HANDS-ON EXERCISE**

**Spring Data JPA with Spring Boot, Hibernate**

1. **Spring Data JPA - Quick Example**

To build a REST API using Spring Boot and Spring Data JPA that performs basic operations (Add and View) on a Student entity using an H2 in-memory database.

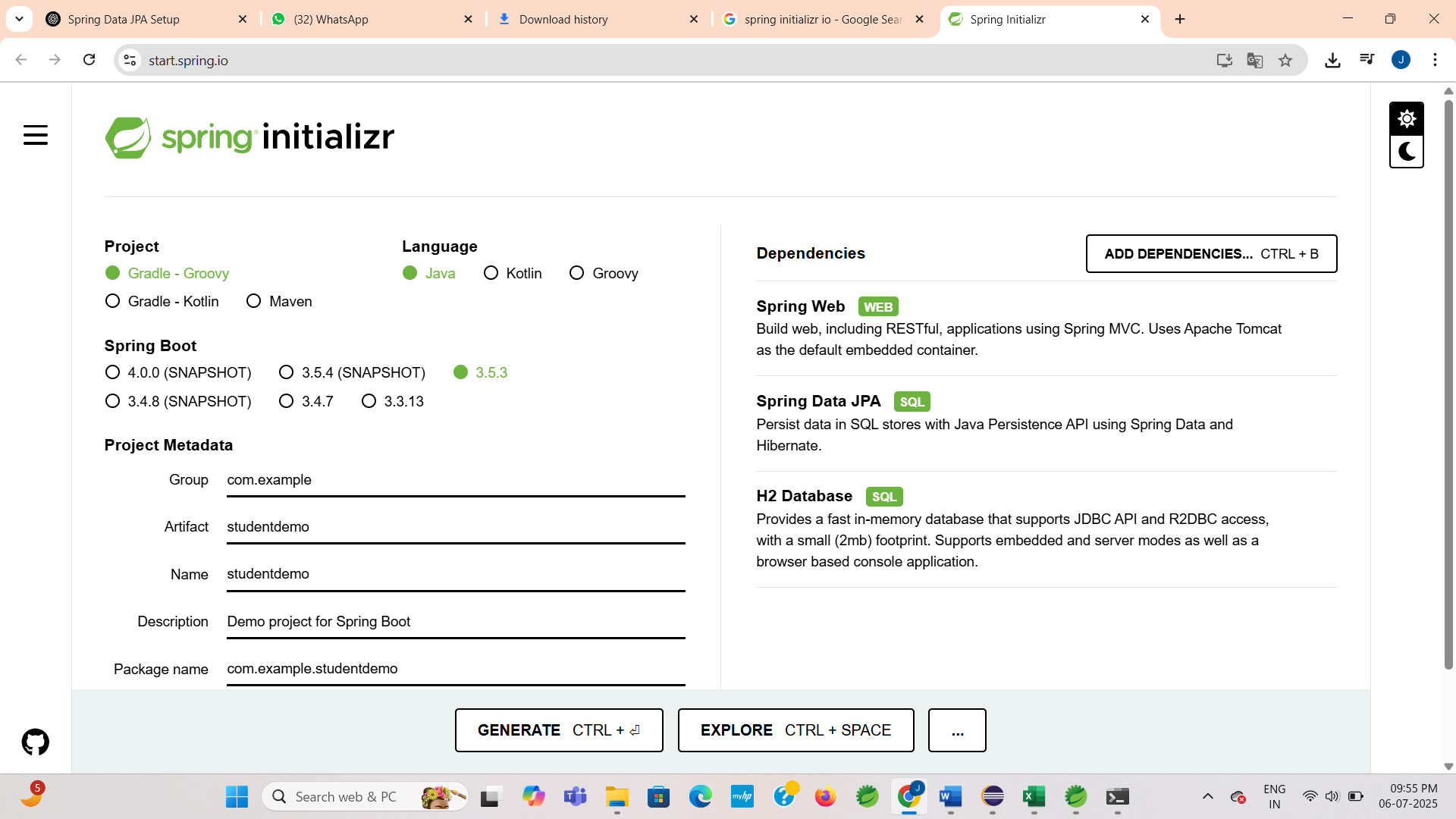
**1.Project Setup – Spring Initializr**

Project was generated using <https://start.spring.io>

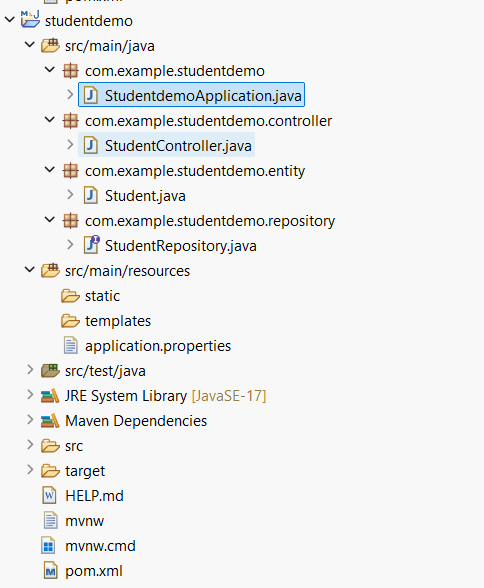
**Configuration Used:**

* Project: Maven
* Language: Java
* Spring Boot: Default
* Group: com.example
* Artifact: studentdemo
* Name: studentdemo
* Dependencies:
  + Spring Web
  + Spring Data JPA
  + H2 Database

After generation, the project was imported into Eclipse.

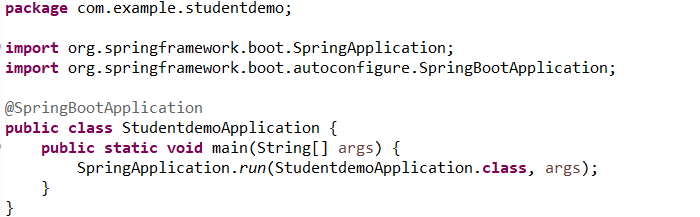


**2. Project Structure – Eclipse**

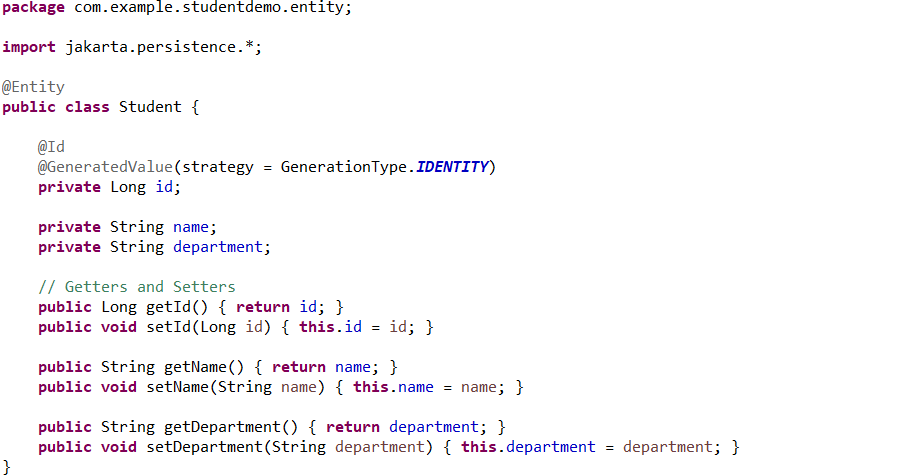
****

**3. Code:**

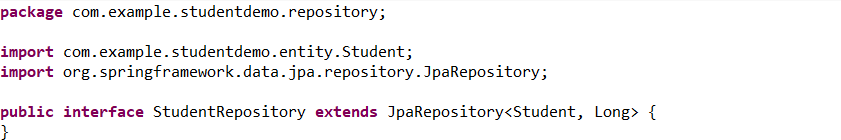
**3.1. StudentdemoApplication.java**

****

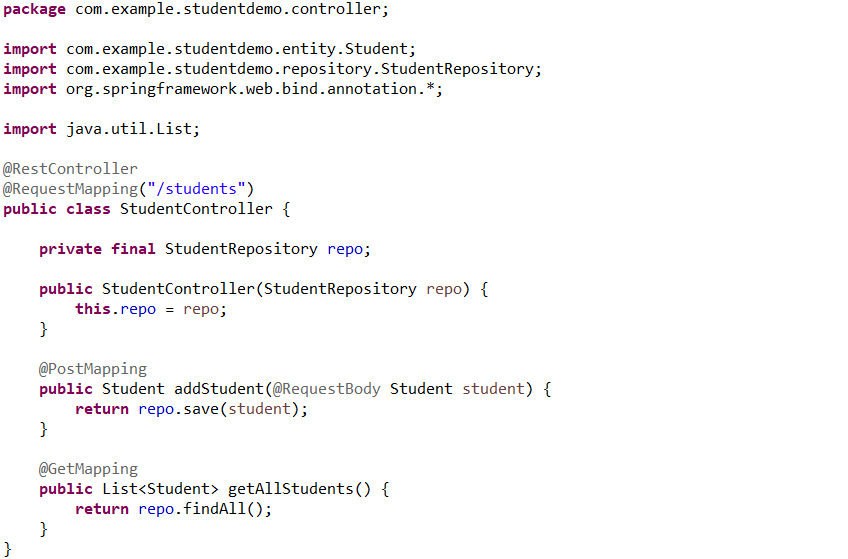
**3.2. Student.java (Entity class)**



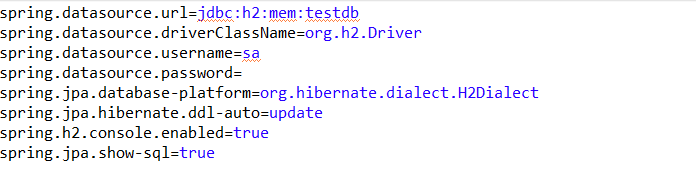
**3.3. StudentRepository.java**



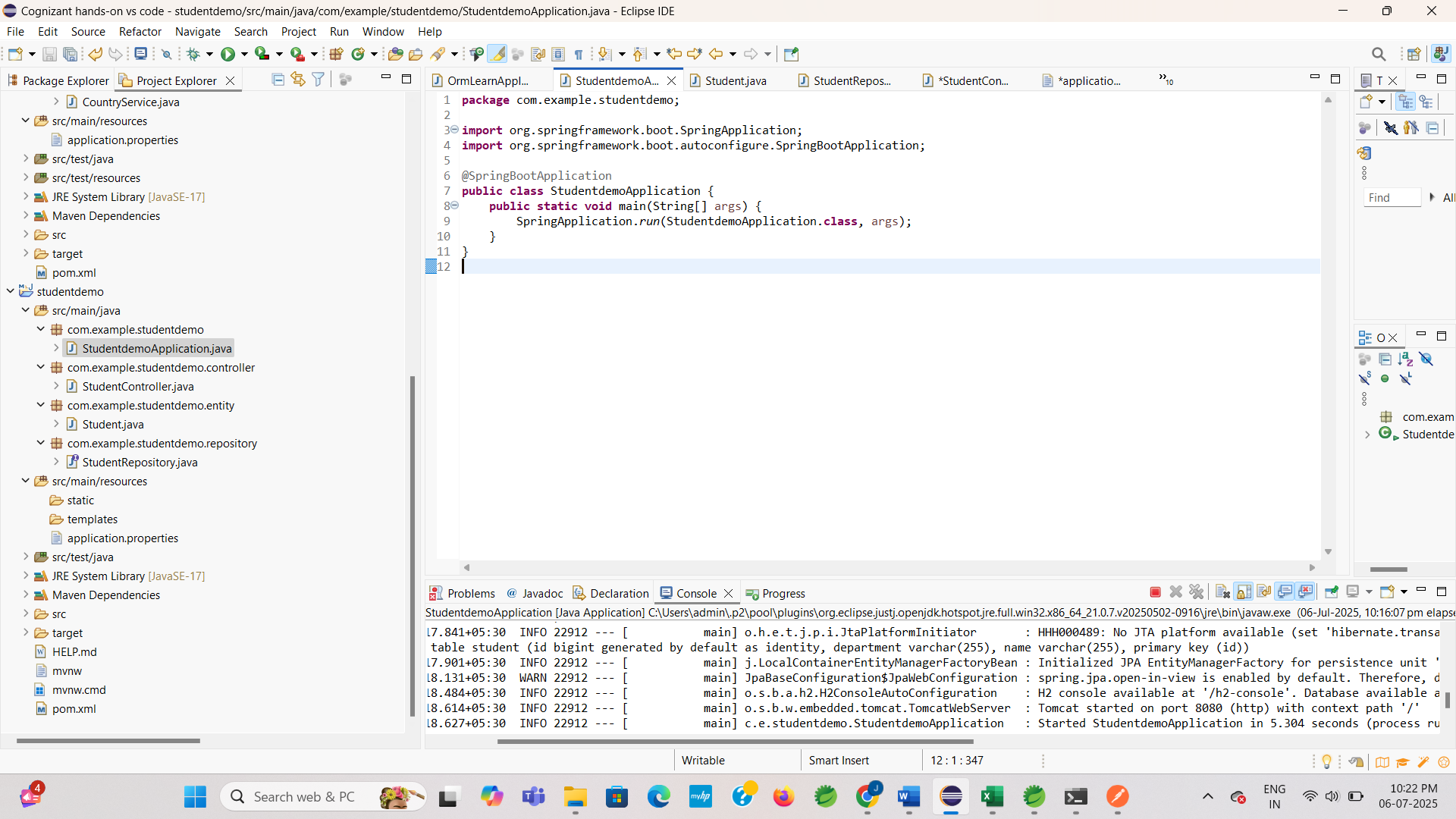
**3.4. StudentController.java**

****

**3.5. application.properties**

**4. Running the Application**

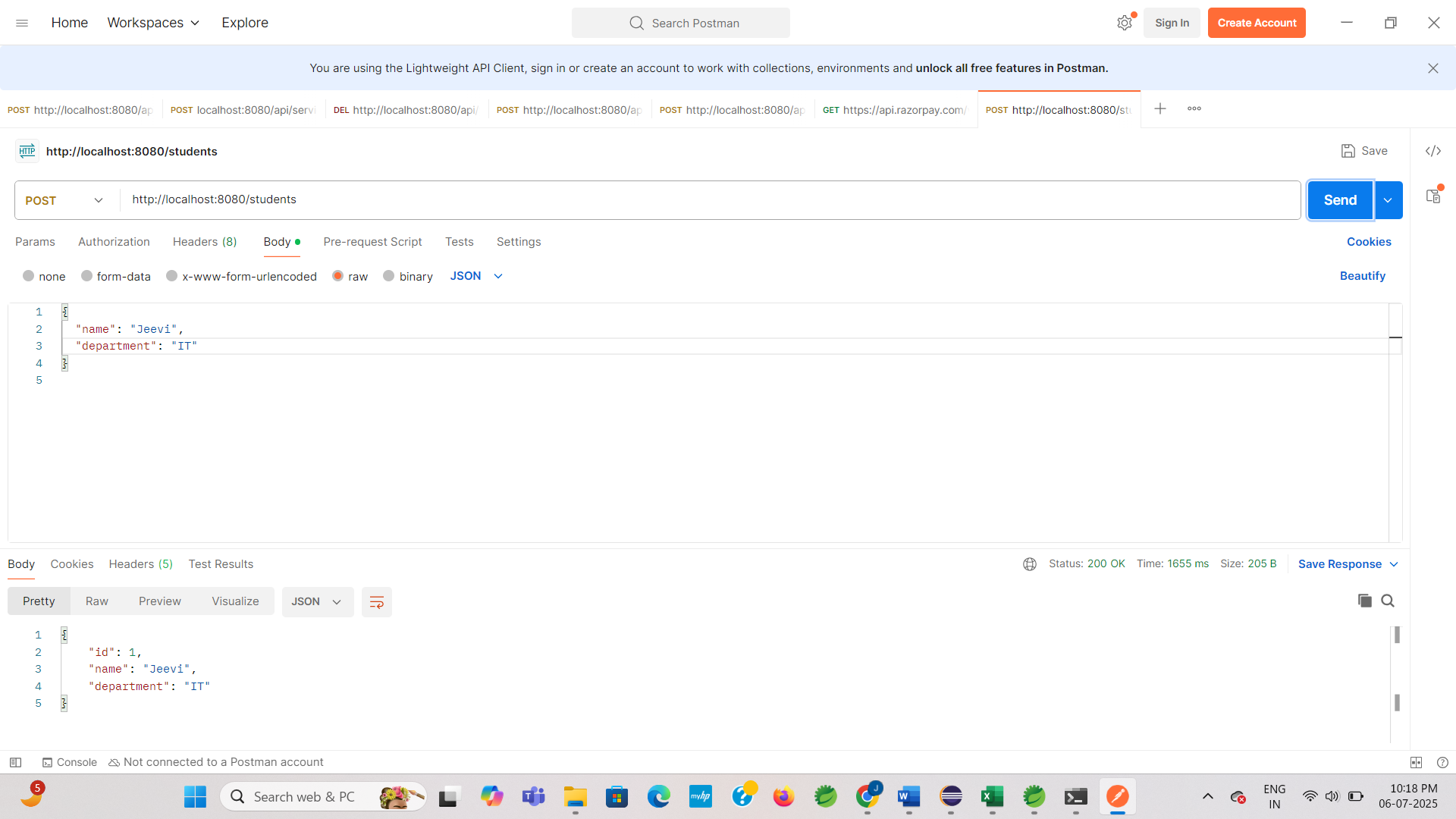
1. Open Eclipse
2. Right-click on StudentdemoApplication.java
3. Choose Run As → Spring Boot App
4. Console output will show:  
   *Started StudentdemoApplication*



**5. API Testing Using Postman**

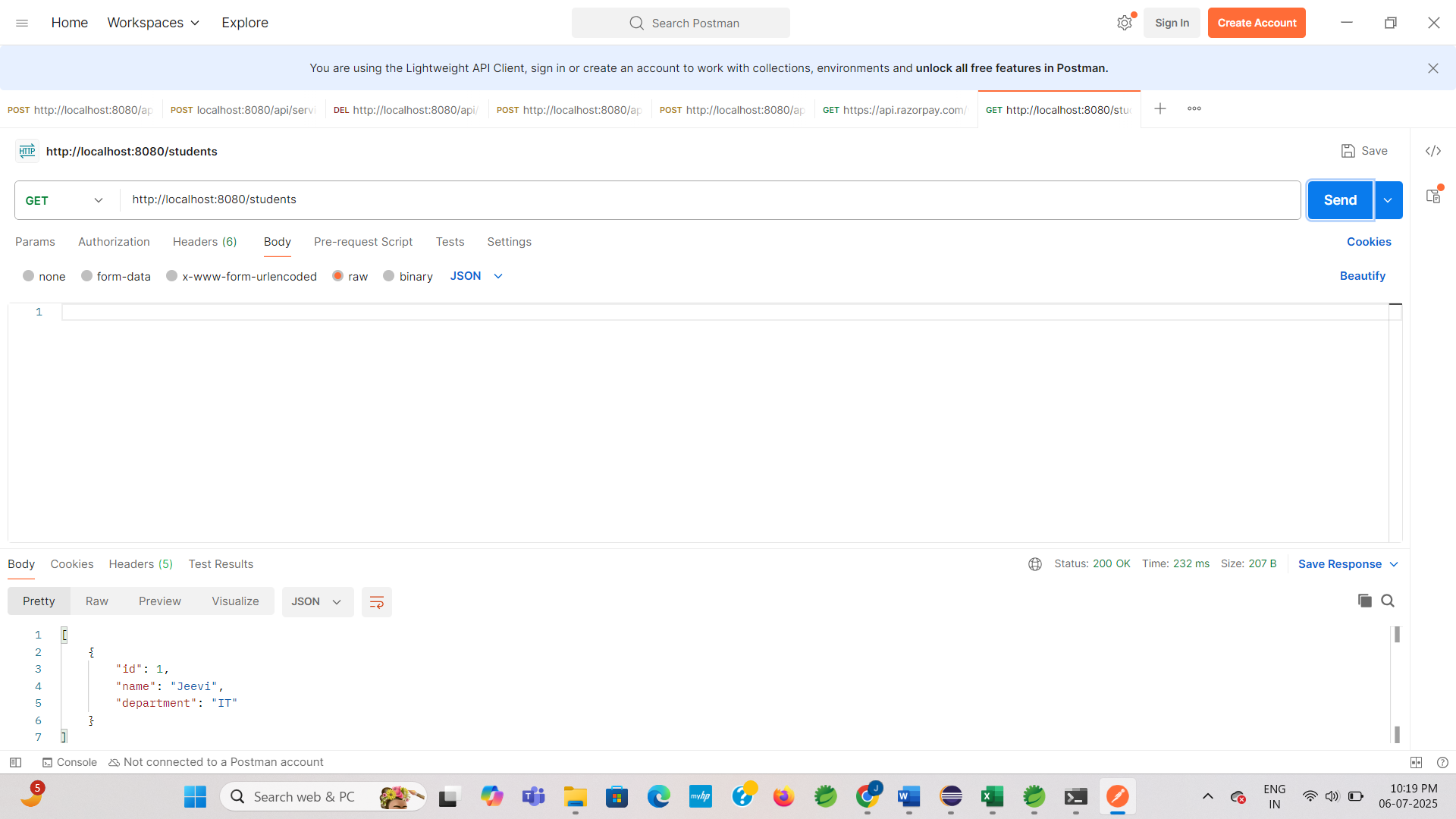
**5.1. Add Student (POST)**

* URL: <http://localhost:8080/students>
* Method: POST



**5.2. Get All Students (GET)**

* URL: <http://localhost:8080/students>
* Method: GET



1. **Difference between JPA, Hibernate and Spring Data JPA**
2. **JPA (Java Persistence API)**

* JPA is a specification, not a tool or framework.
* It defines a standard for mapping Java objects (entities) to relational databases.
* It includes annotations like @Entity, @Id, @OneToMany, etc.
* JPA only provides interfaces — it does not contain any working code.
* It requires a provider (like Hibernate) to actually perform the database operations.

1. **Hibernate**

* Hibernate is a JPA implementation — it provides the working code based on JPA's rules.
* It allows Java applications to interact with the database using objects.
* It supports all JPA features and also includes additional features, such as:
  + Lazy loading
  + Caching
  + Custom HQL (Hibernate Query Language)
* You can use Hibernate directly, or through JPA interfaces.

1. **Spring Data JPA**

* Spring Data JPA is a module in the Spring ecosystem that simplifies JPA-based data access.
* It builds on top of JPA and uses a JPA provider (like Hibernate) under the hood.
* It reduces the amount of boilerplate code needed for repositories.
* Provides features like:
  + Auto-generated queries based on method names
  + Paging and sorting
  + Transaction management
* With Spring Data JPA, developers can create repository interfaces only, and Spring will auto-generate the implementation.

**In Simple Terms**

* **JPA** – Defines the rules (What should be done)
* **Hibernate** – Implements the rules (How it is done)
* **Spring Data JPA** – Makes it easier to use (Done automatically)