Challenges Faced & Solutions Implemented

1. Learning Curve with Node.js & Express

- **Challenge:** I had never used Node.js or Express.js before this project. Picking up a completely new backend framework within a short timeframe was a major challenge.
- **Solution:** I relied on official documentation and YouTube tutorials to quickly learn Express fundamentals such as routing, middleware, validation, and error handling. My background in Java and Laravel helped me draw parallels, making the concepts easier to grasp.
- **Outcome:** I successfully developed a fully functional REST API with authentication, file uploads, and Swagger documentation, all following industry standards.

2. Database Relationships & Constraints

- **Challenge:** Designing relationships (One-to-Many for authors/publishers, Many-to-Many for categories) while ensuring referential integrity was complex. Foreign key constraints also raised challenges when handling deletes.
- **Solution:** I applied proper indexing and foreign key constraints in MySQL. In the API, I implemented validation checks before insert/update/delete to avoid integrity issues.
- **Outcome:** The database schema is normalized, optimized for queries, and prevents accidental orphan records.

3. Search & Pagination Implementation

- Challeng**e:** Implementing search functionality while supporting pagination for performance.
- **Solution:** Used SQL queries with JOINs and LIKE operators for flexible search. For pagination, implemented the standard LIMIT and OFFSET pattern.
- **Outcome:** The API can efficiently handle large datasets while allowing users to filter and navigate results.

4. Authentication & Authorization (JWT)

- **Challenge:** Securing endpoints with authentication and ensuring only authorized users could access protected routes.
- **Solution:** Implemented JWT-based authentication with bcrypt password hashing for secure login. Added a reusable middleware to verify tokens and combined it with validation middlewares.
- **Outcome:** The API enforces strong security, protecting sensitive endpoints such as user profile, book management, and reviews.

5. File Upload for Book Covers

- **Challenge:** Allowing users to upload book cover images while ensuring secure handling and file type validation.
- **Solution:** Integrated Multer middleware for file uploads, enforced file size/type restrictions, and stored covers with unique filenames.
- **Outcome:** The system supports secure and efficient file uploads, preventing issues such as overwriting files or invalid formats.

6. API Documentation

- **Challenge:** Providing clear API documentation for easy testing and team adoption.
- **Solution:** Used Swagger (OpenAPI) annotations to auto-generate interactive API docs covering all endpoints, request/response formats, and authentication.
- **Outcome:** Developers and testers can explore and test endpoints directly from the Swagger UI, improving productivity and collaboration.