**Project Report**

**Book Store API**

**Project Title**: RESTful BookStore API

**Intern Name**: Daulatram kushwah

**Company:** Elevate Labs (Internship)

**Certificate**

This is to certify that [Your Name] has successfully completed the internship project titled “RESTful BookStore API” at Elevate Labs under my guidance.

**Acknowledgement**

I would like to express my sincere gratitude to Elevate Labs for providing me with the opportunity to work on this project. I also extend my thanks to my peers and the organization for creating a learning environment that helped me grow my technical and problem-solving skills.

**Abstract**

The BookStore API is a backend application built using Java Spring Boot, MySQL, and Postman for testing. The purpose of this project is to manage books and authors in a bookstore system, providing APIs for CRUD operations, searching, filtering, pagination, and sorting. It also integrates Swagger (OpenAPI) for documentation. The project demonstrates backend development, API design, database integration, and REST principles.

**Table of Contents**

1. Introduction
2. Objectives & Scope
3. System Requirements
4. System Design
5. Implementation
6. Testing & Results
7. Conclusion
8. Future Enhancements
9. References

**Chapter 1: Introduction**

The BookStore API is designed as a learning project during internship at Elevate Labs. It simulates a real-world bookstore system, where authors and books are managed in a structured manner. It follows RESTful design principles and showcases full-stack backend development using Spring Boot and MySQL.

**Chapter 2: Objectives & Scope**

**Objectives:**

* To design and implement RESTful APIs for managing books and authors.
* To provide CRUD operations for both entities.
* To implement search, filtering, pagination, and sorting.
* To test APIs using Postman.
* To generate interactive API documentation using Swagger.

**Scope:**

* Backend-only application.
* Handles book & author management.
* Extendable for future modules like user authentication, reviews, and purchase orders.

**Chapter 3: System Requirements**

**Software Requirements:**

* Java 17
* Spring Boot 3.x
* MySQL 8.0
* Maven
* Postman
* Swagger (OpenAPI 3)
* Hardware Requirements:
* Processor: Intel i5 or above
* RAM: 8 GB
* Storage: 256 GB SSD

**Chapter 4: System Design**

* Database Schema:
* Author
* id (PK)
* name
* biography
* Book
* id (PK)
* title
* genre
* published\_date
* price
* author\_id (FK → Author)

**Architecture:**

* Controller → Service → Repository → Database
* DTOs & Mappers for entity ↔ API separation
* Swagger for API docs

**Chapter 5: Implementation**

* Created Spring Boot project with JPA, Web, MySQL, Swagger dependencies.
* Implemented Author and Book entities with relationships.
* Used DTOs + Mapper for data transfer.
* Built Service Interfaces & Implementations for business logic.
* Developed REST Controllers for API endpoints.
* Added Spring Data JPA Specifications for search, filtering, pagination, sorting.
* Tested endpoints using Postman.
* Documented APIs using Swagger UI.

**Chapter 6: Testing & Results**

* Testing Tools: Postman, Swagger UI
* Create Author: POST /api/authors
* Get All Authors: GET /api/authors
* Create Book: POST /api/books
* Search Books: GET /api/books?q=java&genre=fiction&minPrice=200
* Pagination: GET /api/books?page=0&size=5&sort=price,desc

Result: All endpoints tested successfully and returned expected JSON responses.

**Chapter 7: Conclusion**

The BookStore API was successfully developed with CRUD, search, filter, pagination, sorting, and documentation features. It enhanced my understanding of backend development, RESTful APIs, database integration, and testing.

**Chapter 8: Future Enhancements**

* Add User Authentication & Authorization (JWT).
* Implement Book Reviews & Ratings.
* Add Order & Payment Management.
* Deploy on Cloud (AWS/Azure/Heroku).

**Chapter 9: References**

* Spring Boot Documentation: https://spring.io/projects/spring-boot
* Spring Data JPA: https://spring.io/projects/spring-data-jpa
* Swagger / OpenAPI: https://swagger.io/
* MySQL Documentation: https://dev.mysql.com/doc/