```
import java.util.List;
import java.util.ArrayList;
import org.springframework.hateoas.Link;
import org.springframework.hateoas.server.mvc.WebMvcLinkBuilder;
import org.springframework.web.bind.annotation.*;

@RestController
@RequestMapping("/books")
public class BookController {

    private final BookRepository bookRepository = new BookRepository();

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

    @GetMapping("/{id}")
    public Book getBook(@PathVariable Long id) {
        Book book = bookRepository.findById(id);
        if (book != null) {
            addHateoasLinks(book);
        }
        return book;
    }

    @GetMapping
    public List<Book> getAllBooks() {
        List<Book> books = bookRepository.findAll();
        books.forEach(this::addHateoasLinks);
        return books;
    }

    @PutMapping("/{id}")
    public Book updateBook(@PathVariable Long id, @RequestBody Book book)
        Book existingBook = bookRepository.findById(id);
        if (existingBook != null) {
            existingBook.setTitle(book.getTitle());
            existingBook.setAuthor(book.getAuthor());
            bookRepository.save(existingBook);
```

```
            addHateoasLinks(existingBook);
            return existingBook;
        }
        return null;
    }

    @DeleteMapping("/{id}")
    public String deleteBook(@PathVariable Long id) {
        boolean deleted = bookRepository.delete(id);
        return deleted ? "Book Deleted Successfully" : "Book Not Found";
    }

    private void addHateoasLinks(Book book) {
        Link selfLink = WebMvcLinkBuilder.linkTo(WebMvcLinkBuilder.methoc
        Link updateLink = WebMvcLinkBuilder.linkTo(WebMvcLinkBuilder.meth
        Link deleteLink = WebMvcLinkBuilder.linkTo(WebMvcLinkBuilder.meth
        book.add(selfLink, updateLink, deleteLink);
    }

    private void addHateoasLinks(List<Book> books) {
        books.forEach(this::addHateoasLinks);
    }
}

class Book {
    private Long id;
    private String title;
    private String author;

    // Getters, setters, and constructors
    public Book(Long id, String title, String author) {
        this.id = id;
        this.title = title;
        this.author = author;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }
```

```
    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public void add(Link... links) {
        // Logic to add links to the book object
    }
}

class BookRepository {
    private final List<Book> books = new ArrayList<>();
    private long currentId = 1;

    public void save(Book book) {
        book.setId(currentId++);
        books.add(book);
    }

    public Book findById(Long id) {
        return books.stream().filter(book -> book.getId().equals(id)).fir
    }

    public List<Book> findAll() {
        return books;
    }

    public boolean delete(Long id) {
        return books.removeIf(book -> book.getId().equals(id));
    }
}

public class BookstoreApplication {
```

```
    public static void main(String[] args) {
        BookController controller = new BookController();

        // Add a new book
        Book book = new Book(null, "The Great Gatsby", "F. Scott Fitzgera
        controller.addBook(book);
        System.out.println("Book Added: " + book);

        // Update the book
        book.setAuthor("F. Scott Fitzgerald (Updated)");
        controller.updateBook(book.getId(), book);
        System.out.println("Book Updated: " + book);

        // Get all books
        List<Book> books = controller.getAllBooks();
        books.forEach(b -> System.out.println("Book: " + b));

        // Delete the book
        controller.deleteBook(book.getId());

        // Attempt to retrieve the deleted book
        Book deletedBook = controller.getBook(book.getId());
        if (deletedBook == null) {
            System.out.println("Attempting to Retrieve Deleted Book: Book
        }
    }
}
```