# Spring Data JPA with Spring Boot and Hibernate

Exercise 1: Spring Data JPA - Quick Example  
  
Objective:  
Set up a basic Spring Boot application using Spring Data JPA and Hibernate to perform CRUD operations.  
  
Steps:  
  
1. Create Spring Boot Application (using Spring Initializr):  
 - Dependencies: Spring Web, Spring Data JPA, H2 Database  
  
2. Configure application.properties:  
  
spring.datasource.url=jdbc:h2:mem:testdb  
spring.datasource.driverClassName=org.h2.Driver  
spring.datasource.username=sa  
spring.datasource.password=  
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect  
spring.h2.console.enabled=true  
  
3. Create Entity Class:  
  
import jakarta.persistence.Entity;  
import jakarta.persistence.GeneratedValue;  
import jakarta.persistence.Id;  
  
@Entity  
public class Student {  
 @Id  
 @GeneratedValue  
 private Long id;  
 private String name;  
 private int age;  
  
 // Getters and Setters  
}  
  
4. Create Repository Interface:  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface StudentRepository extends JpaRepository<Student, Long> {  
}  
  
5. Create Controller Class:  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/students")  
public class StudentController {  
  
 @Autowired  
 private StudentRepository repository;  
  
 @PostMapping  
 public Student addStudent(@RequestBody Student student) {  
 return repository.save(student);  
 }  
  
 @GetMapping  
 public List<Student> getAllStudents() {  
 return repository.findAll();  
 }  
}  
  
6. Run the application and test using Postman or H2 console.  
  
  
Exercise 2: Difference between JPA, Hibernate and Spring Data JPA  
  
Objective:  
Understand how JPA, Hibernate, and Spring Data JPA are related and different.  
  
- JPA (Java Persistence API):  
 \* Specification (interface) for object-relational mapping in Java.  
 \* Does not provide implementation (only API).  
 \* Example: `@Entity`, `@Id`, `EntityManager`.  
  
- Hibernate:  
 \* One of the most popular implementations of the JPA specification.  
 \* Offers additional features like caching, lazy/eager fetching strategies.  
 \* Can be used standalone or via Spring.  
  
- Spring Data JPA:  
 \* Built on top of JPA and Hibernate.  
 \* Reduces boilerplate code for DAO layer.  
 \* Provides methods like `save()`, `findAll()` out of the box through interfaces like `JpaRepository`.  
  
Summary:  
- JPA: Interface/specification.  
- Hibernate: Implementation of JPA.  
- Spring Data JPA: Abstraction over JPA with additional convenience features.