1.Spring Data JPA

Spring Data JPA is a part of the Spring Data family that makes it easier to work with Java Persistence API (JPA). It provides a set of abstractions and tools to simplify database operations in Spring applications.

JPA (Java Persistence API)	-	A standard Java API for object-relational mapping (ORM).
Hibernate	-	The most commonly used JPA provider
Spring Data JPA	-	A Spring module that simplifies CRUD

Reduces Boilerplate Code:

- No need to write complex DAO/Repository implementations.
- Just create a simple interface that extends JpaRepository.

Auto Implementation:

```
public interface BookRepository extends JpaRepository<Book, Long> {}
```

1. **Add Dependencies in pom.xml**

```
          <dependencies>
  <dependency>
     <groupId>org.springframework.boot</groupId>
     <artifactId>spring-boot-starter-data-jpa</artifactId>
```

```xml
    </dependency>


    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>


    <dependency>
        <groupId>com.h2database</groupId>
        <artifactId>h2</artifactId>
        <scope>runtime</scope>
    </dependency>
</dependencies>
```

2. Configure application.properties

```
spring.datasource.url=jdbc:h2:mem:librarydb
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
spring.jpa.hibernate.ddl-auto=update
```

3. Create an Entity Class

```java
import jakarta.persistence.*;

@Entity
```

```
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String title;
    private String author;


    public Book() {}
    public Book(String title, String author) {
        this.title = title;
        this.author = author;
    }


}
```

4. Create a Repository Interface

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

public interface BookRepository extends JpaRepository<Book,
Long> {
}
```

5. Create a REST Controller

```
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/books")
public class BookController {
    @Autowired
    private BookRepository bookRepository;

    @PostMapping
    public Book createBook(@RequestBody Book book) {
        return bookRepository.save(book);
    }

    @GetMapping
    public List<Book> getAllBooks() {
        return bookRepository.findAll();
    }
}
```

Testing

- Run the app.
- Access H2 Console: http://localhost:8080/h2-console
- Use JDBC URL: jdbc:h2:mem:librarydb
- Use Postman or curl:

  POST /books with JSON body {"title": "Spring JPA", "author": "John"}

  GET /books to see all entries