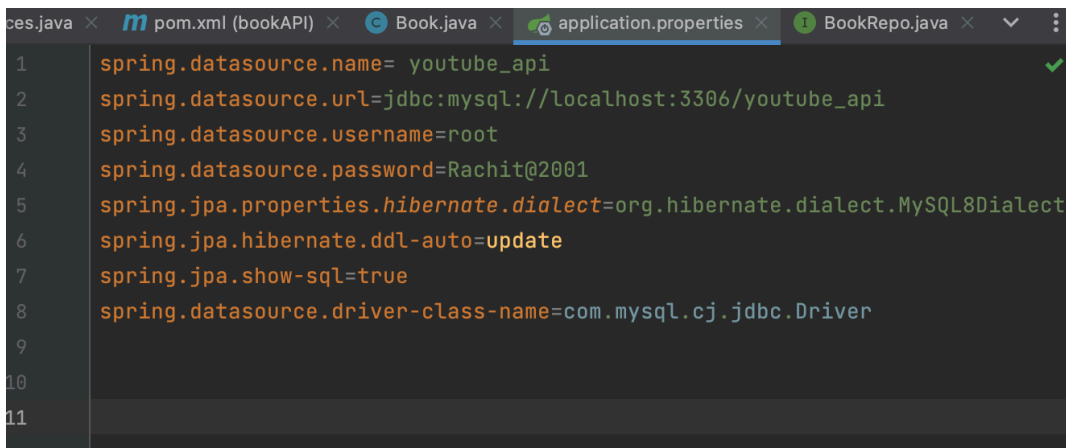


Q1. Using Postman to access data from the database using Spring Boot -

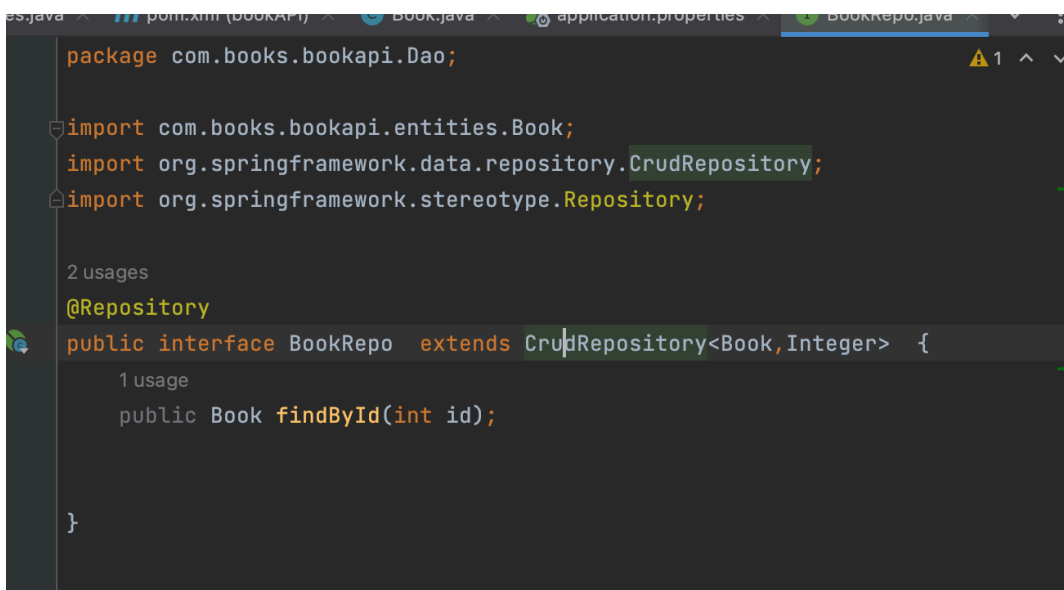
Application.properties -

```
spring.datasource.name= youtube_api  
spring.datasource.url=jdbc:mysql://localhost:3306/youtube_api  
spring.datasource.username=root  
spring.datasource.password=XXXXXX  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.show-sql=true  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

A screenshot of an IDE showing the 'application.properties' file. The file contains database configuration for a Spring Boot application. The properties are: spring.datasource.name= youtube_api, spring.datasource.url=jdbc:mysql://localhost:3306/youtube_api, spring.datasource.username=root, spring.datasource.password=Rachit@2001, spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect, spring.jpa.hibernate.ddl-auto=update, spring.jpa.show-sql=true, and spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver. A green checkmark is visible at the end of the first line.

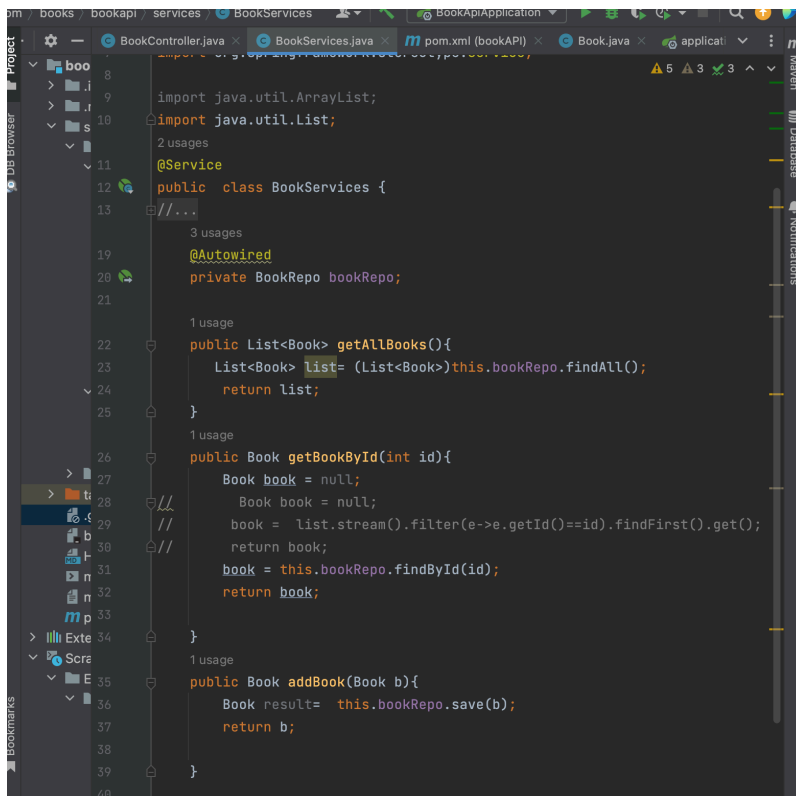
```
1 spring.datasource.name= youtube_api  
2 spring.datasource.url=jdbc:mysql://localhost:3306/youtube_api  
3 spring.datasource.username=root  
4 spring.datasource.password=Rachit@2001  
5 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect  
6 spring.jpa.hibernate.ddl-auto=update  
7 spring.jpa.show-sql=true  
8 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
9  
10  
11
```

Repository -

A screenshot of an IDE showing the 'BookRepo.java' file. The code defines a repository interface. It includes imports for 'com.books.bookapi.entities.Book', 'org.springframework.data.repository.CrudRepository', and 'org.springframework.stereotype.Repository'. The interface is annotated with '@Repository' and extends 'CrudRepository<Book, Integer>'. It contains a single method 'findById(int id)' which returns a 'Book' object.

```
package com.books.bookapi.Dao;  
  
import com.books.bookapi.entities.Book;  
import org.springframework.data.repository.CrudRepository;  
import org.springframework.stereotype.Repository;  
  
2 usages  
@Repository  
public interface BookRepo extends CrudRepository<Book,Integer> {  
    1 usage  
    public Book findById(int id);  
}
```

Service -



```
import java.util.ArrayList;
import java.util.List;

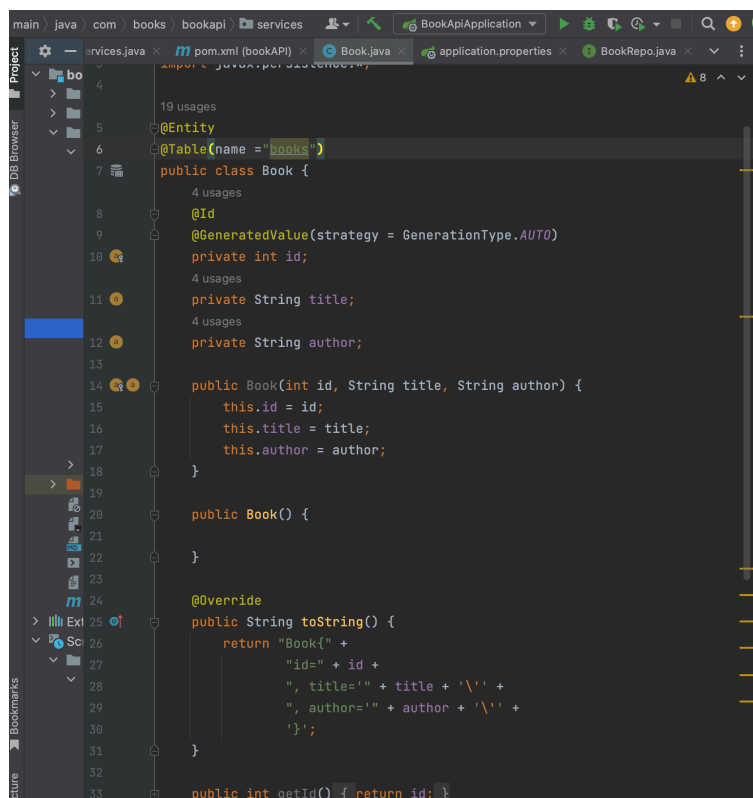
@Service
public class BookServices {
    //...
    @Autowired
    private BookRepo bookRepo;

    public List<Book> getAllBooks(){
        List<Book> list= (List<Book>)this.bookRepo.findAll();
        return list;
    }

    public Book getBookById(int id){
        Book book = null;
        Book book = null;
        //
        //
        book = list.stream().filter(e->e.getId()==id).findFirst().get();
        return book;
        book = this.bookRepo.findById(id);
        return book;
    }

    public Book addBook(Book b){
        Book result= this.bookRepo.save(b);
        return b;
    }
}
```

Entity -



```
@Entity
@Table(name = "books")
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private int id;
    private String title;
    private String author;

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

    public Book() {
    }

    @Override
    public String toString() {
        return "Book{" +
            "id=" + id +
            ", title=" + title + '\'' +
            ", author=" + author + '\'' +
            '}';
    }

    public int getId() { return id; }
}
```

Demonstration - <https://drive.google.com/drive/folders/11tgXHhwyqQ3Sr5jpSovGMb1AcP3WvnOa?usp=sharing>