



SPRINTS  
BANQUE MISR ROWAD 2025 INTERNSHIP PROGRAM  
BACK-END DEVELOPMENT WITH JAVA SPRING BOOT  
SUMMER 2025

---

# BookBazaar Documentation

---

**Prepared By:**

Team 4  
Abdelkawy Nasr Badr  
Mohamed Nabil  
Mohammed Yasser  
Omar Gamal  
Omar Mahmoud  
Omar Safwat

**Under the supervision of:**

Eng. Abdelaziz Ragab

# Contents

<b>1</b>	<b>Project Overview &amp; Goals</b>	<b>2</b>
<b>2</b>	<b>System Setup Instructions</b>	<b>2</b>
2.1	MySQL Setup . . . . .	2
2.2	MongoDB Setup . . . . .	3
<b>3</b>	<b>How to Run Project Scripts</b>	<b>4</b>
3.1	Running <code>schema.sql</code> . . . . .	4
3.2	Running <code>seed.sql</code> . . . . .	5
3.3	Running <code>crud_demo.sql</code> . . . . .	5
3.4	Running <code>crud_reviews.js</code> . . . . .	6
<b>4</b>	<b>Database Schema &amp; Structure</b>	<b>7</b>
4.1	MySQL Schema & ER Diagram . . . . .	7
4.2	MongoDB <code>reviews</code> Collection Structure . . . . .	8
<b>5</b>	<b>Troubleshooting Common Issues</b>	<b>9</b>
5.1	Authentication Issues . . . . .	9
5.2	Port Conflicts . . . . .	9
<b>6</b>	<b>GitHub Repository</b>	<b>9</b>

# 1 Project Overview & Goals

**BookBazaar** is a demonstration project that showcases a **polyglot persistence** architecture. This approach uses multiple database technologies to handle different data requirements within a single application.

The primary goals of this project are:

- **Model Relational Data:** To use a traditional SQL database (MySQL) for core, structured, and transactional data such as books, authors, and users.
- **Model Non-Relational Data:** To use a NoSQL database (MongoDB) for data that is less structured and more dynamic, such as user-submitted reviews.
- **Demonstrate CRUD Operations:** To provide clear, executable scripts that perform Create, Read, Update, and Delete (CRUD) operations on both database systems.

By combining MySQL and MongoDB, BookBazaar leverages the strengths of both relational and document-oriented databases for a robust and flexible solution.

## 2 System Setup Instructions

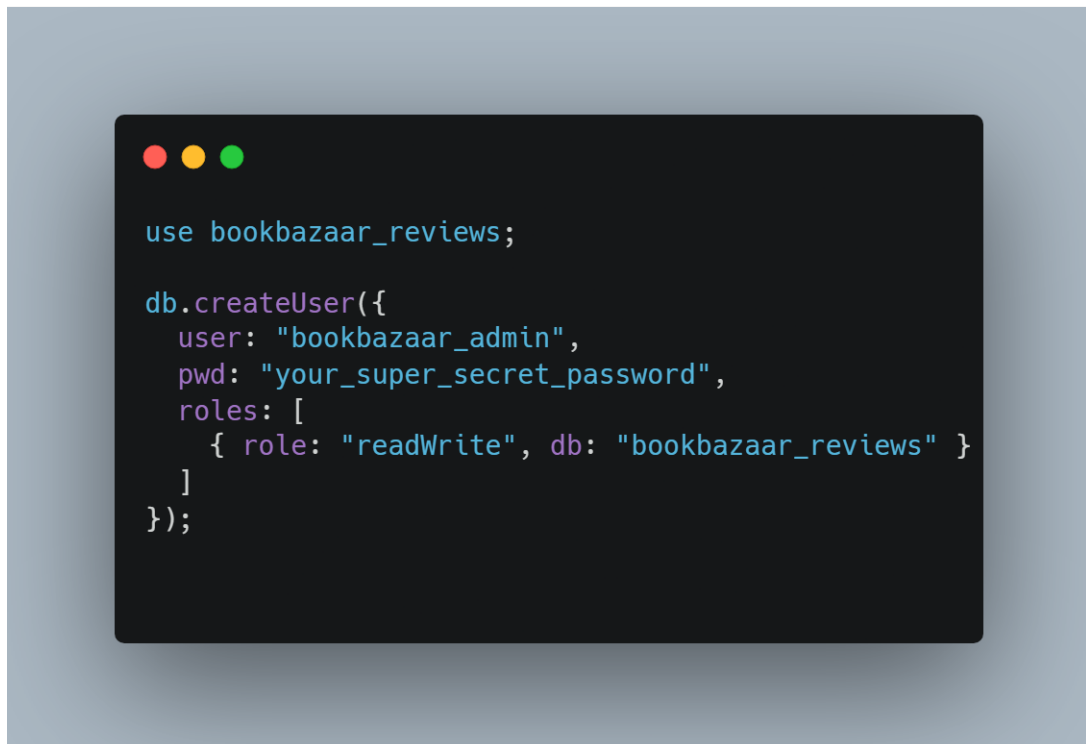
These instructions are designed to be operating system-agnostic.

### 2.1 MySQL Setup

1. **Download & Install:** Download the MySQL Community Server from the official MySQL website. Follow the installation guide for your OS and set a password for the ‘root’ user.
2. **Start the MySQL Service:** Ensure the MySQL server is running on your system.
3. **Prepare for Schema Script:** No manual database creation is needed. The ‘schema.sql’ script will handle the creation of the database (‘bookbazaar’) and a dedicated user (‘bookadmin’). You only need access to a user with ‘CREATE DATABASE’ privileges (like ‘root’) to run it initially.

## 2.2 MongoDB Setup

1. **Download & Install:** Download the MongoDB Community Server from the official MongoDB website and follow the installation instructions.
2. **Start the MongoDB Service:** Ensure the MongoDB service ('mongod') is running.
3. **Create Database and User:** Open the MongoDB Shell ('mongosh') and run the commands shown in Figure 1 to create the 'bookbazaar-reviews' database and a user with 'readWrite' permissions.

A terminal window with a dark background and light blue text. The window has three colored window control buttons (red, yellow, green) in the top-left corner. The text inside the terminal is as follows:


```
use bookbazaar_reviews;  
  
db.createUser({  
  user: "bookbazaar_admin",  
  pwd: "your_super_secret_password",  
  roles: [  
    { role: "readWrite", db: "bookbazaar_reviews" }  
  ]  
});
```

Figure 1: MongoDB Database and User Creation

## 3 How to Run Project Scripts

### 3.1 Running schema.sql

This script creates the ‘bookbazaar’ database, its tables, a dedicated user (‘bookadmin’), and inserts initial sample data. You must have root (or equivalent) access to your MySQL instance to run it.



```
CREATE DATABASE bookbazaar
  DEFAULT CHARSET = utf8mb4
  COLLATE = utf8mb4_unicode_ci;

CREATE USER 'bookadmin'@'localhost' IDENTIFIED BY 'root account password';

GRANT ALL PRIVILEGES ON bookbazaar.* TO 'bookadmin'@'localhost';

use bookbazaar;

CREATE TABLE authors (
  author_id INT PRIMARY KEY AUTO_INCREMENT,
  first_name VARCHAR(50) NOT NULL,
  last_name VARCHAR(50) NOT NULL
);

CREATE TABLE books (
  book_id INT PRIMARY KEY AUTO_INCREMENT,
  isbn CHAR(15) NOT NULL UNIQUE,
  title VARCHAR(100) NOT NULL,
  genre VARCHAR(50) NOT NULL,
  page_count INT NOT NULL CHECK (page_count > 0),
  cost DECIMAL(10, 2) NOT NULL CHECK (cost > 0)
);

CREATE TABLE users (
  user_id INT PRIMARY KEY AUTO_INCREMENT,
  username VARCHAR(50) NOT NULL UNIQUE,
  phone_number CHAR(11) NOT NULL UNIQUE,
  email VARCHAR(50) NOT NULL UNIQUE,
  password_hash VARCHAR(255) NOT NULL,
  country VARCHAR(30) NOT NULL
)
```

Figure 2: Executing schema.sql

### 3.2 Running seed.sql

This script populates the database tables (**authors**, **books**, and **users**) with their initial sample data. You must run **schema.sql** successfully before executing this script.

It is best to run this command with the **bookadmin** user, as it only requires privileges to insert data, not to create tables.

```
INSERT INTO authors(first_name, last_name) VALUES
('Paulo', 'Coelho'),
('Robert', 'Kiyosaki'),
('Charles', 'Dickens');

INSERT INTO books(isbn, title, genre, page_count, cost) VALUES
('abcde12345fghij', 'The Alchemist', 'Drama', 250, 200),
('lmnop12345andfb', 'Rich Dad Poor Dad', 'Personal Finance', 200, 150),
('xyzandabc123456', 'The Spy', 'Spy Fiction', 300, 180),
('123qrstuvwxyz45', 'Oliver Twist', 'Adventure Fiction', 500, 300),
('cristianoisgoat', 'David Copperfield', 'Autobiography Fiction', 400, 350);

INSERT INTO users(username, phone_number, email, password_hash, country) VALUES
('MohamedNabil500', '01011388226', 'mohameddnabil04@gmail.com', 'wxyz1234abcd', 'Egypt'),
('OmarSafwat500', '01020304050', 'omarsafwat03@gmail.com', 'aa12bb34cc56', 'Egypt'),
('AbdelkawyNasr123', '01000055566', 'abdonasr103@gmail.com', 'qwerty123456', 'Egypt');
```

Figure 3: Executing seed.sql to Populate Data

### 3.3 Running crud\_demo.sql

This script demonstrates CRUD operations on the ‘books’ table. It can be run using the ‘bookadmin’ user created by the schema script.

```
USE bookbazaar;

START TRANSACTION;
INSERT INTO books (isbn, title, genre, page_count, cost) VALUES
('thebookisbn1234', 'Atomic Habits', 'Self-Help', 320, 220);
COMMIT;

SELECT * FROM books WHERE title = 'Atomic Habits';

SELECT * FROM books;

START TRANSACTION;
UPDATE books
SET genre = 'Productivity', title = 'Atomic Habits - Revised'
WHERE isbn = 'thebookisbn1234';
COMMIT;

SELECT * FROM books WHERE isbn = 'thebookisbn1234';

START TRANSACTION;
DELETE FROM books WHERE isbn = 'thebookisbn1234';
COMMIT;

SELECT * FROM books WHERE isbn = 'thebookisbn1234';
```

Figure 4: Executing crud\_demo.sql

### 3.4 Running crud\_reviews.js

This script demonstrates CRUD operations on the 'reviews' collection in MongoDB.



```
use bookbazaar_reviews;

print("Inserting a new review...");
const newReview = {
  book_id: 2,
  reviewer: "Grace Miller",
  rating: 4,
  comment: "Well-written and engaging.",
  created_at: new Date()
};
const insertResult = db.reviews.insertOne(newReview);
print("Inserted review ID:");
printjson(insertResult.insertedId);

print("\nAll reviews:");
const allReviews = db.reviews.find();
allReviews.forEach(doc => printjson(doc));

print("\nReviews with rating >= 4:");
const topReviews = db.reviews.find({ rating: { $gte: 4 } });
topReviews.forEach(doc => printjson(doc));

print("\nUpdating review for Grace Miller...");
const updateResult = db.reviews.updateOne(
  { reviewer: "Grace Miller" },
  { $set: { comment: "Well-written, engaging, and informative." } }
);
print("Update result:");
printjson(updateResult);

print("\nDeleting reviews with rating < 3...");
const deleteResult = db.reviews.deleteMany({ rating: { $lt: 3 } });
print("Delete result:");
printjson(deleteResult);

print("\nFinal state of reviews collection:");
db.reviews.find().forEach(doc => printjson(doc));
```

Figure 5: Executing crud\_reviews.js

## 4 Database Schema & Structure

### 4.1 MySQL Schema & ER Diagram

The relational model was designed to capture the core entities of the bookstore. Figure 6 illustrates the intended relationships.

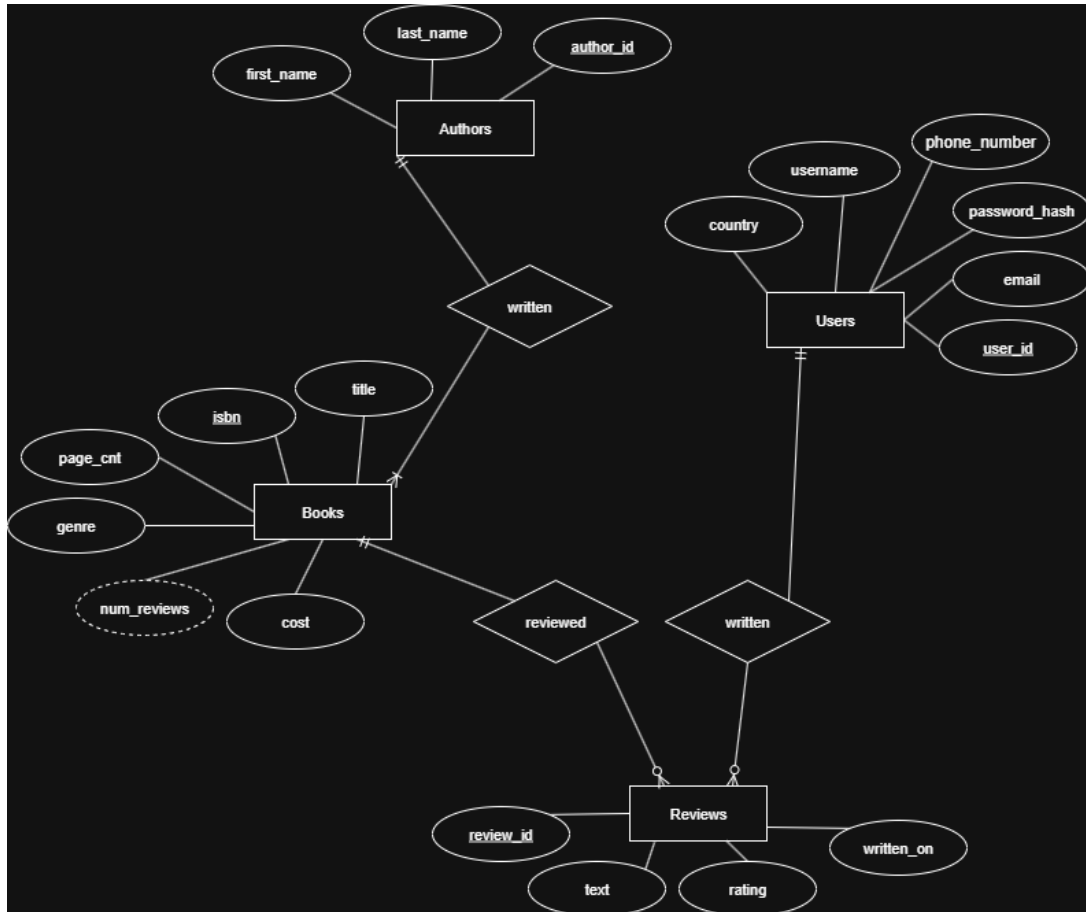


Figure 6: Entity Relationship Diagram for BookBazaar

**Note on Implementation:** The provided 'schema.sql' implements the 'authors', 'books', and 'users' tables. The tables created by 'schema.sql' are defined as follows:



**Table:** authors

Column	Type	Constraints
author_id	INT	PK, AUTO_INCREMENT
first_name	VARCHAR(50)	NOT NULL
last_name	VARCHAR(50)	NOT NULL

**Table:** books

Column	Type	Constraints
book_id	INT	PK, AUTO_INCREMENT
isbn	CHAR(15)	NOT NULL, UNIQUE
title	VARCHAR(100)	NOT NULL
genre	VARCHAR(50)	NOT NULL
page_count	INT	NOT NULL, CHECK (> 0)
cost	DECIMAL(10, 2)	NOT NULL, CHECK (> 0)

## 4.2 MongoDB reviews Collection Structure

The `reviews` collection holds user-submitted reviews. Documents generally follow this structure:

Field	Type	Description
_id	ObjectId	Auto-generated unique identifier.
book_id	int	Foreign key reference to <code>books.book_id</code> in MySQL.
reviewer	string	Name of the reviewer.
rating	int	Numerical rating (e.g., 1-5).
comment	string	Text content of the review.
created_at	ISODate	Timestamp of review creation.

## 5 Troubleshooting Common Issues

### 5.1 Authentication Issues

- **MySQL Symptom:** Error ‘Access denied for user ‘bookadmin’@‘localhost’‘.
- **Cause:** The password for ‘bookadmin’ is set to your ‘root’ password when ‘schema.sql’ is run. If you change your root password, this password will mismatch.
- **Solution:** Log in as ‘root’ and run `ALTER USER ‘bookadmin’@‘localhost’ IDENTIFIED BY ‘new_password’;`.
- **MongoDB Symptom:** Error ‘AuthenticationFailed‘.
- **Cause:** Incorrect credentials or forgetting the authentication database.
- **Solution:** Ensure you use the ‘–authenticationDatabase admin’ flag in your connection command.

### 5.2 Port Conflicts

- **Symptom:** Database service fails to start, with logs mentioning ”address already in use.”
- **Cause:** Another application is using the default port (‘3306’ for MySQL, ‘27017’ for MongoDB).
- **Solution:** Identify and stop the conflicting process or reconfigure the database to use a different port in its configuration file.

## 6 GitHub Repository

The complete source code, including all scripts and documentation, is available on GitHub.

**Repository Link:** <https://github.com/your-username/bookbazaar-project>